Collaborative Workflow Scheduling over MANET, a User Position Prediction-Based Approach

Qinglan Peng[1], Qiang He[2], Yunni Xia[1(✉)], Chunrong Wu[1], and Shu Wang[3]

[1] Software Theory and Technology Chongqing Key Lab, Chongqing University, Chongqing, China
xiayunni@hotmail.com

[2] School of Software and Electrical Engineering, Swiburne University of Technology, Melbourne, Australia
qhe@swin.edu.au

[3] School of information, Liaoning University, Shenyang, China
swang@lnu.edu.cn

Abstract. The explosive increase of mobile devices and advanced communication technologies prompt the emergence of mobile computing. In this paradigm, mobile users' idle resources can be shared as service through device-to-device links to other users. Some complex workflow-based mobile applications are therefor no longer need to be offloaded to remote cloud, on the contrary, they can be solved locally with the help of other devices in a collaborative way. Nevertheless, various challenges, especially the reliability and quality-of-service of such a collaborative workflow scheduling problem, are yet to be properly tackled. Most studies and related scheduling strategies assume that mobile users are fully stable and with constantly available. However, this is not realistic in most real-world scenarios where mobile users are mobile most of time. The mobility of mobile users impact the reliability of corresponding shared resources and consequently impact the success rate of workflows. In this paper, we propose a reliability-aware mobile workflow scheduling approach based on prediction of mobile users' positions. We model the scheduling problem as a multi-objective optimization problem and develop an evolutionary multi-objective optimization based algorithm to solve it. Extensive case studies are performed based on a real-world mobile users' trajectory dataset and show that our proposed approach significantly outperforms traditional approaches in term of workflow success rate.

Keywords: Workflow scheduling · Mobile computing · Quality-of-service · Reliability

1 Introduction

Recent years have witnessed a rapid growth and advances of mobile devices, e.g., smart phones, tablet computers, wearable devices, etc., and mobile services.

H. Gao et al. (Eds.): CollaborateCom 2018, LNICST 268, pp. 33–52, 2019.
https://doi.org/10.1007/978-3-030-12981-1_3

Mobile devices are changing the way people access information in their daily lives. In the mobile computing environment, mobile users can exploit nearby resources, e.g., computing resource, data traffic, sensors, etc., through utilizing mobile services shared in a mobile ad hoc network (MANET). MANET is a self-organized local mobile network built by nodes within each other's communication fields.

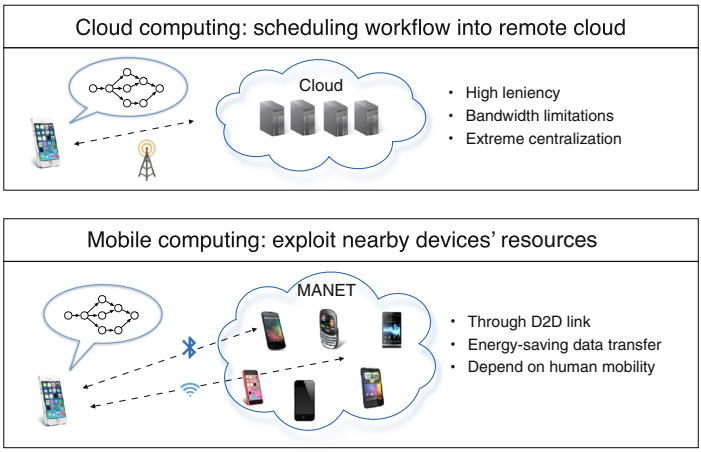


Fig. 1. Mobile computing paradigm.

As illustrated in Fig. 1, the core idea of mobile computing over MANET is sharing. In this paradigm, mobile users are allowed to utilize resources and services shared by other users nearby in a collaborative way, and thus the provisioning capability of involved services is expanded through exploiting direct physical contacts among users. These available resources and services can be shared directly among users in an elastic and on-demand way without time-consuming and energy-requiring communications with pre-existing infrastructure, for example, cellular networks and traditional centralized cloud data centers. Note that, workflow-based mobile applications over MANET (e.g., TensorFlow Lite, Photo editing on mobile, and Online video sharing) usually require huge computational resources and data transfer. Therefore, nearby mobile resources are thus more adept, in terms of timeliness and energy-efficiency, at executing these workflow tasks than remote cloud with the help of device-to-device (D2D) communications such as Bluetooth, Wi-Fi, and NFC. D2D communications are featured by extensively-reduced inter-device delays and energy consumption than traditional cellular networks [3]. It is widely believed to have potential to improve Quality-of-Service (QoS) of mobile services over MANET by providing increased user throughput, reduced cellular traffic, and extended network coverage [5].

