Arabian Journal for Science and Engineering (2019) 44:2867–2897
https://doi.org/10.1007/s13369-018-3614-3

REVIEW - COMPUTER ENGINEERING AND COMPUTER SCIENCE

Quality of Service (QoS) Aware Workflow Scheduling (WFS) in Cloud Computing: A Systematic Review

Simranjit Kaur[1] · Pallavi Bagga[2] · Rahul Hans[1] · Harjot Kaur[3]

Received: 11 June 2018 / Accepted: 21 October 2018 / Published online: 3 November 2018

Abstract
Workflow scheduling concerns the mapping of complex tasks to cloud resources by taking into account various Quality of Service requirements. In virtue of continuous proliferation in the exploration of cloud computing, it has become stringent to find the proper scheduling scheme for the execution of workflow under user specifications. Moreover, till date, there exists no systematic review of the existing numerous techniques for this NP-complete problem in the cloud. Taking this into account, the present study seeks to address this gap and spotlights the comprehensive taxonomy of various scheduling schemes as well as extensively compares them by illuminating their objectives, features, merits, and demerits. This paper also highlights the future research challenges with an aim to foster more research in the realm of workflow scheduling as an optimization task.

Keywords Quality of Service · Optimization · Workflow scheduling · Cloud computing

1 Introduction

In the present days, the workflow scheduling (WFS) is considered as one of the eminent issues in distributed computing which allows the mapping of inter-dependent tasks to virtual machines (VMs) in such a way that the execution of workflow application gets completed within the specified Quality of Service (QoS) requirements. Basically, in cloud, the services like Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) are consumed by the users on the basis of Service Level Agreement (SLA) with the help of QoS constraints [1]. In general, a workflow is a flow of complex tasks that are bounded together through dependencies [2]. Moreover, workflows have particular constraints in terms of deadlines, resource utilization, etc., whereas in other schedulings, decision is taken to order the execution of independent tasks which have no relationship with each other. Individual tasks may have priorities or may need specific resources. It includes "e-science" and "e-business" like various complex applications [3]. Probing further, the workflow has a wide range of applications that can be programmed in distributed computing environments like cloud and grid [4]. Moreover, it has been acknowledged from the previous studies that many researchers have been shifted towards cloud computing in order to achieve high performance by virtue of which, this literature review mainly spotlights the survey incorporating WFS issues in cloud environment without neglecting the role of grid computing techniques in WFS.

1.1 Motivation

With the emergence of cloud computing area, the WFS is gaining more consideration. Also, the evidence of this can be found in the literature in form of various surveys. Masdari et al. [2] have reviewed different algorithms like Heuristic, Meta-heuristic and Hybrid by considering numerous QoS constraints but they have just mentioned the basic information of algorithms and have missed the detailed information such as pros and cons of various approaches. On the other hand, Arya et al. [5] have presented a sur-

✉ Pallavi Bagga
pallavibagga315@gmail.com

Simranjit Kaur
93simranjotkaur@gmail.com

Rahul Hans
rahulhans@gmail.com

Harjot Kaur
harjotkaursohal@rediffmail.com

[1] Department of Computer Science and Engineering, DAV University, Jalandhar, Punjab, India

[2] Royal Holloway, University of London, Egham, UK

[3] Guru Nanak Dev University, Regional Campus, Gurdaspur, Punjab, India

vey of WFS algorithms without any knowledge about QoS constraints, Scheduling strategies, types of workflows and different schemes. Therefore, in order to entirely cover up the knowledge of WFS schemes, numerous algorithms, QoS constraints, scheduling strategies, types of workflows, and positive results and loopholes of different algorithms in cloud computing and grid computing (to some extent), it is mandatory to provide the practitioners and researchers an up-to-date state-of-the-art research as well as guide them to figure out relevant studies of their own needs with regard to QoS-based WFS, their challenges, aspects and parameters. As a result, this literature review elucidates a meticulous knowledge of all the above-mentioned areas in a single paper. Specifically, the authors aim at a finer level of granularity in classifying the WFS schemes based on single-objective and multi-objective optimization. The outline of the present study can be fairly explored from Fig. 1, and a list of abbreviations used in this literature review with interpretations can be seen in Table 1.

1.2 Systematic Review Process

In this subsection, a clear description of all the phases for conducting this review has been discussed, which explains how the existing research is identified, evaluated and interpreted with regard to a specific topic or an area of interest. Firstly, the research questions are defined to evaluate whether the objectives of present study will be achieved or not. Subsequently, the quest approach is devised to maximize the possibility of discovering relevant research results as well as some segregate criteria are made to include or exclude particular research articles from the review process. Finally, all the data are collected to relate in a meaningful way and are synthesized to answer all the research questions.

1.2.1 Research Questions (RQ)

While planning this review, a set of research questions were framed which are listed as follows, and the answers to which will be provided by the subsequent sections.

RQ 1. What are the gaps in existing workflow scheduling approaches?
RQ 2. Which workflow scheduling algorithms perform better for application specific QoS constraints?
RQ 3. Which existing heuristic, meta-heuristic or hybrid techniques do support WFS?
RQ 4. Why researchers are shifting towards new scheduling strategies?
RQ 5. What do numerous future perspectives exist in the WFS?

1.2.2 Quest Approach (QA)

For conducting this review, a quest approach has been devised in such a way that it results in the wide-ranging and unprejudiced solutions from the literature related to different RQs. The relevant publications related to cloud computing, optimization, workflow, scheduling, heuristic techniques, meta-heuristic and hybrid optimization techniques are leveraged. For the sufficient scope of each quest area, the alternative words of each term as well as the abbreviations are also extensively explored during the review process.

1.2.3 Sample Segregation Strategy (SSS)

This strategy is essential to identify the aptness and unsuitability of existing research articles for addressing the research questions. Almost 300 papers from the year 2000 to year 2018 have been reviewed out of which nearly 150 are included that relate to Workflow Scheduling, Optimization Problem, Types of Scheduling Problems, Task Scheduling in Cloud, QoS constraints, Heuristic Scheduling Schemes, Meta-heuristic Scheduling schemes and Hybrid Schemes , whereas the researches related to scheduling schemes used specifically in operating system and parallel computing environment are wholly excluded from the review process.

1.2.4 Data Elicitation and Organization (DEO)

During DEO activity, the comprehensive systematic Tables 3, 4 and 5 are created to record as well as correlate the elicited information, and further organize the suitable and sufficient information for answering the targeted RQs.

1.2.5 Target Audience

This survey is intended for readers who are interested to acquaint with the techniques for scheduling workflows specifically based on QoS constraints. Also, the researchers with a wide variety of backgrounds in, for example, Distributed Computing, Cloud Computing, Big Data, Hadoop, and Parallel Computing may get profit by learning how workflows can be scheduled using techniques inspired by the past many decades of research. Furthermore, the experts in meta-heuristic and heuristic optimization techniques may also be lured by the comprehensive description of the roles of these techniques in cloud computing as diverse workflow scheduling schemes.

1.2.6 Organization of Paper

The rest of the paper is organized as: Sect. 2 engenders the knowledge about the variants of WFS strategies and the QoS constraints of workflow. Subsequently, Sect. 3 describes the

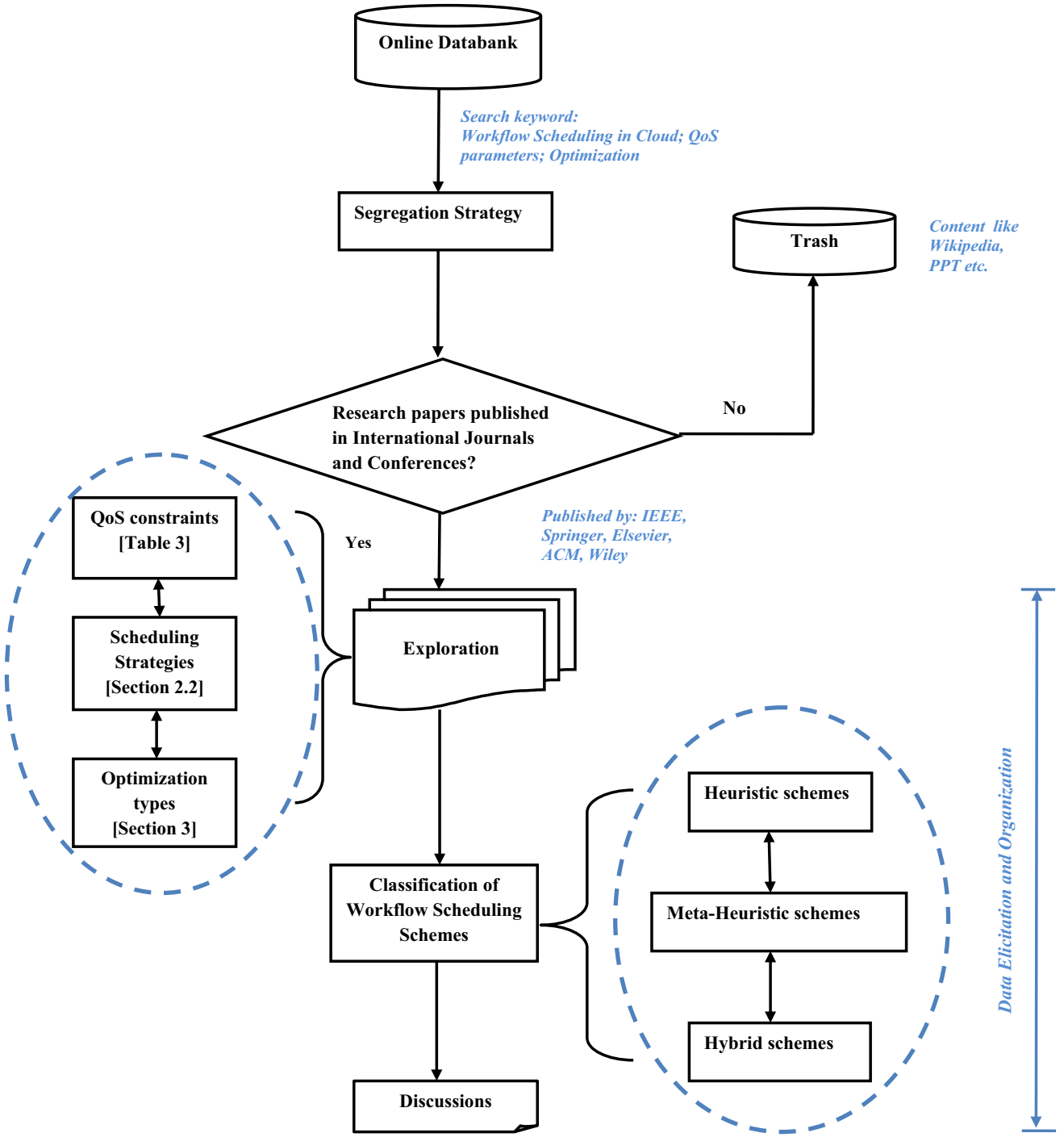

Fig. 1 An outline of systematic literature review

WFS as an optimization problem. Further, Sect. 4 throws light upon the Classification and Scheduling Schemes of Workflow scheduling, whereas Sect. 5 adds discussions and makes comments over the strengths and limitations of this review. Lastly, some concluding remarks and future directions are presented in Sect. 6.