However, users in a MANET often have high mobility, which resulting in topological changes in the MANET over time. Under such circumstances, it has

become a great challenge how to compose and schedule reliable workflow tasks over a versatile MANET and fulfill users' quality-of-service (QoS) requirements in the meantime.

To cope with aforementioned challenges and concerns, in this study, we propose a predictive reliability-aware mobile workflow scheduling approach over MANET. We first present the concept of mobile resources sharing community and the corresponding mobile resources reliability evaluation method. Then, a gaussian mixture model for user position prediction is used to capture the moving trend of mobile users and make a prediction of resource providers' reliability. Finally, we develop a multi-objective optimization based composition algorithm named MDEWS, the predicted reliability values are feed into this algorithm to yield workflow schedules. The results of experiments conducted on a real-world user movement dataset show that our approach is capable of dynamically capturing the mobility of mobile users and achieving higher success rates of workflows than traditional approaches.

2 Related Work

Workflow scheduling aims to schedule tasks into proper time slice of computing resources at proper time. As a well-known NP-hard problem, extensive studies have devoted into this problem in the past decades. Typically, most workflow scheduling approaches can be classified into two categories in term of computing platform: one is traditional multiprocessor and grid system, another is IaaS cloud which has attracted great attention recent years. In this section, we first review workflow scheduling problems in grid system and IaaS cloud, then discuss the challenges and concerns of scheduling workflow in mobile computing environment.

Grid can be seen as a service-oriented paradigm for resource-intensive applications. In a grid, every resource can be represented as a service and these resources are delivered through a utility computing models based service provisioning. Many heuristics and meta-heuristics based algorithms have been proposed to schedule workflow applications in grid. For example, Maheswaran *et al.* [10] studied on-line and batch heuristics for workflows scheduling in heterogeneous distributed system, they proposed three heuristics strategies: Min-Min, Max-Min and Sufferage. Topcuoglu *et al.* [20] presented two algorithms named Heterogeneous Earliest-Finish-Time (HEFT) and Critical-Path-on-a-Processor (CPOP) for workflow scheduling over heterogeneous processors with bounded number. Zhao *et al.* [16] assign a weight to each node and edge in a workflow, then they use a HEFT algorithm to schedule the tasks of workflow onto heterogeneous machines with bounded number.

Cloud computing is becoming an increasingly popular platform for workflow-based applications such as scientific workflows. Recent years, workflow scheduling in cloud has attracted great attention and extensive efforts have been devoted to this field based on the features of IaaS cloud [17,21,22]. For instance, Mao *et al.* [11] proposed an auto-scaling workflow scheduling approach, they consider

not only user performance requirements but also budget minimization. Abrishami *et al.* [1] extended the Partial Critical Paths (PCP) algorithm for utility grid to IaaS cloud which named Cloud Partial Critical Paths (IC-PCP). The scheduling goal of IC-PCP is minimizing execution cost while meeting deadline constrain. Rodriguez and Buyya [15] presented a static, deadline-constrained, cost-aware workflow scheduling approach based on Particle Swarm Optimisation (PSO). IaaS cloud features such as elastic and unlimited resource provision and VM performance variation are considered in their system model. Li *et al.* [8] proposed a predictive, fluctuation-aware workflow scheduling approach. They consider fluctuant VM performance and use an ARMA (Auto-Regressive Moving Average) model to make a prediction about VMs' future performance to achieve lower SLA (Service-Level-Agreement) violation rate. Zhu *et al.* [24] proposed a evolutionary multi-objective workflow scheduling algorithm named EMS-C, they model the workflow scheduling problem as a multi-objective optimization problem which optimizes both makespan and cost, then a NSGA-II based meta-heuristic algorithm is developed to solve it.