2 Workflow Scheduling

2.1 Workflow

Workflow is the execution and automation of an orchestrated and repeatable pattern of business processes where tasks, information or documents are passed from one partici-

Table 1 Abbreviations with their interpretation

Abbreviation	Interpretation	Abbreviation	Interpretation
WFS	WorkFlow Scheduling	PaaS	Platform as a Service
IaaS	Infrastructure as a Service	SaaS	Software as a Service
QoS	Quality of Service	ASJS	Adaptive Scoring Job Scheduling
RQ	Research Questions	QA	Quest Approach
SSS	Sample Segregation Strategy	DEO	Data Elicitation and Organization
LIGO	Laser Interferometer Gravitational Wave Observatory	QDA	QoS-based Deadline Allocation
SIPHT	sRNA identification Protocol using High Throughput technology	SABA	Security Aware and Budget Aware
SLA	Service Level Agreement	MOP	Multi-objective optimization
VM	Virtual Machine	WMTG-min	Weighted Mean execution Time Guided-minimum
NP	Non-probabilistic	WMTSG-min	Weighted Mean execution Time Sufferage Guided-minimum
DAG	Directed Acyclic Graph	SHEFT	Scalable Heterogeneous Earliest Finish Time
SOP	Single-objective optimization	HCOC	Hybrid Cloud Optimized Cost
HEFT	Heterogeneous Earliest Finish Time	3S	Super-job Static Scheduling
PCP	Partial Critical Path	MOLS	Multi-Objective List Scheduling
EDF_BF_IC	Earliest Deadline First Best Fit with Imprecise Computation	SC-PCP	SAAS Cloud Partial Critical Path
BDA	Bi-direction Adjust heuristic	PBSA	Priority Based Scheduling Algorithm
PCH	Path Clustering Heuristic	TOF	Transformation based Optimization Framework
ADOS	Adaptive Dual Objective Scheduling	MCPCPP	Multi Cloud Partial Critical Path with Pretreatment
HHSA	Hyper Heuristic Scheduling Algorithm	BOSS	Bi-Objective Scheduling Strategy
IOO	Iterative Ordinal Optimization	LJFN	Longest Job on Fastest Node
CEVAET	Cost-Effective Virtual Machine Allocation Algorithm within Execution Time Bound	SJFN	Shortest Job on Fastest Node
PSO	Particle Swarm Optimization	LAGA	Look Ahead GA
RDPSO	Revised Discrete Particle Swarm Optimization	SLPSO	Self adaptive Learning PSO
BPSO	Bi-criteria priority based PSO	DBD-CTO	Deadline and Budget Distribution based Cost Time Optimization
GA	Genetic Algorithm	BDLS	Bi-objective Dynamic Level Scheduling
DOGA	Dynamic Objective GA	TS	Tabu Search
MOGA	Multi-Objective GA	SMS	Stochastic Multi-stage
CSO	Cat Swarm Optimization	SPEA	Strength Pareto Evolutionary Algorithm
ACO	Ant Colony Optimization	NSGA	Non-dominated Sorting Genetic Algorithm
ABC	Artificial Bee Colony	LDDLS	Levelwise Deadline Distributed Linewise Scheduling
MQMW	Multiple QoS constrained scheduling strategy for Multiple Workflows	HSGA	Hybrid Heuristic Scheduling based on GA
QSMTS_IP	QoS-based meta-task scheduling for time-invariant resource usage penalties	RHDPSO	Rotary Hybrid Discrete PSO
SARS	Self Adaptive Reduce Scheduling	BGA	Bi-objective GA
NS	Not Specified	FCFS	First Come First Serve
CGS	Chaos- Genetic Scheduling	MOEA	Multi-Objective Evolutionary Algorithm
ACS	Ant Colony System	PAES	Pareto Archived Evolution Strategy
MDP	Markov Decision Process	RD	Reliability Driven
S-CLPSO	Set-based Comprehensive Learning PSO	CDCGA	Cloud Deadline Coevolutional GA
VNS	Variable Neighbourhood Search	DWSGA	Dynamic Bi-objective Schedule based on GA
MMGWS	Minimum Makespan Grid Workflow Scheduling	BDHEFT	Budget and Deadline constrained Heterogeneous Earliest Finish Time

Table 1 continued

Abbreviation	Interpretation	Abbreviation	Interpretation
CPOP	Critical Path on a Processor	LDD-LS	Levelwise Deadline Distributed Linewise Scheduling
MER	Maximum Effective Reduction		

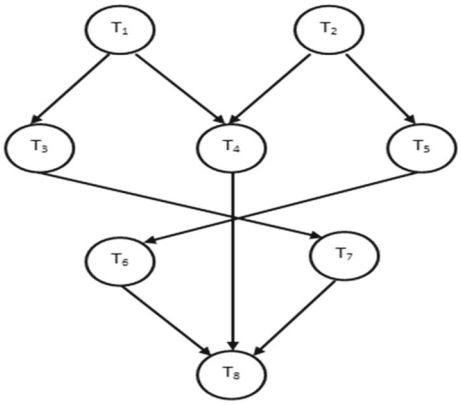

Fig. 2 Simple workflow DAG

pant to other for specific action. Corporations use workflows to coordinate tasks between people and synchronize data between systems, with the aim of enhancing corporational efficiency, responsiveness and profitability [6]. Workflow applications are generally represented as a DAG. A DAG $(D = T, E)$ is a graph having dependencies between the tasks, in which no child task can be executed until all its parent tasks have completed their execution successfully [3]. Figure 2 demonstrates a DAG: $D = (T, E)$ where, T is a set of tasks $(\boldsymbol{T_1}, \boldsymbol{T_2}, \boldsymbol{T_3}, \ldots, \boldsymbol{T_n})$ and E is the set of edges $(\boldsymbol{E1}, \boldsymbol{E2}, \boldsymbol{E3}, \ldots, \boldsymbol{Em})$ that indicates the dependencies between the tasks [7]. Workflow scheduling is one of the key issues in the management of workflow execution. Basically, the workflow scheduling interprets the execution of workflows on well-suited resources by satisfying the QoS requirements [1]. There are basically two types of workflows that are discussed as follows [2]:

1. *Simple workflow* It defines the execution of any simple job in which the set of tasks are represented in form of DAG as shown in Fig. 2.
2. *Scientific workflow* It includes the huge amount of complex data in the form of thousands of tasks that are represented as a DAG [8]. The several real-world examples of workflow on the basis of scientific applications are demonstrated in Fig. 3, which are also briefly discussed in Table 2 [9,10].

2.2 Scheduling

Scheduling is a course of action to deal with the mapping of tasks on well-suited resources with specified user constraints. The classification of scheduling strategies is shown in Fig. 4 [11].

1. *Static* In this, the mapping and scheduling of tasks are completed before the execution of workflow jobs [7]. Also, no run-time fault tolerance and failure recovery is considered in static scheduling.
2. *Dynamic* In this type of scheduling, an information of job components is not known before and the updations in schedule like allocation of resources to incoming tasks, execution time, etc., can be done in run-time. So, the decisions are made in real time [12].
3. *User level* In this, the scheduler manages the problems that are proposed by the service provision between the users and providers. It is efficient where market-based resources are virtualized and delivered to user as a service.
4. *System level* This type of scheduling tackles the resource management within the data centres and it appreciably impacts the performance of data centre.
5. *Online or immediate mode* In this, the scheduling of tasks to specific resource is done at the same time when the scheduler receives it without any delay [13].
6. *Offline or Batch mode* In this, the scheduler gathers the incoming tasks and assigns them to the resources after doing observation at some prescheduled time [13].
7. *Pre-emptive* It defines that the execution of currently on-going task is interrupted on temporary basis with the appearance of any high priority task and the interrupted task will be executed later [14].
8. *Non-pre-emptive* It does not allow the suspension of any task and reschedules it later on [14]. If any task enters the queue, then it has to wait until the current task finishes its execution.
9. *Centralized* In this scheduling type, the master processor unit gathers all the tasks and sends them to other processing units and a dispatch queue is preserved for every processor [15].
10. *Distributed* This type of scheduling involves no central control unit. So, it is the responsibility of local schedulers to handle the incoming requests and give updations to all other processors to maintain their status [16].

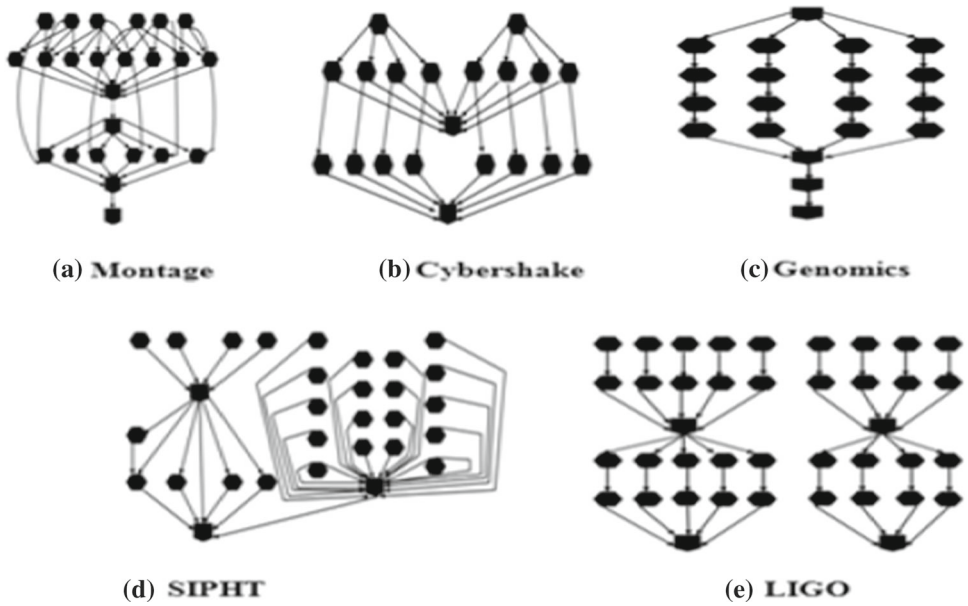

Fig. 3 Pictorial representation of scientific workflows

Table 2 Description of scientific workflows

S. no.	Scientific workflow	Description	Application area
1	Montage	A workflow used to create an image mosaic of sky	Astronomy
2	Cybershake	A workflow used to depict the earthquake threats in an area with the help of Probabilistic Seismic Hazard Analysis method	Earthquake
3	Epigenomics	A workflow interprets the processing of genetic data to implement the different genome sequencing operations	Genetic data
4	SIPHT	A workflow that implements the bioinformatics problems	Bioinformatics
5	LIGO	A workflow used for the identification of gravitational waves and to examine the data attained from compact binary systems	Gravitational works

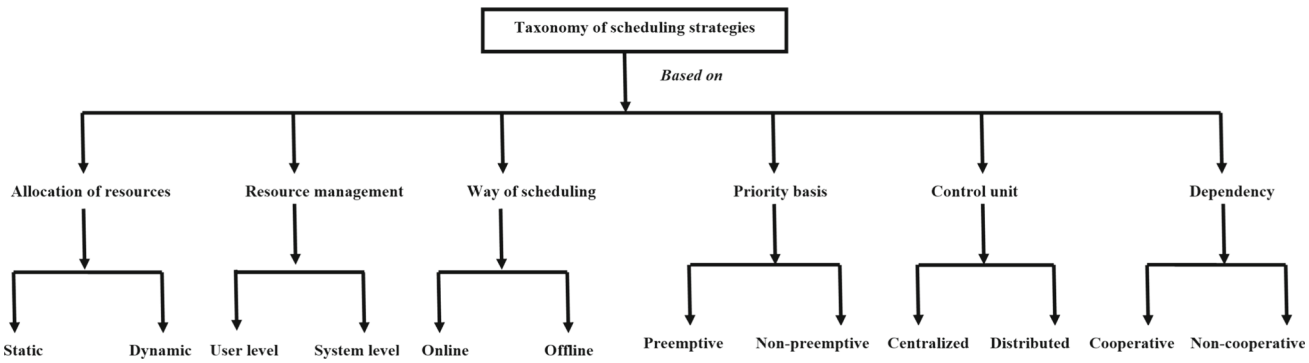

Fig. 4 Different strategies of scheduling

11. *Cooperative* In this, there must be collaboration between all the processors while scheduling to attain the common goal [17]. A cooperative scheduler allows tasks to be scheduled through the use of a periodic timer that creates a system tick and it executes tasks that occur at a time periodic table.
12. *Non-cooperative* While scheduling, the decisions made by every individual processor are independent and do not affect the other processors [17].

2.3 QoS Constraints

It has been interpreted from various workflow scheduling schemes that the predominant challenge in WFS is to satisfy

Table 3 Different QoS constraints for workflow scheduling

S. no.	Constraints	Description
1	Makespan	It defines the time when the execution is started until the completion time of last task in workflow [18]
2	Cost	It defines the price that the users have to pay for scheduling their workload on the resources provided by the cloud providers [19]
3	Throughput	It is the total number of user requests that are completed within a specified time period [20]
4	Reliability	It defines that how many tasks are executed successfully from the total number of tasks with the help of service provided to the user [21]
5	Resource utilization	It defines the proper usage of resources that are provided to the user for scheduling the workload by utilizing the idle time gaps [22]
6	Turnaround time	It defines the difference between the completion time of the task and the time at which the task is submitted [23]
7	Success rate	It defines that how many workflows are executed within the user-defined constraints from the total number of submitted workflows [24]
8	Tardiness	It describes the delay in execution of workflow task means that the completion time of task goes beyond the estimated due time [25]
9	Resource availability	It defines the number of available resources for mapping the tasks to reduce the task failure rate [26]
10	Load balancing	It defines to eschew the burden of cloud resources, a scheduler should optimize the utilization of resources [2]
11	Response time	It defines the time duration between the arrival of task and the completion of task [27]
12	Budget	It defines the cost that is approved by the user for specific time period to get hold of the cloud resources [28]
13	Deadline	It defines the user-specified time bounds within which the workflow should be executed [29]
14	Waiting time	It defines the time elapsed between the ready time of task to the actual initiation of task [30]
15	Execution time	It defines the time taken by the resource to execute the job when it starts executing on the resource [18]
16	Security	It defines a protected scheduling by a secure scheduler to diminish the effects of security attacks that are done by the attackers by misusing the cloud services [2]

the user requirements. QoS constraints are specified by the user as per their requirements for workflow scheduling. The authors' acknowledge thirteen QoS constraints from literature which are described in Table 3.

3 Workflow Scheduling as Optimization Problem

Workflow scheduling in cloud computing is an NP-complete optimization problem. An optimization problem can be stated as:

Let T be the finite set of tasks $T_i (1 \leq i \leq n)$ and S be the finite set of schedules $S_i (1 \leq i \leq n)$, each containing all the tasks from set T. The best workflow schedule S_i is chosen based on the constraint or objective 'O', whether it is to be minimized (Eq. 1) or maximized (Eq. 2), such that

$$S_i\,(O_{\min}) < S_j\,(O_{\min}) \quad \forall i, j \in n, i \neq j \tag{1}$$

$$S_i\,(O_{\max}) > S_j\,(O_{\max}) \quad \forall i, j \in n, i \neq j \tag{2}$$

To attain an optimal solution for WFS problem in cloud and grid environment, there are numerous criteria that can be optimized by exploring different algorithms. All the optimization goals are mentioned in Table 3, and the algorithms which worked on single criteria, bi-criteria and multi-criteria are mentioned in Sect. 4. There are basically two types of optimizations:

- *SOP* A general single-objective optimization problem is defined as minimizing (or maximizing) $O(x)$ objective function. A solution minimizes (or maximizes) the scalar $O(x)$ where x is an n-dimensional decision variable vector $X = (x_1, \ldots, x_n)$ from some universe Ω.
- *MOP* The MOP problems deal with the task of simultaneously optimizing two or more conflicting objectives with respect to a set of certain constraints. Suppose the different objective functions are designated as

$O_1(x), O_2(x), \ldots, O_k(x)$, where k is the number of objective functions in the MOP being solved; and the general objective function is represented by Eq. 3:

$$O(x) = [O_1(x), O_2(x), \ldots, O_k(x)]^{\mathrm{T}} \tag{3}$$

The authors have acknowledged that the researchers are moving towards new scheduling schemes as they do work by considering more than one QoS constraint under different user specifications; and this statement targets the RQ4.

4 Taxonomy of Workflow Scheduling Schemes

Workflow scheduling can be broadly classified into heuristic, meta-heuristic and hybrid schemes as per the authors' literature survey. The classification of workflow scheduling schemes is shown in Fig. 5 with their standard algorithms. Considering RQ3, most of the researchers focus on the following techniques to schedule the workflow in distributed environment:

4.1 Heuristic Algorithms

Heuristic means "to discover by trial and error". So, this category of algorithms includes the algorithms that can provide solutions of an optimization problem in a reasonable amount of time, but there is no guarantee that optimal solutions are reached. This is good when we do not necessarily want the best solutions rather good solutions which can be reached easily [31]. As per authors' knowledge, these algorithms have been widely used by the previous researchers.

- *HEFT* Topcuoglu et al. [32] have used HEFT and CPOP algorithm with fixed number of processors having different configurations. The selection of tasks by HEFT is on the basis of rank strategy while doing sorting of tasks that help in minimizing the earliest finish time and generates feasible results for DAG related issues.
- *HCOC* Bittencourt et al. [33] have presented an algorithm known as HCOC that can execute the workflow in hybrid clouds by increasing the speed of execution and decreasing the cost of execution. HCOC follows two prominent steps: firstly, an "initial schedule" of the workflow is generated with the aid of private cloud. Secondly, comparison of makespan with deadline is done. On the basis of these, tasks and the resources on which they are scheduled or rescheduled are selected. They have concluded that HCOC is robust and can provide efficient results on comparison with greedy algorithms.
- *Qsufferage algorithm* Weng et al. [34] have presented an algorithm Qsufferage for the scheduling of "Bag-of-Tasks" related problems that is created with "independent tasks". Moreover, this application includes the tasks that are waiting for the execution and can be sorted in any order. In addition to this there is no need of "inter-task communication". Qsufferage works on three essential steps that are: "computation of expected execution time of each task on each host", "sufferage value of each task is calculated" and the last is "to find out the task having highest sufferage value and hand over this task to equivalent host". They have concluded that when size of input data is changed, Qsufferage provides efficient results in terms of makespan and response ratio.
- *PCP* Abrishami et al. [35] have proposed an algorithm known as PCP, in order to schedule the workflow while satisfying the QoS constraints. PCP basically works on two stages that are deadline distribution and planning stage. At first stage, "overall deadline of the workflow is distributed on individual tasks", in such a way that the entire execution of workflow is completed before its specific deadline. In next stage, "planner" is used for the selection of low-priced resource for every task that meet up its subdeadline. The intention of PCP is to arrange the workflow in such a way that it will satisfy the requirements of user and bring out the results at low price.
- *SHEFT* Lin et al. [36] have proposed this algorithm that extends the HEFT algorithm. They have included the concept of clustering and evaluate their proposed algorithm by doing simulation. In clustering, resources having the similar "computing capability" are grouped to form a cluster. Moreover, all the resources are partitioned in distinct number of clusters. SHEFT is used after "task prioritizing" to map the given workflow on "bounded number of processors". The results have concluded that the performance of SHEFT is more effective than HEFT in terms of makespan and also provision dynamic resources within workflow execution.
- *MOLS* Fard et al. [37] have proposed an algorithm that is based on static list scheduling algorithm known as MOLS. They have applied a dual approach that is to maximize the "distance to constraint vector" for dominating solutions and to minimize it if not and evaluated their proposed approach for four different objectives that are execution time, cost, reliability and energy consumption. They have divided MOLS algorithm into three stages: "Constant vector partitioning", "Activity ordering" and "Activity mapping" and concluded that MOLS creates efficient solutions than "Bi-criteria scheduling heuristic" and "Bi-criteria GA".
- *PCH* Bittencourt et al. [38] have used the PCH that is the combination of clustering and list scheduling heuristics. In PCH, "clustering scheme" is used to generate

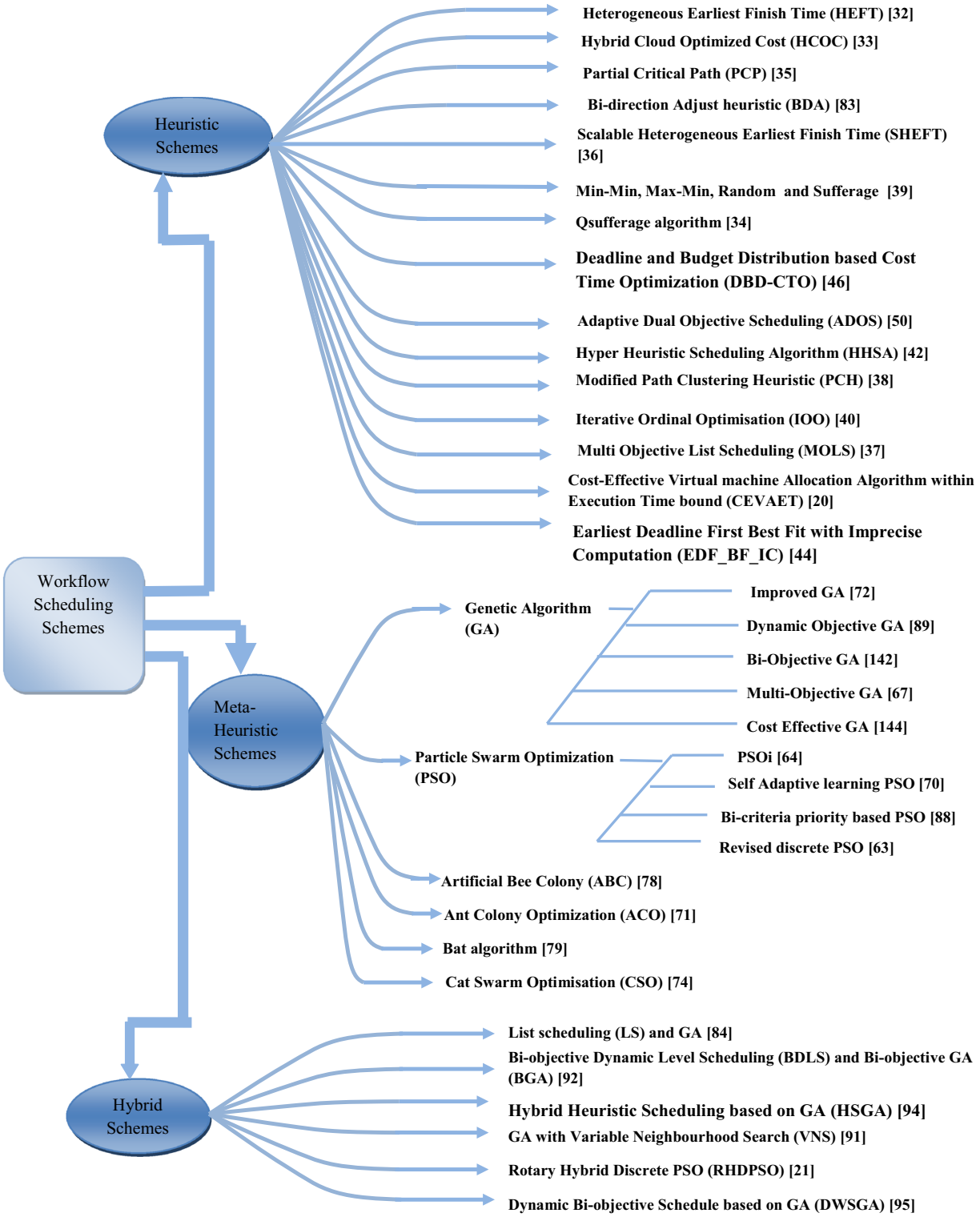

Fig. 5 Classification of scheduling schemes

clusters of tasks. However, "list scheduling heuristics" is used for the selection of tasks and resources. They have used it for more than one workflow in which they have not done any biasing while allocating the processors to all workflows for scheduling and apply four different approaches: "Sequential", "Gap search", "Interleave" and "Group DAGs" that are used for scheduling of multiple workflows and evaluated it on the basis of makespan and fairness and also analyzed that less work is done in scheduling of multiple workflows.

- *Min–Min, Max–Min, Random, Sufferage and HEFT* Lopez et al. [39] have examined these five heuristic algorithms while using static and dynamic scheduling schemes. For the evaluation of all these heuristic algorithms, they have considered some features that are "Number of machines", "DAG" and "Variation" and examined the sensitivity to inaccurate completion time of all these heuristics.
- *Iterative Ordinal Optimization (IOO)* Zhang et al. [40] have presented a technique that is related to IOO. They have also discussed about Monte Carlo and Blind-Pick techniques and compared these two techniques with the presented IOO. To shrink the search area and to decrease the overhead, IOO is used. In Monte Carlo, feasible schedules are created under "heavy scheduling overhead". They have demonstrated that the performance of IOO is very effective for run-time workflow applications like LIGO.
- *WMTG-min and WMTSG-min* Jinquan et al. [41] have presented these two algorithms in which WMTG-min is used to achieve high throughput. WMTSG-min has done some enhancements in sufferage algorithm, whereas the main motive behind "Sufferage algorithm" is that superior scheduling can be done by "mapping a resource to a task that would suffer most in terms of execution time if that specific resource is not assigned to it". It is concluded that their algorithms outperform in terms of makespan and time complexity.
- *HHSA* Tsai et al. [42] have presented an algorithm known as HHSA that focused on raising the diversity detection operators so that the "intensification" and "diversification" for finding the results have to be improved. The results have concluded that the convergence rate of HHSA is better for mapping the resources to workflow tasks and improves the capability to schedule the problems.
- *CEVAET* Zhu et al. [20] have proposed an algorithm that is CEVAET which includes two stages that are described in [20]. At first stage, "topological sorting" is inculcated for the determination of mapping and scheduling is done on the basis of least "End-to-End Delay (EED)". In the next stage, there is an enhancement in the "resource utilization rate" by plummeting the overhead of VM. They have worked on cost and makespan and concluded that their proposed algorithm is effective in terms of both specified objectives.
- *Sufferage Min* Han et al. [43] have presented this algorithm in which they have divided the QoS objectives in two different stages that are "High QoS level" and "Low QoS level". To bolster this algorithm, at first step, sufferage number is computed for every task before the execution of scheduling and sorting is done according to maximum sufferage value. Sufferage value is calculated by subtracting the "next earliest completion time with the earliest completion time". Moving ahead, at second step, "min–min approach" is applied. They have compared the Sufferage Min algorithm with QoS Guided Min–Min heuristic and results show that Sufferage Min provide better results than QoS guided Min–Min in terms of execution time.
- *EDF_BF_IC* Stavrinides et al. [44] have used EDF to propose an algorithm known as EDF_BF_IC with two main aims: (i) to assure that the execution will be completed within user-defined time period and (ii) to accomplish the workflow in less time and at low cost. In the imprecise computation scheme, "a real-time application" is used to get intermediate (imprecise) solutions of poorer, however, of satisfiable quality, "when the deadline of application can not meet". They have also examined the QoS constraints and concluded that the proposed strategy generates good results.
- *BDHEFT* Verma et al. [9] have proposed this technique that is an extension of HEFT. In BDHEFT, they have considered both the time and cost under budget and deadline QoS constraints. BDHEFT is divided into two stages that are "Service level scheduling" and "Task level scheduling. They have considered five workflow applications and on the basis of two parameters that are "NSC" and "NSL" they have done the comparison of BDHEFT with BHEFT and concluded that BDHEFT creates better results than BHEFT in terms of makespan and cost under budget and deadline constraints.
- *MQMW* Xu et al. [45] have offered a method that is MQMW to deal with more than one workflow at a time while satisfying all the QoS constraints. In this scheme, the tasks are organized in a specific order and the sorting is done on the basis of "minimum available service number", "least time and cost" and "minimum covariance". They have used this method to enhance the performance of scheduling and the results of MQMW and RANK_HYBD are compared and concluded that their method is effective in terms of execution time and cost.
- *DBD-CTO* Verma et al. have proposed an algorithm in [46] to reduce the cost and time to execute the schedule and have also considered dynamic rescheduling. In DBD-CTO, the first and foremost aim is to find out the accessible services and ask for "QoS parameters of services for each task". After this step, they have started the workflow partitioning and computation of smallest completion time and cost of every task. Moving further, they have done the distribution of "user's overall deadline and budget" on the basis of some rules. At last, a specific resource is selected for the execution of task in such a way that the execution is finished within deadline and budget. It is demonstrated that the proposed algorithm

helps in creating effective schedule in terms of deadline and budget.