It can be seen that most approaches and algorithms are designed for cloud computing environment or grid system. Cloud and grid share the same characteristic of relative stability, and they are usually deployed in permanent data centers or distributed systems. However, these scheduling methods cannot be applied to the mobile computing environment directly because they usually do not consider resource reliability or they just consider constant reliability. In mobile computing environment, the topological structure of an MANET can change at any time and the fluctuation of reliability could be very volatile, which makes them fail to find a reliable schedule. Therefore, to schedule reliable workflow over MANET, approaches which adapt to dynamic mobile computing environment are required. In this paper, we propose a reliability-aware workflow scheduling approach based on user position prediction. To capture the variation trend of reliability, the predicted mobile user positions are used to evaluate the reliability of resource provider, then these reliability values are fed into a multi-objective evolutionary algorithm to generate reliable-aware schedule plans.

3 Preliminaries

3.1 Mobile Resources Sharing Community

A Mobile Resources Sharing Community (MRSC) is a mobile ad hoc network for mobile resources sharing. It is usually constructed by nearby mobile devices and sink nodes. It can be formally described as a 2-tuple $M = (N, C)$, where N is the set of mobile devices and sink nodes in an MRSC, C the set of connections for every communication path. Figure 2 shows an example MRSC established in a coffee shop, where mobile users are within each other's D2D transmission ranges.

An MRSC has three characteristics: (1) locality: A mobile user can perceive available computing resources around and deploy workflow tasks to other mobile devices in the same MRSC, and locality of computing resources can thus

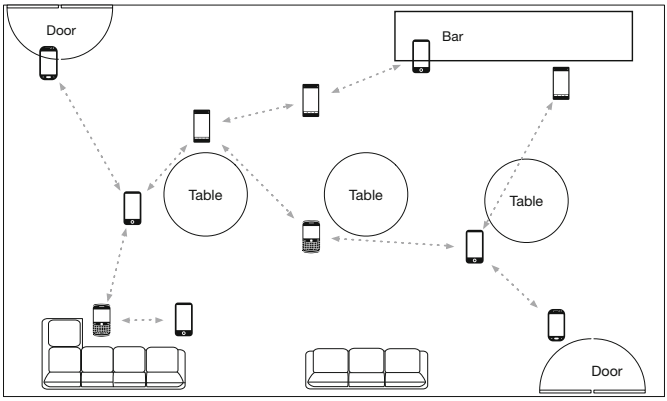


Fig. 2. An MRSC example in a café.

be exploited and utilized; (2) mobility: In an MRSC, it is not uncommon that all computing nodes are constantly moving during task executing time; (3) dynamicity: Mobile users may join or leave an MRSC automatically when they enter or leave a participating user's transmission range.

3.2 Mobile Resource Reliability

It can be seen that scheduling workflow tasks over an MANET is unreliable due to the high mobility of mobile devices. In this paper, the reliability of D2D links between mobile devices in MANET are considered when evaluating the reliability of mobile resource provider [7]. Suppose that there are a total of $|N|$ nodes and $|C|$ connections in an MRSC at time t, the reliability of computing resource provided by provider p for requester r can be calculated as the reliability between these two devices p and r. In an MRSC, each edge has its operational probability ρ, which can be estimated from the received signal strength indicator (RSSI) value or GPS data easily [13]. The state of the MRSC at time t can thus be represented as $S(t) = [S_1(t), S_2(t), ..., S_{|C|}(t)]$, where the i-th element $S_i(t)$ is assigned to 1 if the i-th edge is working at time t, otherwise 0. Thus, the probability of an MRSC being in a given state can be calculated as follows:

$$\mathbf{P}(S(t)) = \prod_{i=1}^{|C|} \rho_i^{S_i(t)} (1 - \rho_i)^{1 - S_i(t)} \tag{1}$$

then the reliability of a D2D link between p and r can be expressed as follows:

$$RL_{(s,d)}[G(t)] = \sum_{all\ S(t)} \phi(S(t), p, r) \mathbf{P}(S(t)) \tag{2}$$

where $\phi(S(t), p, r)$ is the function to identify whether there are available paths between device p and r. If at state $S(t)$, there is at least one path between p and r, then $\phi(S(t), p, r) = 1$, otherwise 0.

It can be seen that the reliability of a resource provider in an MANET varies over time and is closely related to the communication distance between the workflow application requester and the mobile resources providers. A mobile provider currently observed to be available may become unavailable in the near future due to the change in this distance.