- *QSMTS_IP* Dogan et al. [47] have presented an algorithm QSMTS_IP to handle the issues regarding QoS constraints in scheduling and considered the viewpoints of both the users and system in terms of QoS constraints. At the end, they have concluded that their algorithm is efficient in providing requirements to more than one user at the same time.
- *FCFS, Random and Backfill* Hamscher et al. [48] have discussed the different heuristic algorithms like FCFS, Random and Backfill. For evaluation of these algorithms, they have done simulation. In FCFS, the tasks are scheduled and executed in "order of their submission". Moreover, if at present, the machines are not vacant then scheduler should have to wait till the starting of job. In Random algorithm, the selection of subsequent job for scheduling is done on random basis, due to this, the scheduling is "non-deterministic". Apart from this, in random technique, there is no preference for jobs. However, Backfill algorithm is also considered as "out-of-order version of FCFS" that aims to avoid needless idle time that is due to wide-jobs.

Table 4 presents a comparative analysis of different heuristic algorithms that were used by the researchers to handle the problems regarding workflow scheduling also it presents a year-wise systematic review of various heuristic algorithms with their advantages and disadvantages which target RQ1 and RQ2. Also, it incorporates the sources of papers with their citations.

4.2 Meta-heuristic Algorithms

The Meta word means "beyond" or "higher level" and generally, the meta-algorithms perform better than simple heuristics, with the use of certain trade-off of randomization and local search. Here, the randomization provides good solutions to escape from local search, and therefore all meta-heuristic algorithms intend to be suitable for global optimization [31].

- *PSO* Pandey et al. [62] have considered PSO for run-time WFS to find out the global optimum solution of computational and communication costs. For optimized solutions two factors are utilized: one is the specific scheduling heuristic method, whereas other is the PSO used to get optimized results for "task-resource mapping".
- *RDPSO* Wu et al. [63] have proposed a PSO-based algorithm known as RDPSO in which every solution is defined in pairs of task-service set. In this, Greedy Randomized Adaptive Research Procedure (GRASP) is used for initial optimized population of swarms. Then, a three-step procedure is followed to create new location of swarms. At first stage, particles having higher probability are selected from "gbest and pbest". In the next step, because of "discrete property" of scheduling, there are not that much optimized pairs of gbest to produce new location, due to this individuals will learn from its "previous location" and in third step, the tasks that are not mapped should select the resources from other optimized pairs. It is concluded that RDPSO outperforms PSO in terms of cost minimization.
- *PSOi* Thanh et al. [64] have proposed an algorithm known as PSOi that is a new variant of PSO to solve workflow scheduling issue. PSOi includes some strategies that improve the optimal solutions by not getting stuck in local optima. It uses one new method known as "Inverse" for the movement of particles to a new space. Moreover, PSOi assists in updation of particle's location after every iteration.
- *Fuzzy PSO with LJFN and SJFN* Liu et al. have proposed this algorithm in [65] by using LJFN and SJFN heuristics. They have used the concept of swapping, in which LJFN and SJFN heuristics are swapped in alternate manner whenever any new job is allocated to grid nodes. But if grid nodes are more in count than the number of tasks, allocation is done on the basis of FCFS and LJFN. The proposed method helps to create optimal solutions at runtime to minimize the scheduling time and to enhance the efficiency of resource usage.
- *MOEA* Yu et al. [66] have introduced a technique that works on multiple objectives and based on evolutionary algorithms. They mainly focused on three distinct techniques for resolving the "workflow planning execution issue". In first algorithm that is NSGAII, ranking of solutions is done. Moving further, in SPEA, there is a creation of "external archive" for the selection of next generation elements. Furthermore, the algorithm PAES applies the concept of "local search" from existing solutions to create new solutions and comparison is done with the help of "dominance criteria and density estimation" between both the solutions. They have focused on reducing the makespan and cost and demonstrated that the proposed technique is able to provide efficient solutions with more flexibility.
- *MOGA* Singh et al. have presented MOGA in [67] that focused resource provisioning while considering more than one objective like cost, makespan and utilization of resources and also discussed about the idea of "PARETO OPTIMALITY" in which from the group of solutions to multi-objective issue, the Pareto optimal set is "the set of solutions that are not dominated by any other solution in the whole group". The "Pareto-optimal set" gives permission to user for normalization of objective function values. In addition to this, MOGA works on "objective

Table 4 Review of QoS constraints in Heuristic workflow scheduling

Authors' name	Algorithm name	Year	Makespan	Cost	Execution time	Reliability	Utilization	Response time	Budget	Deadline	Throughput	SO/MO	Scheduling strategy	Pros	Cons
Weng et al. [34]	Qsufferage algorithm	2005	✓	✘	✘	✘	✓	✓	✘	✘	✘	MO	Distributed	Outperforms than other heuristic algorithms	Worked well for a fixed experiment condition only
Etminani et al. [49]	Selective (Min–Min and Max–Min) algorithm	2007	✓	✘	✓	✘	✓	✘	✘	✘	✓	MO	Static and dynamic	Outperforms the Min–Min and Max–Min heuristics	Have not worked on deadline, cost of execution and cost of communication
Choon Lee et al. [50]	ADOS	2009	✓	✘	✘	✘	✓	✘	✘	✘	✘	MO	Static and dynamic	Improves resource utilization without sacrificing too much of Makespan	Focused on utilization by ignoring makespan and other objectives
Xu et al. [45]	MQMW	2009	✓	✓	✓	✘	✘	✘	✘	✘	✘	MO	Dynamic	Higher success rate in terms of cost and execution time than RANK-HYBD	Handle less QoS constraints
Bittencourt et al. [51]	PCH	2010	✓	✓	✓	✘	✘	✘	✘	✓	✘	MO	Dynamic	Capable of scheduling workflows by reducing its cost and execution time	Did not deal with budget under hybrid clouds
Bittencourt et al. [33]	HCOC	2011	✓	✓	✓	✘	✓	✘	✓	✓	✘	MO	Dynamic	Can be easily adapted to handle budgets instead of deadlines, this makes it flexible	Does not deal with multiple workflows

Table 4 continued

Authors' name	Algorithm name	Year	Makespan	Cost	Execution time	Reliability	Utilization	Response time	Budget	Deadline	Throughput	SO/MO	Scheduling strategy	Pros	Cons
Lin et al. [36]	SHEFT	2011	✘	✘	✓	✘	✘	✘	✘	✘	✘	SO	Dynamic	Outperform HEFT for optimizing workflow execution time and resources can scale elastically	NS
Zhang et al. [40]	IOO	2011	✘	✘	✘	✘	✘	✘	✘	✘	✓	SO	Dynamic	Has less overhead and IOO upgrades the throughput and reduce memory demand	Slow execution
Abrishami et al. [52]	SC-PCP	2012	✓	✓	✓	✘	✘	✘	✘	✓	✘	MO	Static	Outperforms another highly cited algorithm called deadline-MDP, cost and time both are low with SC-PCP	No support for cloud models like IAAS
Abrishami et al. [35]	PCP	2012	✓	✓	✘	✘	✘	✘	✘	✓	✘	MO	Static	Outperforms other highly cited algorithm called Deadline-MDP. The computation time is very low for decrease cost	Low performance on parallel pipelines

Table 4 continued

Authors' name	Algorithm name	Year	Makespan	Cost	Execution time	Reliability	Utilization	Response time	Budget	Deadline	Throughput	SO/MO	Scheduling strategy	Pros	Cons
Zhu et al. [20]	CEVAET	2012	✘	✓	✓	✘	✓	✘	✘	✘	✓	MO	Dynamic	Reduces VM cost and lower total execution time	No support for real life scientific workflows
Fard et al. [37]	MOLS	2012	✓	✓	✘	✓	✘	✘	✓	✘	✘	MO	Static	The obtained solutions dominate the user-specified constraints	No support for dynamic workflow scheduling
Amalarethinam et al. [53]	MMGWS	2012	✓	✓	✘	✘	✓	✘	✘	✘	✘	MO	Static	Used both static and dynamic scheduling strategy to handle two objectives makespan and resource utilization	Objective cost is not considered in this paper
Fard et al. [54]	BOSS	2013	✓	✓	✓	✘	✘	✘	✘	✘	✘	MO	Dynamic	A better coverage in terms of execution time and cost	Do not considered other parameters
Pawar et al. [55]	PBSA	2013	✘	✘	✓	✘	✓	✘	✘	✘	✘	MO	Dynamic	Performs better than CMMS in resource contention situation	Not able to predict that which VM will be free earlier
Han et al. [43]	Sufferage-Min	2013	✓	✘	✘	✘	✘	✘	✘	✘	✘	SO	Dynamic	Had a significant performance gain in terms of reduced makespan	Performance is good in terms of execution time only

Table 4 continued

Authors' name	Algorithm name	Year	Makespan	Cost	Execution time	Reliability	Utilization	Response time	Budget	Deadline	Throughput	SO/MO	Scheduling strategy	Pros	Cons
Zhou et al. [56]	TOF	2014	✘	✓	✓	✘	✘	✘	✓	✓	✘	MO	Dynamic	Outperforms the auto scaling by 30% for cost optimization and 21 % for execution time optimization	Performance is low for cross-cloud network bandwidth
Tsai et al. [42]	HHSA	2014	✓	✘	✘	✘	✓	✘	✘	✘	✘	MO	Dynamic	The HHSA converges faster than other heuristics	Detection operators are not effective
Zeng et al. [57]	SABA	2014	✓	✓	✓	✘	✘	✘	✓	✘	✘	MO	Static and dynamic	Provides efficient solutions for communication-intensive workflows	Does not consider security, utilization and budget in dynamic environment
Stavrinides et al. [44]	EDF_BF_IC	2015	✓	✓	✓	✘	✘	✘	✘	✓	✘	MO	Dynamic	Outperforms the baseline policy in terms of makespan, cost and deadline	Not good solutions for energy efficiency
Verma et al. [9]	BDHEFT	2015	✓	✓	✓	✘	✘	✘	✓	✓	✘	MO	Static	Outperforms HEFT in terms of cost and makespan	Not good performance, so improvement is needed
Tang et al. [58]	SARS	2015	✘	✘	✓	✘	✘	✓	✘	✘	✘	MO	Dynamic	Effective in minimizing the response time up to 11–29% in comparison to FIFO	Does not deal with hetero-geneous situation

Table 4 continued

Authors' name	Algorithm name	Year	Makespan	Cost	Execution time	Reliability	Utilization	Response time	Budget	Deadline	Throughput	SO/MO	Scheduling strategy	Pros	Cons
Lee et al. [59]	MER	2015	✓	✓	✘	✘	✓	✘	✘	✓	✘	MO	Static	By allowing a small degree of makespan increase there is a reduction in resource usage	Not good results, so resource performance profiling is needed
Lin et al. [60]	MCPCPP	2016	✘	✓	✓	✘	✘	✘	✘	✓	✘	MO	Static	Scheduling in multi clouds perform superior than single cloud under "loose deadline"	Inaccurate execution and data transfer time for big data workflows
Zhu et al. [61]	Stochastic Multi-stage Job scheduling (SMS) with Job collecting and Timetabling methods	2017	✘	✓	✓	✘	✓	✘	✘	✓	✘	MO	Dynamic	Not sensitive to instance parameters and maximize the utilization of rented VMs over specific time period	Problem with non-linear task dependency

variables space" which means it determines the fitness of "allocation cost and scheduling cost" of resources by implying HEFT algorithm. They concluded that MOGA provides optimal solutions of all objectives.

- *RD reputation and LAGA* Wang et al. [68] have proposed RD and LAGA to support reliable scheduling because it is a prominent issue in WFS. So, they have presented the "RD reputation" that depends on time and helps in evaluating the reliability of resources. RD includes failure rate for the reputation of resources at run-time. However, LAGA makes use of RD reputation to get feasible solutions of makespan and reliability by using one more heuristic mechanism of "Resource priority".
- *CGS* Gharooni-fard et al. [69] have presented CGS that uses "chaotic variables" that spread out the solutions and explore in better way within the entire search space to deal with budget and deadline constraints in scheduling. Basic meaning of Chaos is "randomness created by simple deterministic systems". The first and foremost step in CGS is to utilize chaotic variables to create "individuals of subgenerations dispensed ergodically in defined area". This helps in circumventing the premature convergence of solutions. In next step, CGS uses converging features of GA to surmount randomness and generate global feasible solutions.
- *SLPSO* Zuo et al. [70] have proposed a strategy based on PSO; however, PSO already has a benefit of "quick convergence". The proposed method includes different "velocity updating methods" that help to not get trapped in local optima and enhance the results. SLPSO incorporates some policies that are "adaptively adjusting parameters", "designing distinct population topologies", "make use of multi-population in standard PSO", "include bio-inspired methods in basic PSO" and last one is "to combine PSO with some another adaptive schemes". By inculcating these steps, SLPSO creates efficient results within the user-specified QoS constraints.
- *ACO* Liu et al. [71] have proposed a model known as "Service flow scheduling" and for optimization they have considered ACO. ACO is inspired by the foraging behaviour of ants. They have used ACS algorithm for their scheduling issues that consist QoS constraints like reliability, cost, response time and security. ACS mainly focuses on "heuristic information" that provides guidance to ants.
- *Improved GA* Kumar et al. [72] have proposed an improved GA algorithm in which they have used the two most popular heuristic algorithms that are "Max–Min" and "Min–Min" to generate initial population. In "Max–Min", the jobs that are selected for scheduling are arranged on the basis of "maximum-time" and a job having higher value of time is mapped to one of the matching resources. However, in "Min–Min" algorithm, the priority is given to that job which is having minimum running time. After this, there will be an updation in running time of all other jobs. Authors found that the improved GA is more effective in reducing makespan and also in utilization of resources than GA.
- *GA with Fuzzy theory* Javanmardi et al. [73] have proposed a scheme of fuzzy theory that is used for the modification of GA and becomes helpful in minimizing the number of iterations for generating initial population. They have applied this concept of fuzzy theory mainly on two stages that are "crossover" and "fitness evaluation". This scheme is used to assign the resources on the basis of "resource ability" and the "length of jobs". They have shown that it enhances the solutions of cost and makespan.
- *CSO* Bilgaiyan et al. [74] have discussed about CSO for finding optimal solution in terms of cost. Authors have proposed CSO-based algorithm that is encouraged by two types of cats' behaviours: "seeking mode in which cats do not move" and "tracing mode in which cats move towards next best locations with some velocity". By using some steps, algorithm generates optimal schedule for the execution of workflow in less iterations and concluded that proposed algorithm outperforms than PSO in terms of cost having better rate of convergence.
- *GA* Singh et al. have proposed an algorithm in [75] known as "Budget constrained time minimization GA" that is based on GA that helps in reliable WFS by minimizing the failure rate. In basic GA, mainly three steps are followed that are: "Create initial population", "Selection to generate new individuals" and "Evaluation of fitness value". However, in the proposed strategy, GA comprises some more steps including these mentioned steps. Firstly, there is "Encoding of individuals" then, "Creation of initial population". Moving further, there is an "Evaluation" for objective function in addition to which, "selection procedure" is followed. After these stages, "Crossover" and "Mutation" is done before "Termination". Authors have compared the proposed algorithm with min–min and max–min algorithms and concluded that the proposed algorithm decreases the execution time and the scheduling is completed within the budget constraint.
- *PSO with VNS* This technique is proposed by Netjinda et al. [76] that is the combination of 4 procedures: "initialization", "particle string update", "fitness calculation" and "solution selection". Prior to these procedures, there is a necessity to create "particle string" that focused on encoding of promising solutions. VNS is applied in solution selection phase to enhance the value of solutions.
- *S-CLPSO* Chen et al. [77] have proposed this approach on the basis of PSO to handle the user-specified constraints and explained the Set-based PSO approach and how it is appropriate for WFS. In S-CLPSO algorithm,

velocity and position are updated in each iteration. S-PSO technique is applied in S-CLPSO and it seems to be better choice for WFS issues because in S-PSO "service instances in cloud are considered as resource set"and with the aid of S-PSO it is effortless to "accelerate search". Thus, it is concluded that the results produced by S-CLPSO are very promising under tight constraints.

- *ABC* Liang et al. [78] have presented two algorithms based on ABC with some proposed strategies and some rules that basically focused on finding the optimal solution of execution time when the performance ability of cloud VM's is low. The main idea behind ABC is that how group of bees work mutually to find out the food sources (solutions). They have considered more than one workflow and on applying both ABC-I and ABC-II, the results have demonstrated that the performance of ABC-II is more effective than ABC-I in terms of execution time.
- *Bat algorithm* Kaur et al. [79] have presented an algorithm known as Bat algorithm that examined the issues related to workflow scheduling for different objectives such as makespan, cost and reliability. The complete working of bat algorithm is supported by "echolocation nature of virtual bats" that is implied to sense the range of solutions. They have compared the Bat algorithm with "basic randomized evolutionary algorithm" (BREA) and demonstrated that Bat algorithm is more effective in terms of specified budget and other defined QoS constraints.
- *PSO-ACO* Dr. George [80] has projected a new algorithm that is the fusion of PSO and ACO. The proposed algorithm has focused on minimization of cost and time in which PSO is used for the "evaluation of fitness" and ACO finds out the "global optimal solutions". In starting, " initialization of population" is done and then there is an "iterative loop" in which different values are compared on the basis of objective function. After this, the steps are repeated by changing the "velocity and location of particles" till an entire schedule is generated. Moving further, ACO is applied with "global updation procedure" and "task rescheduling" is also handled by ACO as well.
- *PSO and TS* Dr. Sridhar et al. [81] have projected a new algorithm that includes TS and PSO in which PSO is used to perform "global search" and TS performed "local search". The plan behind this hybrid approach is to enhance the solutions both globally and in confined space as well. Moreover, this assist in finding feasible solutions and also prevent the solutions to stuck in local optima.
- *GA-ACO* LIU et al. [82] have planned an algorithm that is the combination of two meta-heuristic algorithms known as "genetic-ant colony" algorithm. Firstly, the benefits of GA is used for "global search" and after going through every step of GA, feasible solutions are created and then these solutions become initial population for ACO. In addition to this, they have utilized the ACO for better "convergence rate". The functioning of the proposed algorithm is explained in the paper with proper policies and plans. Also, Sathish and Reddy [83] have projected GA-ACO to schedule the workflow in grid environment to gratify all the user-specified constraints.

Table 5 presents a comparative analysis of different meta-heuristic algorithms that were used by the researchers to handle the problems regarding workflow scheduling also it presents a year-wise systematic review of various heuristic algorithms with their advantages and disadvantages which target RQ1 and RQ2 and the sources of papers with their citations.

4.3 Hybrid Algorithms

This category includes a blend of heuristic and meta-heuristic algorithms. A year-wise systematic analysis of different hybrid algorithms that have been proposed by the researchers so far (as per the authors' knowledge) for workflow scheduling is shown in Table 6 with their strengths and shortcomings which target RQ1 and RQ2. In addition, the sources of papers are included with their citations.

- *List scheduling (LS) and GA* Loukopoulos et al. [84] have presented an algorithm that includes GA with "MaxMin and MinMin" for the improvement in basic algorithms. They have exploited scheduling heuristics that are "LS-based" and "GA-based" where LS heuristics sort the tasks on the basis of weight function and make a schedule in accordance with their arrangement in sorted list. On the other hand, GA heuristics include both Simple GA (SGA) and Improved GA (IGA). In IGA, they have launched a new crossover scheme known as "node-crossover". Both LS and GA result in the collapsed makespan.

- *GA with VNS* Kardani-Moghaddam et al. [91] have offered an algorithm for the reduction in cost with the reasonable completion time by combining the exploration aptness of GA and the exploitation potential of VNS with the purpose to improve the solutions in the whole population.
- *BDLS and BGA* Dogan and Ozguner [92] have presented "matching and scheduling algorithms" which focus on two objectives "scheduling length and failure probability". On every scheduling step, DLS selects the next task to schedule and the machine on which that task is to be implemented, by finding the ready task and machine pair that has the highest dynamic level. However, this algorithm does not trace the reliability of resources when matching and scheduling decisions are made. So, to

Table 5 Review of QoS constraints in meta-heuristic workflow scheduling

Authors' name	Algorithm name	Year	Makespan	Cost	Execution time	Reliability	Utilization	Response time	Budget	Deadline	Throughput	SO/MO	Scheduling strategy	Pros	Cons
Xu et al. [85]	ANT optimization	2003	✘	✘	✘	✘	✓	✓	✘	✘	✘	MO	Dynamic	Evaluate ANT algorithm in terms of scalability	Low performance
Yu et al. [86]	GA with Markov Decision Process (MDP)	2006	✘	✓	✓	✘	✘	✘	✓	✘	✘	MO	Static	MDP in GA improve convergence rate under low budget condition	Does not consider other QoS constraints like reliability and security
Singh et al. [67]	MOGA	2007	✓	✓	✘	✘	✓	✘	✘	✘	✘	MO	Dynamic	MOGA outperforms the Best effort execution under different variations in size and load	They have considered only provisioning of compute resources
Chen et al. [87]	ACS	2009	✓	✓	✓	✓	✘	✘	✓	✓	✘	MO	Static	Effective performance of ACS	They have not check the performance of their proposed algorithm in cloud environment
Pandey et al. [62]	PSO	2010	✘	✓	✘	✘	✘	✘	✘	✘	✘	MO	Dynamic	PSO attains less cost than Best Resource Selection (BRS)	Does not consider real applications

Table 5 continued

Authors' name	Algorithm name	Year	Makespan	Cost	Execution time	Reliability	Utilization	Response time	Budget	Deadline	Throughput	SO/MO	Scheduling strategy	Pros	Cons
Wu et al. [63]	RDPSO	2010	✓	✓	✘	✘	✘	✘	✓	✓	✘	MO	Dynamic	RDPSO provides feasible solutions	NS
Gharooni-fard et al. [69]	CGS algorithm	2010	✘	✓	✓	✘	✘	✘	✓	✓	✘	MO	Dynamic	Efficient results by avoiding premature convergence	QOS parameters like reliability do not consider
Kumar et al. [72]	Improved GA	2012	✓	✘	✓	✘	✓	✘	✘	✘	✘	MO	Static	Efficient performance of proposed algorithm in terms of utilization and makespan than standard GA	Low performance for cost parameter
Chen et al. [77]	S-CLPSO	2012	✓	✓	✘	✓	✘	✘	✓	✓	✘	MO	Static	S-CLPSO produce efficient solutions while considering QoS constraints	NS
Verma et al. [88]	BPSO	2014	✘	✓	✓	✘	✘	✘	✓	✓	✘	MO	Static	BPSO outperforms than PSO in terms of cost	They have not considered the load of resources
Bilgaiyan et al. [74]	CSO	2014	✘	✓	✘	✘	✘	✘	✘	✘	✘	SO	Static	CSO is effective than PSO in terms of convergence rate.	CSO does not consider more than one objective

Table 5 continued

Authors' name	Algorithm name	Year	Makespan	Cost	Execution time	Reliability	Utilization	Response time	Budget	Deadline	Throughput	SO/MO	Scheduling strategy	Pros	Cons
Singh et al. [75]	GA	2014	✓	✓	✓	✘	✘	✘	✓	✘	✘	MO	Static	GA provides feasible results in terms of makespan and budget	Does not deal with load balancing and resource utilization
Liang et al. [78]	ABC	2014	✓	✘	✘	✘	✘	✘	✘	✘	✘	SO	Static	ABC-II converges faster than ABC-I in terms of makespan	They have not considered large size of workflows
Javanmardi et al. [73]	Modify GA with Fuzzy theory	2014	✓	✓	✓	✘	✘	✘	✘	✘	✘	MO	Static	Modification in GA is effective in terms of cost about 45% and makespan about 50%	Performance is not good on other QoS objectives
LIU et al. [82]	GA-ACO	2014	✘	✘	✓	✘	✘	✘	✘	✘	✘	SO	Static	Solve task scheduling effectively by improving its search efficiency in less time	Efficiency is good only in terms of time
Thanh et al. [64]	PSOi	2015	✘	✘	✓	✘	✘	✘	✘	✘	✘	SO	Static	PSOi is effective in terms of makespan in small scale cloud	Low performance of PSOi to solve large instances in less execution time

Table 5 continued

Authors' name	Algorithm name	Year	Makespan	Cost	Execution time	Reliability	Utilization	Response time	Budget	Deadline	Throughput	SO/MO	Scheduling strategy	Pros	Cons
Chen et al. [89]	DOGA	2015	✘	✓	✓	✘	✘	✘	✘	✓	✘	MO	Static	DOGA is more adaptive to meet the user-specified constraints like deadline	Have not proved its efficiency for dynamic and multiple objectives
Kaur et al. [79]	BAT	2016	✘	✓	✓	✓	✘	✘	✓	✘	✘	MO	Static	BAT outperforms Basic Randomized Evolutionary Algorithm (BREA) in terms of reliability within specified budget	Have not considered the deadline constraint
Lal et al. [90]	ACO	2018	✘	✓	✓	✘	✓	✓	✘	✓	✘	MO	Static	Significant performance in comparison to GA	NS