3.3 Gaussian Mixture Model for User Position Prediction

A recent study [18] reports that there is a potential 93% average predictability in user mobility. For example, Fig. 3 shows pedestrians' trajectories on a campus. We can clearly see that most trajectories share similarity and regularity patterns. Such similarity, periodicity, and regularity can be formally and properly described with novel methods [2,9,14].

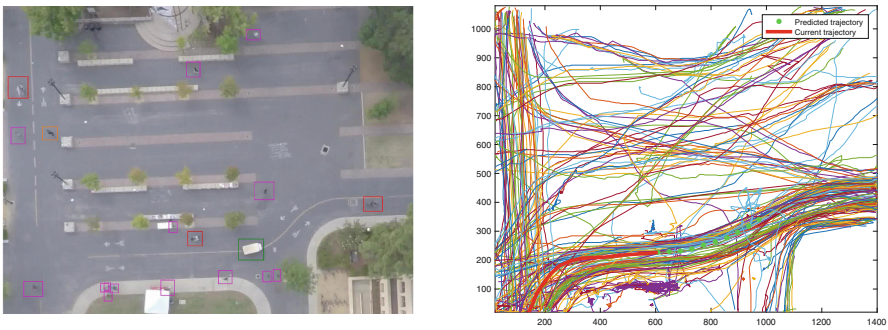


Fig. 3. Trajectories in a campus.

Human trajectories are usually fulfill multiple mobility patterns, depending on the subjective destination, the limit of objective environment, other people's movement and so on. Each pattern within a trajectory can be effectively described by a Gaussian process and the entire trajectories thus corresponds to a Gaussian Mixture Model (GMM).

In a GMM, users' history trajectory data can be described as follows:

$$\begin{aligned} U &= \{\Gamma_1, \Gamma_2, ..., \Gamma_n\} \\ &= \{(\overrightarrow{x_1}, \overrightarrow{y_1}), (\overrightarrow{x_2}, \overrightarrow{y_2}), ..., (\overrightarrow{x_n}, \overrightarrow{y_n})\} \\ &= \{\overrightarrow{X}, \overrightarrow{Y}\} \end{aligned} \tag{3}$$

where Γ_i denotes the i-th user's trajectories, $\overrightarrow{X}$ and $\overrightarrow{Y}$ the mapping vector of these trajectories in X and Y directions, respectively. A trajectory $\Gamma_i = (\overrightarrow{x_i}, \overrightarrow{y_i})$ can be expressed as a multiple different Gaussian processes as follows:

$$p(\overrightarrow{x_n}|\lambda) = \sum_{i=1}^{K} \omega_i GP(\overrightarrow{x_n}|\mu_{(x,i)}, \sigma_{(x,i)}) \tag{4}$$
$$p(\overrightarrow{y_n}|\lambda) = \sum_{i=1}^{K} \omega_i GP(\overrightarrow{y_n}|\mu_{(y,i)}, \sigma_{(y,i)})$$

where $GP(\overrightarrow{x_n}|\mu_{(x,i)}, \sigma_{(x,i)})$ denotes the probability function of trajectory Γ_n's X direction in the i-th trajectory pattern, K the number of all trajectory patterns, ω_i the weight of the i-th trajectory pattern with $\sum_{i=1}^{K} \omega_i = 1$, $\mu_{(x,i)}$ and $\mu_{(y,i)}$ the means of the i-th trajectory pattern in directions X and Y, $\sigma_{(x,i)}$ and $\sigma_{(y,i)}$ the covariances of the i-th trajectory pattern in directions X and Y, respectively. We use λ to denote the set of parameters $\{\omega_i, \mu_i, \sigma_i\}$ where $i \in \{1, 2, ..., N\}$. Therefore, the likelihood function of GMM for training set $U = \{\overrightarrow{X}, \overrightarrow{Y}\}$ can be expressed as follow:

$$P(\overrightarrow{X}|\lambda) = \prod_{n=1}^{K} p(\overrightarrow{x_n}|\lambda) \tag{5}$$
$$P(\overrightarrow{Y}|\lambda) = \prod_{n=1}^{K} p(\overrightarrow{y_n}|\lambda)$$

The forecasting process consists three steps: (1) applying a Gaussian Mixture clustering method [23] to trajectory dataset U to obtain K clusters, which correspond to K different trajectory patterns; (2) an expectation-maximization algorithm is applied to estimate parameter λ; (3) forecast a mobile user's future position based on his/her recent trajectory. The prediction process is employed in Sect. 4 to obtain the prediction results of mobile devices' position. Then the reliability of providers is evaluated based on its predicted position is further fed into the optimization formulation to facilitate workflow schedules.