Table 6 Review of QoS constraints in hybrid workflow scheduling

Authors name	Algorithm	Year	Makespan	Cost	Execution time	Reliability	Utilization	Response time	Budget	Deadline	Throughput	SO/MO	Scheduling strategy	Pros	Cons
Loukopoulos et al. [84]	GA and List Scheduling (LS)	2007	✓	✘	✓	✘	✘	✘	✘	✘	✘	SO	NS	Offer most promising results with minimum node completion time	Dependencies between the tasks are not considered
Tao et al. [21]	RHDPSO	2009	✓	✓	✘	✓	✓	✘	✘	✘	✘	MO	Static	Shows speed of convergence and ability to obtain faster and feasible schedules	Performs well only in grid background
Kardani-Moghaddam et al. [91]	GA-VNS	2012	✘	✓	✘	✘	✘	✘	✘	✘	✘	SO	Static	Performs much better in terms of execution cost	Not suitable for hard deadlines
Delavar et al. [94]	HSGA	2013	✓	✓	✘	✓	✘	✓	✘	✘	✘	MO	Static	Reduces the task execution time, supports load balancing, and reliability	Communication-intensive applications are not considered
Rahman et al. [96]	Dynamic critical path for grid (DCP-G)	2013	✓	✓	✓	✘	✘	✘	✓	✓	✘	MO	Dynamic	Better schedule for most of workflow types in dynamic environment	Do not generate better results for multi-objective functions

Table 6 continued

Authors name	Algorithm	Year	Makespan	Cost	Execution time	Reliability	Utilization	Response time	Budget	Deadline	Throughput	SO/MO	Scheduling strategy	Pros	Cons
Tao et al. [97]	DGWS, Markov chain based resource availability prediction model	2013	✓	✘	✓	✓	✘	✘	✘	✓	✘	MO	Static	Results are promising in terms of reliability, completion time	Do not consider fault tolerant
Sridhar et al. [81]	Hybrid PSO	2015	✓	✘	✓	✘	✘	✘	✘	✘	✘	SO	Dynamic	Performs better in terms of schedule length	NS
Chen et al. [98]	Cost and privacy aware method based on GA (CP-GA)	2015	✓	✓	✘	✘	✘	✘	✘	✘	✘	MO	Static	CP-GA minimizes user cost with privacy protection requirement	Do not support prediction mechanism for practical workflow applications
Choudhary et al. [99]	HSGA	2017	✓	✓	✓	✘	✘	✘	✘	✘	✘	MO	NS	Reduces makespan and cost with fixed bandwidth of VMs	Does not work for variable bandwidth between VMs

#Citations are considered up to June, 2018
NS not specified

enhance DLS the "analogous incremental cost" is calculated when the "dynamic level of task-machine pair" is evaluated. This procedure helps in figuring out the failure probability of application. With DLS, SGA is also applied to generate better results.

- *LDD-LS and CDCGA* Visheratin et al. [93] have planned a new hybrid algorithm for scheduling the workflow under user-specified QoS constraints. The main aim of LDD-LS is to make lines for workflow and analyze the computation time for each line. After which, the total deadline is allocated across levels of the workflow. However, CDCGA works on two different "heterogeneous populations". The sorted list of task identifiers is presented in one population in parallel with the computational resource indexes, whereas the sorted list of computational powers of resources is mentioned in the other population, concluding that the implementation of CDCGA is more effective than LDD-LS.
- *HSGA* Delavar and Aryan [94] have projected an algorithm for the minimization of run-time of workflow tasks. HSGA generates initial population by the virtual list of resources and their updated properties. To create initial population, all the tasks are sorted by utilizing priority method on the basis of graph topology and then resources will be allocated by merging the characteristics of "Best-Fit" and "Round-Robin" algorithms. They have explained the "Best-Fit and Round-Robin" algorithm and have proved that HSGA provide results that are more efficient than other presented algorithms of this paper.
- *DWSGA* Aryan and Delavar [95] have planned an algorithm that includes "Bi-Directional task prioritization" and "optimization of solutions". The main concern of DWSGA is to trim down the "GA operation iterations" by building up an optimized initial population in order to facilitate suitable workload distribution on resources. It also uses a particular mutation process that is very supportive in getting effective solutions.
- *RHDPSO* Tao et al. [21] have presented a hybrid approach that is useful in avoiding "premature convergence and local optimum". They have recommended two methods: one is "discretization method" in which multi-QoS constraint workflow scheduling issue is resolved and the other is "random time sequence method" to disturb the double extremums of particles, as a result of which "premature convergence and local optimum" issue is resolved. This hybrid method outperforms the standard PSO algorithm.

5 Discussions

This section targets the principal findings of the systematic literature review. It begins with the discussion of key sub-areas followed by the historical distribution of conspicuous researches. Furthermore, the practical impacts of study are analyzed as well as the multi-disciplinary scope is conferred followed by the validity of present study.

5.1 Key Sub-areas

While writing this paper, many other survey papers such as [100–102] from different fields of study as well as different journals have been analyzed thoroughly to accustom the authors' art of showing up their works, which eventually emanates this literature work.
The literature is categorized into following four distinct sub-areas attaining prominence from various research perspectives:

i.) QoS constraints for WFS
ii.) Heuristic WFS Schemes
iii.) Meta-heuristic WFS Schemes
iv.) Hybrid WFS Schemes

Various QoS metrics to be specified by the user while workflow scheduling are presented in Sect. 2.1. However, the limelight of this survey is the classification of various WFS schemes into Heuristic, Meta-heuristic and hybrid algorithms which are described in Sect. 4 and are comprehensively summarized in tables 3, 4 and 5, respectively. These schemes also highlight the open research challenges in respective spheres of research. In order to limit the survey, the issues related to load balancing, security, failure prediction and fault tolerance during WFS are not considered by the authors. Moreover, the review is constrained only to single-objective and multi-objective optimization techniques for the scheduling of workflows.

5.2 QoS and Workflow Benchmarks

The literature has acknowledged that different workflow scheduling algorithms have been developed to address the challenges of different QoS constraints. However, there are no workflow and QoS standards as per the author's review, which can evaluate and compare the QoS constrained workflow scheduling algorithms. Therefore, setting up some benchmarks has become the need of the hour with the proliferation of workflow techniques in cloud computing [103].

5.3 Strengths of Literature Review

The importance of this literature review is that it provides a reasonable amount of information about many approaches used for workflow scheduling with their advantages and disadvantages shown in Tables of Sect. 4. Tables 4, 5 and 6, being the limelight of this paper, will help the researchers to

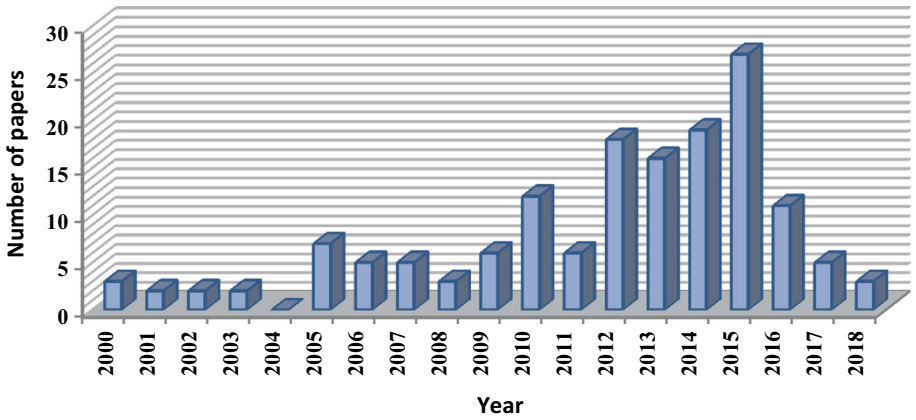

Fig. 6 Year-wise distribution of papers considered for literature review

know about various workflow schemes and scheduling strategies. Moreover, we have shown that how much work is done in all the different schemes by considering the range of year from 2000 to 2018.

5.4 Limitations of Literature Review

After analyzing the data collected from this literature review related to workflow scheduling, we have observed that there are following limitations of this review:

- We have not defined on what parameters, which approach performs better while considering different databases.
- We have not covered all the QoS constraints such as security, load balancing, etc.
- We have not defined the accuracy of any algorithm.

5.5 Historical Distribution

This subsection provides the result about the growth of specific studies done in the past many years. The relevant publications throughout the period between 2000 and 2018 are considered after performing QA, SSS and DEO, which are able to answer different RQs from Sect. 1 for answering the gaps in existing approaches (RQ1) to Sect. 6 for stating the future perception of the research (RQ5). The year-wise distribution of these papers is also shown in Fig. 6, and it is apparent that the proportion of research papers concerned with WFS is maximum in the year 2015. Pondering over, it has been acknowledged that more than half of the total existing researchers have preferred heuristic schemes for the scheduling of workflows in the context of cloud, whereas about one-fourth of the research sample involves the meta-heuristic schemes as depicted in Fig. 7.

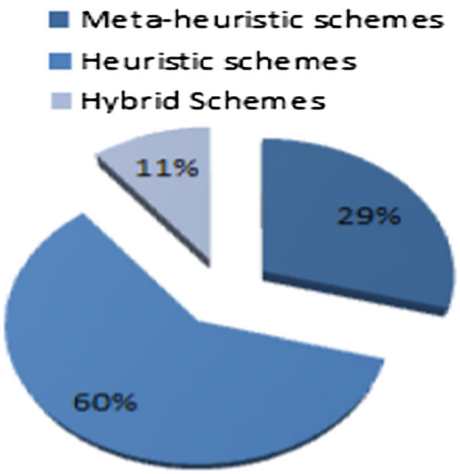

Fig. 7 Types and percentage of different scheduling schemes in existing literature

5.6 Analysis of Practical Impact

Though analyzing the effectiveness of existing researches has been one of the challenging tasks during this survey, yet the authors have made an attempt to scrutinize their practical impact by taking citations into account. To limit the data, Table 7 shows only top 10 cited researches.

5.7 Multi-disciplinary Applications

In the literature, it has been found that scope of the existing WFS schemes is not limited to cloud computing only, but software engineering [104], telecommunications [105,106], bioinformatics [107], economics and finance [108], scheduling [109], and cutting and packing [110] as well. Moreover, QoS constraints discussed in this paper have shown their relevance in different realms such as computer networking, optimization, etc. Clearly, the authors can precisely state the multi-disciplinary nature of QoS-based WFS algorithms mentioned in this article.

Table 7 Top 10 cited researches in the literature

Author(s)	Year	Total citations	Rationale
Pandey et al. [62]	2010	**666**	PSO is used to schedule applications taking into account both computation cost and data transmission cost. Authors have used it to establish new hybrid techniques by making its variants while considering QoS objectives
Yu et al. [19]	2005	**462**	Researchers have used it mostly for optimization of deadline and budget constraints for both static and dynamic scheduling in distributed environment
Chen [87]	2009	**327**	Researchers have focused on wireless sensor networks, scheduling of scientific workflows, text categorization and researchers those who want to launch new hybrid approach
Xu et al. [45]	2009	**230**	Resource provisioning and scheduling of workflows while considering Deadline constraint, fault tolerance in mobile social cloud computing, clustering based scheduling, load balancing in cloud, etc
Yu et al. [86]	2006	**214**	Scheduling the workflow under cost and time constraints and those who have focused on fault tolerance and to optimize multiple objectives in cloud computing
Rodriguez et al. [111]	2013	**284**	This paper is Followed by those who have focused on multi-objective optimization in scheduling, data retrieval for cloud robotic systems, fault tolerant scheduling and big data applications
Bittencourt et al. [33]	2011	**213**	Researchers who have focused on cost and budget constraints in workflow scheduling, clustering in scheduling and multi-objective scheduling in big data processing
Etminani et al. [49]	2007	**187**	Researchers those who are interested in heuristic techniques of scheduling and to create new variants of heuristics have mainly focused on this paper
Xu et al. [85]	2003	**156**	Researchers who want to do scheduling in grid computing for multiple QoS objectives and who have focused on optimization and wireless networking
Abrisha-mi et al. [35]	2012	**185**	Researchers have focused on cost, time, energy efficiency, budget, deadlines and data-intensive workflows in this paper

Number of Citations are obtained from Google Scholar and as of date 04-06-2018

5.8 Research Validity

The authors have attempted to survey the existing literature cautiously. However, some primary studies have remained untouched because of the devised Search Strategy since different researchers use different synonyms related to their studies. Additionally, in order to avoid the biasness of study selection problem, the authors have reviewed the techniques thoroughly during DEO activity as well.

6 Conclusions and Future Work

Cloud computing has drawn voluminous attention from the researchers' community and the industry from the first light where one of the prime issues aspired by the researchers so far to be delved into is Workflow Scheduling (WFS). In this regard, the authors have attempted to present a systematic review of the existing literature related to WFS in Cloud Computing with an intention to spot the distinct trends of WFS and investigated the various aspects ranging from workflow types, QoS constraints, WFS schemes, as well as their categorization into heuristic, meta-heuristic and hybrid schemes, their practical impacts and their multi-disciplinary applications. Moreover, through in-depth analysis and interpretation of the collected data, they have obtained splendid findings along with the pros and cons of these techniques. Another point worth considering here is that it has been witnessed from other existing surveys of WFS that no one has considered all the key aspects with such meticulous information of different scheduling schemes, numerous QoS constraints, distinct scheduling strategies as well as types of optimizations, and due to this, authors have decided to present a survey in which anyone can get up-to-date knowledge of issues related to WFS. It has also been found that most of the researchers have preferred WorkflowSim over CloudSim simulator for scheduling workflow applications in cloud since WorkflowSim allows the modelling and simulation of the cloud environment, VMs, data centres, and cloudlets by taking into account only a single work load.

In previous researches, the makespan and cost are the only two objectives which have been focused upon by the researchers so far (as per author's survey). Therefore, the following are the promising future directions:

- Work can be extended by considering objectives other than makespan and cost.
- The fault tolerance, load balancing and security of workflows as well as cloud resources can be incorporated.
- Pondering over, the catastrophe consequences of workflow failures on different cloud resources can be trimmed down by envisaging the failures perceptively with the aid of machine-learning approaches.
- Last, but not the least, the work can also be extended by implementing up-to-the-minute optimization algorithms for developing optimal schedules for the purpose of WFS.

References

1. Abrishami, S.; Naghibzadeh, M.; Epema, D.H.: Deadline-constrained workflow scheduling algorithms for Infrastructure as a service clouds. Future Gener. Comput. Syst. **29**(1), 158–169 (2013)
2. Masdari, M.; ValiKardan, S.; Shahi, Z.; Azar, S.I.: Towards workflow scheduling in cloud computing: a comprehensive analysis. J. Netw. Comput. Appl. **66**, 64–82 (2016)
3. Arabnejad, H.; Barbosa, J.G.: A budget constrained scheduling algorithm for workflow applications. J. Grid Comput. **12**(4), 665–679 (2014)
4. Wu, F.; Wu, Q.; Tan, Y.: Workflow scheduling in cloud: a survey. J. Supercomput. **71**(9), 3373–3418 (2015)
5. Arya, L.K.; Verma, A.: Workflow scheduling algorithms in cloud environment: a survey. In: 2014 Recent Advances in Engineering and Computational Sciences (RAECS), pp. 1–4. IEEE (2014)
6. Bala A.; Chana I. : Design and deployment of workflows in cloud environment. Int. J. Comput. Appl. 51(11), 9–15 (2012). https://doi.org/10.5120/8084-1536
7. Cao, H.; Jin, H.; Wu, X.; Wu, S.; Shi, X.: DAGMap: efficient and dependable scheduling of DAG workflow job in Grid. J. Supercomput. **51**(2), 201–223 (2010)
8. Deelman, E.; Singh, G.; Su, M.H.; Blythe, J.; Gil, Y.; Kesselman, C.; Laity, A.: Pegasus: a framework for mapping complex scientific workflows onto distributed systems. Sci. Program. **13**(3), 219–237 (2005)
9. Verma, A.; Kaushal, S.: Cost-time efficient scheduling plan for executing workflows in the cloud. J. Grid Comput. **13**(4), 495–506 (2015)
10. Bharathi, S.; Chervenak, A.; Deelman, E.; Mehta, G.; Su, M.H.; Vahi, K.: Characterization of scientific workflows. In: Third Workshop on Workflows in Support of Large-Scale Science, 2008. WORKS 2008, pp. 1–10. IEEE (2008)
11. Chawla, Y.; Bhonsle, M.: A study on scheduling methods in cloud computing. Int. J. Emerg. Trends Technol. Comput. Sci. (IJETTCS) **1**(3), 12–17 (2012)
12. Xu, Y.; Li, K.; He, L.; Zhang, L.; Li, K.: A hybrid chemical reaction optimization scheme for task scheduling on heterogeneous computing systems. IEEE Trans. Parallel Distrib. Syst. **26**(12), 3208–3222 (2015)
13. Vijayalakshmi, R.; Vasudevan, V.: Static batch mode heuristic algorithm for mapping independent tasks in computational grid. J. Comput. Sci. **11**(1), 224–229 (2015)
14. Nesmachnow, S.; Cancela, H.; Alba, E.: A parallel micro evolutionary algorithm for heterogeneous computing and grid scheduling. Appl. Soft Comput. **12**(2), 626–639 (2012)
15. Kaleeswaran, A.; Ramasamy, V.; Vivekanandan, P.: Dynamic Scheduling of Data Using Genetic Algorithm in Cloud Computing. Park College of Engineering and Technology, Coimbatore (1963)
16. Patel, S.; Bhoi, U.: Priority based job scheduling techniques in cloud computing: a systematic review. Int. J. Sci. Technol. Res. **2**(11), 147–152 (2013)
17. Casavant, T.L.; Kuhl, J.G.: A taxonomy of scheduling in general-purpose distributed computing systems. IEEE Trans. Softw. Eng. **14**(2), 141–154 (1988)
18. Blythe, J.; Jain, S.; Deelman, E.; Gil, Y.; Vahi, K.; Mandal, A.; Kennedy, K.: Task scheduling strategies for workflow-based applications in grids. In: IEEE International Symposium on Cluster Computing and the Grid, 2005. CCGrid 2005, vol. 2, pp. 759–767. IEEE (2005)
19. Yu, J.; Buyya, R.; Tham, C.K.: Cost-based scheduling of scientific workflow applications on utility grids. In: First International Conference on e-Science and Grid Computing, 2005. IEEE (2005)
20. Zhu, M.; Wu, Q.; Zhao, Y.: A cost-effective scheduling algorithm for scientific workflows in clouds. In: Performance Computing and Communications Conference (IPCCC), 2012 IEEE 31st International, pp. 256–265. IEEE (2012)
21. Tao, Q.; Chang, H.; Yi, Y.; Gu, C.; Yu, Y.: QoS constrained grid workflow scheduling optimization based on a novel PSO algorithm. In: Eighth International Conference on Grid and Cooperative Computing, 2009. GCC'09, pp. 153–159. IEEE (2009)
22. Tsai, Y.L.; Liu, H.C.; Huang, K.C.: Adaptive dual-criteria task group allocation for clustering-based multi-workflow scheduling on parallel computing platform. J. Supercomput. **71**(10), 3811–3831 (2015)
23. Yu, Z.; Shi, W.: A planner-guided scheduling strategy for multiple workflow applications. In: International Conference on Parallel Processing-Workshops, 2008. ICPP-W'08, pp. 1–8. IEEE (2008)
24. Zheng, W.; Xu, C.; Bao, W.: Online scheduling of multiple deadline-constrained workflow applications in distributed systems. In: 2015 Third International Conference on Advanced Cloud and Big Data, pp. 104–111. IEEE (2015)
25. Rimal, B.P.; Maier, M.: Workflow scheduling in multi-tenant cloud computing environments. IEEE Trans. Parallel Distrib. Syst. **28**(1), 290–304 (2017)
26. Tao, Y.; Jin, H.; Wu, S.; Shi, X.; Shi, L.: Dependable grid workflow scheduling based on resource availability. J. Grid Comput. **11**(1), 47–61 (2013)
27. Richard, P.; Cottet, F.; Richard, M.: On-line scheduling of real-time distributed computers with complex communication constraints. In: Seventh IEEE International Conference on Engineering of Complex Computer Systems, 2001. Proceedings, pp. 26–34. IEEE (2001)
28. Shi, J.; Luo, J.; Dong, F.; Zhang, J.: A budget and deadline aware scientific workflow resource provisioning and scheduling mechanism for cloud. In: Proceedings of the 2014 IEEE 18th Inter-

national Conference on Computer Supported Cooperative Work in Design (CSCWD), pp. 672–677. IEEE (2014)
29. Shi, J.; Luo, J.; Dong, F.; Zhang, J.; Zhang, J.: Elastic resource provisioning for scientific workflow scheduling in cloud under budget and deadline constraints. Clust. Comput. **19**(1), 167–182 (2016)
30. Yan, Y.; Chapman, B.: Scientific workflow scheduling in computational grids planning, reservation, and data/network-awareness. In: Proceedings of the 8th IEEE/ACM International Conference on Grid Computing, pp. 18–25. IEEE Computer Society (2007)
31. Yang, X.S.: Nature-Inspired Metaheuristic Algorithms. Luniver press, Bristol (2010)
32. Topcuoglu, H.; Hariri, S.; Wu, M.Y.: Performance-effective and low-complexity task scheduling for heterogeneous computing. IEEE Trans. Parallel Distrib. Syst. **13**(3), 260–274 (2002)
33. Bittencourt, L.F.; Madeira, E.R.M.: HCOC: a cost optimization algorithm for workflow scheduling in hybrid clouds. J. Internet Serv. Appl. **2**(3), 207–227 (2011)
34. Weng, C.; Lu, X.: Heuristic scheduling for bag-of-tasks applications in combination with QoS in the computational grid. Future Gener. Comput. Syst. **21**(2), 271–280 (2005)
35. Abrishami, S.; Naghibzadeh, M.; Epema, D.H.: Cost-driven scheduling of grid workflows using partial critical paths. IEEE Trans. Parallel Distrib. Syst. **23**(8), 1400–1414 (2012)
36. Lin, C.; Lu, S.: Scheduling scientific workflows elastically for cloud computing. In: 2011 IEEE International Conference on Cloud Computing (CLOUD), pp. 746–747. IEEE (2011)
37. Fard, H.M.; Prodan, R.; Barrionuevo, J.J.D.; Fahringer, T.: A multi-objective approach for workflow scheduling in heterogeneous environments. In: Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012), pp. 300–309. IEEE Computer Society (2012)
38. Bittencourt, L.F.; Madeira, E.R.: Towards the scheduling of multiple workflows on computational grids. J. Grid Comput. **8**(3), 419–441 (2010)
39. Lopez, M.M.; Heymann, E.; Senar, M.A.: Analysis of dynamic heuristics for workflow scheduling on grid systems. In: The Fifth International Symposium on Parallel and Distributed Computing, 2006. ISPDC'06, pp. 199–207. IEEE (2006)
40. Zhang, F.; Cao, J.; Hwang, K.; Wu, C.: Ordinal optimized scheduling of scientific workflows in elastic compute clouds. In: 2011 IEEE Third International Conference on Cloud Computing Technology and Science (CloudCom), pp. 9–17. IEEE (2011)
41. Jinquan, Z.; Lina, N.; Changjun, J.: A heuristic scheduling strategy for independent tasks on grid. In: Eighth International Conference on High-Performance Computing in Asia-Pacific Region, 2005. Proceedings. IEEE (2005)
42. Tsai, C.W.; Huang, W.C.; Chiang, M.H.; Chiang, M.C.; Yang, C.S.: A hyper-heuristic scheduling algorithm for cloud. IEEE Trans. Cloud Comput. **2**(2), 236–250 (2014)
43. Han, H.; Deyui, Q.; Zheng, W.; Bin, F.: A QoS guided task scheduling model in cloud computing environment. In: 2013 Fourth International Conference on Emerging Intelligent Data and Web Technologies (EIDWT), pp. 72–76. IEEE (2013)
44. Stavrinides, G.L.; Karatza, H.D.: A cost-effective and QoS-aware approach to scheduling real-time workflow applications in PaaS and SaaS clouds. In: 2015 3rd International Conference on Future Internet of Things and Cloud (FiCloud), pp. 231–239. IEEE (2015)
45. Xu, M.; Cui, L.; Wang, H.; Bi, Y.: A multiple QoS constrained scheduling strategy of multiple workflows for cloud computing. In: 2009 IEEE International Symposium on Parallel and Distributed Processing with Applications, pp. 629–634. IEEE (2009)
46. Verma, A.; Kaushal, S.: Deadline and budget distribution based cost-time optimization workflow scheduling algorithm for cloud. In: Proceedings of the IJCA on International Conference on Recent Advances and Future Trends in Information Technology (iRAFIT'12), pp. 1–4 (2012)
47. Dogan, A.; Ozguner, F.: On QoS-based scheduling of a meta-task with multiple QoS demands in heterogeneous computing. In: Parallel and Distributed Processing Symposium. Proceedings International, IPDPS 2002, Abstracts and CD-ROM. IEEE (2001)
48. Hamscher, V.; Schwiegelshohn, U.; Streit, A.; Yahyapour, R.: Evaluation of job-scheduling strategies for grid computing. In: International Workshop on Grid Computing, pp. 191–202. Springer, Berlin (2000)
49. Etminani, K.; Naghibzadeh, M.: A min–min max–min selective algorithm for grid task scheduling. In: 3rd IEEE/IFIP International Conference in Central Asia on Internet, 2007. ICI 2007, pp. 1–7. IEEE (2007)
50. Lee, Y.C.; Subrata, R.; Zomaya, A.Y.: On the performance of a dual-objective optimization model for workflow applications on grid platforms. IEEE Trans. Parallel Distrib. Syst. **20**(9), 1273–1284 (2009)
51. Bittencourt, L.F.; Senna, C.R.; Madeira, E.R.: Scheduling service workflows for cost optimization in hybrid clouds. In: 2010 International Conference on Network and Service Management (CNSM), pp. 394–397. IEEE (2010)
52. Abrishami, S.; Naghibzadeh, M.: Deadline-constrained workflow scheduling in software as a service cloud. Sci. Iran. **19**(3), 680–689 (2012)
53. Amalarethinam, D.G.; Selvi, F.K.M.: A minimum makespan grid workflow scheduling algorithm. In: 2012 International Conference on Computer Communication and Informatics (ICCCI), pp. 1–6. IEEE (2012)
54. Fard, H.M.; Prodan, R.; Fahringer, T.: A truthful dynamic workflow scheduling mechanism for commercial multicloud environments. IEEE Trans. Parallel Distrib. Syst. **24**(6), 1203–1212 (2013)
55. Pawar, C.S.; Wagh, R.B.: Priority based dynamic resource allocation in cloud computing. In: 2012 International Symposium on Cloud and Services Computing (ISCOS), pp. 1–6. IEEE (2012)
56. Zhou, A.C.; He, B.: Transformation-based monetary costoptimizations for workflows in the cloud. IEEE Trans. Cloud Comput. **2**(1), 85–98 (2014)
57. Zeng, L.; Veeravalli, B.; Li, X.: SABA: a security-aware and budget-aware workflow scheduling strategy in clouds. J. Parallel Distrib. Comput. **75**, 141–151 (2015)
58. Tang, Z.; Jiang, L.; Zhou, J.; Li, K.; Li, K.: A self-adaptive scheduling algorithm for reduce start time. Future Gener. Comput. Syst. **43**, 51–60 (2015)
59. Lee, Y.C.; Han, H.; Zomaya, A.Y.; Yousif, M.: Resource-efficient workflow scheduling in clouds. Knowl. Based Syst. **80**, 153–162 (2015)
60. Lin, B.; Guo, W.; Xiong, N.; Chen, G.; Vasilakos, A.V.; Zhang, H.: A pretreatment workflow scheduling approach for big data applications in multicloud environments. IEEE Trans. Netw. Serv. Manag. **13**(3), 581–594 (2016)
61. Zhu, J.; Li, X.; Ruiz, R.; Xu, X.: Scheduling stochastic multi-stage jobs to elastic hybrid cloud resources. IEEE Trans. Parallel Distrib. Syst. **29**(6), 1401–1415 (2018)
62. Pandey, S.; Wu, L.; Guru, S.M.; Buyya, R.: A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. In: 2010 24th IEEE International Conference on Advanced Information Networking and Applications (AINA), pp. 400–407. IEEE (2010)
63. Wu, Z.; Ni, Z.; Gu, L.; Liu, X.: A revised discrete particle swarm optimization for cloud workflow scheduling. In: 2010 International Conference on Computational Intelligence and Security (CIS), pp. 184–188. IEEE (2010)
64. Thanh, T.P.; The, L.N.; Doan, C.N.: A novel workflow scheduling algorithm in cloud environment. In: 2015 2nd National Foun-