3.4 Proposed Approach Architecture

In an MRSC, each mobile device represent a mobile resource provider which able to provide computing capability to nearby devices. The reliability of mobile resource providers is varying due to the high dynamic of MANET. In order to capture the reliability variations at run-time to realize reliable scheduling, we proposed a user position prediction-based workflow scheduling approach for MANET. As shown in Fig. 4, the process of our approach consists of three typical steps: (1) a workflow template is constructed when a mobile user wants to launch a workflow request. A workflow template usually has multiple tasks and each task can be scheduled to an available mobile resource provider. (2) then, it begins to discover potential providers in the same MRSC. At the same time, the reliability of these providers are evaluated according to its predicted positions. A resource pool which containing available providers is constructed in this step; (3) next, it will decide which provider to select for each task to realize the workflow

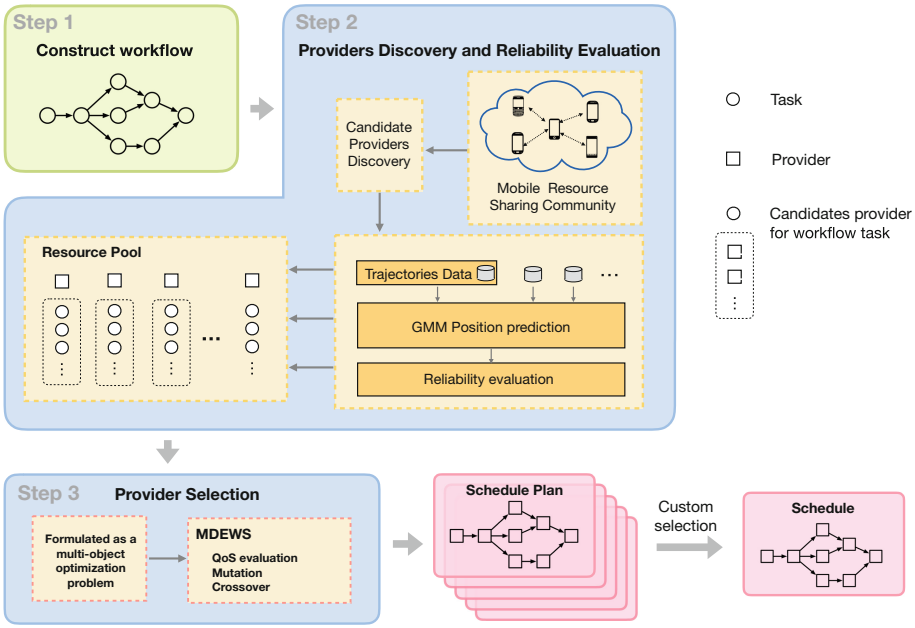


Fig. 4. The process of mobile workflow scheduling.

scheduling with satisfactory reliability and makespan. The decision making is transformed into a multi-objective optimization problem. Then an evolutionary-based algorithm named MDEWS is employed to yield a set of solutions, which are equally optimal from the view of Pareto fronts [4] and can be selected based on user preferences.

4 System Model and Problem Formulation

A workflow application is described by a Directed-Acyclic-Graph (DAG) $W = (T, E)$, where $T = (t_1, t_2, ..., t_m)$ denotes the set of tasks and E the set of edges. Without loss of generality, t_1 and t_m are considered to be the entry and exit tasks, respectively. The edge $e_{i,k}$ indicates that t_k can be executed after t_i is accomplished. ${}^*t_i$ and t_i^* denote the parent and child sets of t_i, respectively. The workflow starts and concludes by the entry and exit tasks, respectively. Figure 5 shows an example of sample workflow with 8 tasks. D denotes the user-recommended constraint of the completion time of the workflow, usually specified in SLA documents.

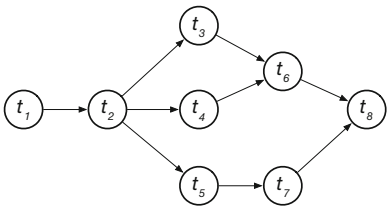


Fig. 5. An example of workflow.

An MRSC supports workflows application through mobile resource providers. These providers are selected from a resource provider pool, $P = \{p_1, p_2, ..., p_n\}$, and at most n providers are required at runtime if no two tasks share the same providers. Providers can be different in their CPU speed, memory, and pricing configurations, and each provider have an available period for tasks. The starting time of tasks is decided by their supporting provider and the completion time of their preceding tasks. If task t_i connects t_k through edge $e_{i,k}$ and they are executed by different providers, the transfer time, $D_{i,k}$, is inevitable because inter-device data and control signal transfer is required. Otherwise, $D_{i,k} = 0$ if both tasks are on the same provider.

Finally, the problem of workflow scheduling over MANET can therefore be formulated as:

$$\begin{aligned} Min: \quad & y = f(x) = (1 - \xi(x), \tau(x))^T \\ s.t: \quad & \tau(x) \leq D \\ & x = [x_1, x_2, ..., x_n]^T \in \Theta \\ & x_i^{min} \leq x_i \leq x_i^{max} \quad (i = 1, 2, ..., n) \end{aligned} \tag{6}$$

where $\xi(x)$ and $\tau(x)$ are two functions to identify the estimated reliability and makespan required for schedule x respectively, Θ stands for the decision space (i.e., resource pool).

We use function $w(t_i)$ to identify which provider does task t_i is going to be scheduled into, $\xi(x)$ can thus be aggregated by the reliability of each D2D communication in scheduling plan as follows:

$$\xi(x) = log \prod_{i=1}^{|E|} RL_{(w(s_i),w(d_i))}[G(t)] \tag{7}$$

where s_i and s_i are father and child tasks in edge e_i respectively. The derivation of $\tau(x)$ requires some efforts, $\tau(x)$ can be calculated as the estimated end time of the last task t_m in a workflow:

$$\tau(x) = d_m \tag{8}$$

where d_m denotes the estimated end time of task t_m. We use d_i to denote the estimated end time of task t_i, it can be iteratively calculated as:

$$d_i = e_i + b_i \tag{9}$$

where b_i denotes the estimated start time of executing t_i and e_i the execution time of t_i itself. b_i is decided by the estimated end time of its immediately preceding tasks and the time required for data transfer. Let γ_i denote the estimated time when all earlier tasks scheduled to the same provider to t_i are finished, we have:

$$\gamma_i = max\{d_j \mid t_j \in {}^*t_i \wedge w(t_i) = w(t_j)\} \tag{10}$$

where ${}^*t_i$ denotes the immediately preceding tasks of t_i, i.e., those which directly connect t_i through edges in the workflow. $w(t_i) = w(t_j)$ indicates that t_i and t_j are scheduled into the same provider.

Note that the dependency constraint requires that a task be executed only if its immediately preceding ones successfully terminate and transfer data. We use y_i to denote the estimated earliest time when the described condition holds for t_i.

$$y_i = max\{d_k + D_{k,i} \mid t_k \ \in \ {}^*t_i \wedge \ w(t_k) \neq w(t_k) \ \} \tag{11}$$

The earliest possible time to execute b_i, can therefore be calculated as:

$$b_i = max\{\gamma_i, y_i\} \tag{12}$$

The first task of a workflow has no preceding task and therefore its estimated ending time is obtained as:

$$d_1 = \delta + e_1 \tag{13}$$

where δ is the time between receiving a workflow request and generating a corresponding schedule.

Since optimal reliability and makespan are two conflicting quality, we consider Pareto domination as the measure of the optimality of candidate solutions. Consequently, for solution $u, v \in \Theta$, u dominates v when:

$$\begin{cases} f_i(u) \leq f_i(v) & \forall \ i \in [1, n] \\ f_j(u) < f_j(v) & \exists \ j \in [1, n] \end{cases} \tag{14}$$

A solution x^* is Pareto-optimal if it is not dominated by any other solutions. The set of all Pareto-optimal solutions in the objective space is called a Pareto front. For the workflow scheduling problem, solution u dominates solution v if $\xi(u) \leq \xi(v) \wedge \tau(u) < \tau(v)$ or $\xi(u) < \xi(v) \wedge \tau(u) \leq \tau(v)$.