dation for Science and Technology Development Conference on Information and Computer Science (NICS), pp. 125–129. IEEE (2015)
65. Liu, H.; Abraham, A.; Hassanien, A.E.: Scheduling jobs on computational grids using a fuzzy particle swarm optimization algorithm. Future Gener. Comput. Syst. **26**(8), 1336–1343 (2010)
66. Yu, J.; Kirley, M.; Buyya, R.: Multi-objective planning for workflow execution on grids. In: 2007 8th IEEE/ACM International Conference on Grid Computing, pp. 10–17. IEEE (2007)
67. Singh, G.; Kesselman, C.; Deelman, E.: A provisioning model and its comparison with best-effort for performance-cost optimization in grids. In: Proceedings of the 16th International Symposium on High Performance Distributed Computing, pp. 117–126. ACM (2007)
68. Wang, X.; Yeo, C.S.; Buyya, R.; Su, J.: Optimizing the makespan and reliability for workflow applications with reputation and a look-ahead genetic algorithm. Future Gener. Comput. Syst. **27**(8), 1124–1134 (2011)
69. Gharooni-fard, G.; Moein-darbari, F.; Deldari, H.; Morvaridi, A.: Scheduling of scientific workflows using a chaos-genetic algorithm. Procedia Comput. Sci. **1**(1), 1445–1454 (2010)
70. Zuo, X.; Zhang, G.; Tan, W.: Self-adaptive learning PSO-based deadline constrained task scheduling for hybrid IaaS cloud. IEEE Trans. Autom. Sci. Eng. **11**(2), 564–573 (2014)
71. Liu, H.; Xu, D.; Miao, H.K.: Ant colony optimization based service flow scheduling with various QoS requirements in cloud computing. In: 2011 First ACIS International Symposium on Software and Network Engineering (SSNE), pp. 53–58. IEEE (2011)
72. Kumar, P.; Verma, A.: Scheduling using improved genetic algorithm in cloud computing for independent tasks. In: Proceedings of the International Conference on Advances in Computing, Communications and Informatics, pp. 137–142. ACM (2012)
73. Javanmardi, S.; Shojafar, M.; Amendola, D.; Cordeschi, N.; Liu, H.; Abraham, A.: Hybrid job scheduling algorithm for cloud computing environment. In: Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014, pp. 43–52. Springer (2014)
74. Bilgaiyan, S.; Sagnika, S.; Das, M.: Workflow scheduling in cloud computing environment using cat swarm optimization. In: Advance Computing Conference (IACC), 2014 IEEE International, pp. 680–685. IEEE (2014)
75. Singh, L.; Singh, S.: A genetic algorithm for scheduling workflow applications in unreliable cloud environment. In: International Conference on Security in Computer Networks and Distributed Systems, pp. 139–150. Springer, Berlin (2014)
76. Netjinda, N.; Sirinaovakul, B.; Achalakul, T.: Cost optimal scheduling in IaaS for dependent workload with particle swarm optimization. J. Supercomput. **68**(3), 1579–1603 (2014)
77. Chen, W.N.; Zhang, J.: A set-based discrete PSO for cloud workflow scheduling with user-defined QoS constraints. In: 2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 773–778. IEEE (2012)
78. Liang, Y.C.; Chen, A.H.L.; Nien, Y.H.: Artificial Bee Colony for workflow scheduling. In: 2014 IEEE Congress on Evolutionary Computation (CEC), pp. 558–564. IEEE (2014)
79. Kaur, N.; Singh, S.: A budget-constrained time and reliability optimization BAT algorithm for scheduling workflow applications in clouds. Procedia Comput. Sci. **98**, 199–204 (2016)
80. George, S.: Truthful workflow scheduling in cloud computing using hybrid PSO-ACO. In: 2015 International Conference on Developments of E-Systems Engineering (DeSE), pp. 60–64. IEEE (2015)
81. Sridhar, M.; Babu, G.R.M.: Hybrid particle swarm optimization scheduling for cloud computing. In: Advance Computing Conference (IACC), 2015 IEEE International, pp. 1196–1200. IEEE (2015)
82. Liu, C.Y.; Zou, C.M.; Wu, P.: A task scheduling algorithm based on genetic algorithm and ant colony optimization in cloud computing. In: 2014 13th International Symposium on Distributed Computing and Applications to Business, Engineering and Science (DCABES), pp. 68–72. IEEE (2014)
83. Sathish, K.; Reddy, A.R.: Workflow scheduling in grid computing environment using a hybrid GAACO approach. J. Inst. Eng. (India) Ser. B **98**(1), 121–128 (2017)
84. Loukopoulos, T.; Lampsas, P.; Sigalas, P.: Improved genetic algorithms and list scheduling techniques for independent task scheduling in distributed systems. In: Eighth International Conference on Parallel and Distributed Computing, Applications and Technologies, 2007. PDCAT'07, pp. 67–74. IEEE (2007)
85. Xu, Z.; Hou, X.; Sun, J.: Ant algorithm-based task scheduling in grid computing. In: Canadian Conference on Electrical and Computer Engineering, 2003. IEEE CCECE 2003, vol. 2, pp. 1107–1110. IEEE (2003)
86. Yu, J.; Buyya, R.: A budget constrained scheduling of workflow applications on utility grids using genetic algorithms. In: Workflows in Support of Large-Scale Science, 2006. WORKS'06. Workshop on, pp. 1–10. IEEE (2006)
87. Chen, W.N.; Zhang, J.: An ant colony optimization approach to a grid workflow scheduling problem with various QoS requirements. IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.) **39**(1), 29–43 (2009)
88. Verma, A.; Kaushal, S.: Bi-criteria priority based particle swarm optimization workflow scheduling algorithm for cloud. In: 2014 Recent Advances in Engineering and Computational Sciences (RAECS), pp. 1–6. IEEE (2014)
89. Chen, Z.G.; Du, K.J.; Zhan, Z.H.; Zhang, J.: Deadline constrained cloud computing resources scheduling for cost optimization based on dynamic objective genetic algorithm. In: 2015 IEEE Congress on Evolutionary Computation (CEC), pp. 708–714. IEEE (2015)
90. Lal, A.; Krishna, C.R.: Critical path-based ant colony optimization for scientific workflow scheduling in cloud computing under deadline constraint. In: Ambient Communications and Computer Systems, pp. 447–461. Springer, Singapore (2018)
91. Kardani-Moghaddam, S.; Khodadadi, F.; Entezari-Maleki, R.; Movaghar, A.: A hybrid genetic algorithm and variable neighborhood search for task scheduling problem in grid environment. Procedia Eng. **29**, 3808–3814 (2012)
92. Doğan, A.; Özgüner, F.: Biobjective scheduling algorithms for execution time-reliability trade-off in heterogeneous computing systems. Comput. J. **48**(3), 300–314 (2005)
93. Visheratin, A.A.; Melnik, M.; Nasonov, D.: Workflow scheduling algorithms for hard-deadline constrained cloud environments. Procedia Comput. Sci. **80**, 2098–2106 (2016)
94. Delavar, A.G.; Aryan, Y.: HSGA: a hybrid heuristic algorithm for workflow scheduling in cloud systems. Clust. Comput. **17**(1), 129–137 (2014)
95. Aryan, Y.; Delavar, A.G.: A bi-objective workflow application scheduling in cloud computing systems. Int. J. Integr. Technol. Educ. **3**(2), 51–62 (2014)
96. Rahman, M.; Hassan, R.; Ranjan, R.; Buyya, R.: Adaptive workflow scheduling for dynamic grid and cloud computing environment. Concurr. Comput. Pract. Exp. **25**(13), 1816–1842 (2013)
97. Tao, Y.; Jin, H.; Wu, S.; Shi, X.; Shi, L.: Dependable grid workflow scheduling based on resource availability. J. Grid Comput. **11**(1), 47–61 (2014)
98. Chen, C.; Liu, J.; Wen, Y.; Chen, J.; Zhou, D.: A hybrid genetic algorithm for privacy and cost aware scheduling of data intensive workflow in cloud. In: International Conference on Algorithms and Architectures for Parallel Processing, pp. 578–591. Springer (2015)

99. Choudhary, A.; Gupta, I.; Singh, V.; Jana, P.K.: A GSA based hybrid algorithm for bi-objective workflow scheduling in cloud computing. Future Gener. Comput. Syst. **83**, 14–26 (2018)
100. Muram, F.U.; Tran, H.; Zdun, U.: Systematic review of software behavioral model consistency checking. ACM Comput. Surv. **50**(4), 17:1–17:39 (2017)
101. Bagga, P.; Hans, R.: Mobile agents system security: a systematic survey. ACM Comput. Surv. **50**(5), 65:1–65:45 (2017)
102. Jatoth, C.; Gangadharan, G.R.; Buyya, R.: Computational intelligence based QoS-aware web service composition: a systematic literature review. IEEE Trans. Serv. Comput. **10**(3), 475–492 (2017)
103. Yu, J.: Qos-based scheduling of workflows on global grids. Doctoral dissertation (2007)
104. Díaz, E.; Tuya, J.; Blanco, R.: Automated software testing using a metaheuristic technique based on Tabu search. In: 18th IEEE International Conference on Automated Software Engineering, 2003. Proceedings, pp. 310–313. IEEE (2003)
105. Alba, E.; Chicano, J.F.: Evolutionary algorithms in telecommunications. In: Electrotechnical Conference, 2006. MELECON 2006. IEEE Mediterranean, pp. 795–798. IEEE (2006)
106. He, L.; Mort, N.: Hybrid genetic algorithms for telecommunications network back-up routeing. BT Technol. J. **18**(4), 42–50 (2000)
107. Armañanzas, R.; Inza, I.; Santana, R.; Saeys, Y.; Flores, J.L.; Lozano, J.A.; Larrañaga, P.: A review of estimation of distribution algorithms in bioinformatics. BioData Min. **1**(1), 6 (2008)
108. Ponsich, A.; Jaimes, A.L.; Coello, C.A.C.: A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications. IEEE Trans. Evol. Comput. **17**(3), 321–344 (2013)
109. Zobolas, G.I.; Tarantilis, C.D.; Ioannou, G.: Minimizing makespan in permutation flow shop scheduling problems using a hybrid metaheuristic algorithm. Comput. Oper. Res. **36**(4), 1249–1267 (2009)
110. Bortfeldt, A.: A genetic algorithm for the two-dimensional strip packing problem with rectangular pieces. Eur. J. Oper. Res. **172**(3), 814–837 (2006)
111. Rodriguez, M.A.; Buyya, R.: Deadline based resource provisioningand scheduling algorithm for scientific workflows on clouds. IEEE Trans. Cloud Comput. **2**(2), 222–235 (2014)