5 Multi-objective Differential Evolution for Workflow Scheduling

For the problem formulated in last section, methods such as multiple-objective-integer-linear-programming and multi-objective-branch-and-bound can be used for solutions. However, such method are usually considered to be with high time-complexity and thus could be impractical due to the fact that the problem space could be very large (the number of candidate mobile resource providers for one task can be 100+ for some typical cases, e.g., shopping mall and subway station. The number of tasks could be 50+ for some typical complex scientific application, e.g., Montage and Cybershake). In contrast, Multi-objective differential evolution (MODE) has been shown to be a simple yet efficient evolutionary algorithm for multi-objective optimization problems in diverse domains. It is featured by its strong parallelizability of genetic operators and good convergence

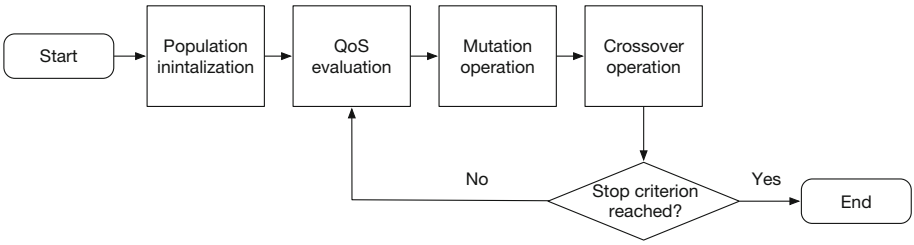


Fig. 6. The process of MDEWS.

properties than other traditional EMO algorithms. For the above problem formulated, we propose an improved MODE algorithm, named MDEWS, short for Multi-objective Differential Evolution for Workflow Scheduling to find solutions.

MDEWS developed in this work is a kind of meta-heuristic procedure similar to the process of natural selection. The process of MDEWS is shown in Fig. 6, it is used to yield high-quality solutions for optimization and searching problems by employing bio-inspired operations, e.g., mutation and crossover. A population of its candidate solutions to an optimization problem keeps evolving toward better solutions.

5.1 Encoding and Population Initialization

In MDEWS, a schedule is expressed as an individual which described by a vector of positive integers. The length of vectors are m, i.e., the number of tasks in a workflow. The i-th entry of the vector, in turn, refers to the mobile provider which i-th task in the workflow is scheduled into. Figure 7 shows the encoding scheme of a schedule and its deployment details for the sample workflow given earlier. In this schedule, task t_1, t_2, t_7 are scheduled into provider p_2 to execute, t_3, t_4, t_8 are scheduled into p_1, t_5 is scheduled into p_3 and t_6 is scheduled into p_4.

MDEWS starts with population initialization. The initial population with y individuals consists of three parts: (1) one individual i_1 with the highest $\xi(i_1)$ regardless of its makespan; (2) one individual i_2 with the shortest $\tau(i_2)$ regardless of its reliability; and (3) $y-2$ individuals are randomly generated according to the current resource provider pool.

5.2 Mutation

The mutation operator simulates the evolutionary activity that an individual directionally learns from other individuals. To speed up the convergence and optimize the exploration ability, we consider an improved mutation strategy as follows:

$$V_i = X_i + F(X^* - X_i) + F(X^\# - X_i) + F(X_r^1 - X_r^2) \tag{15}$$

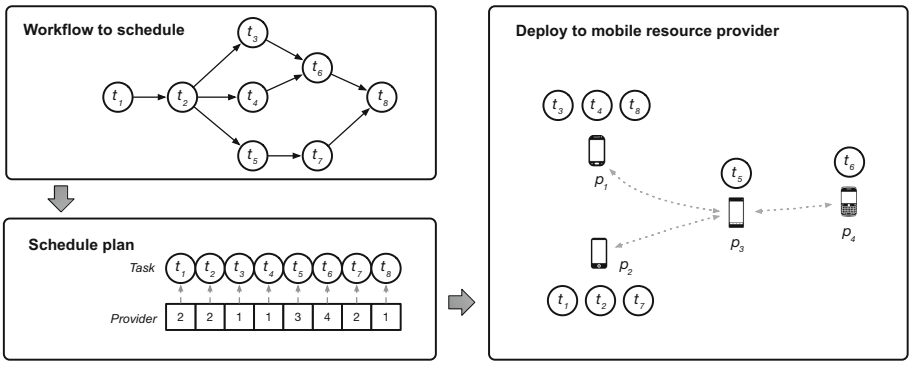


Fig. 7. An example of encoding.

where F is a scale factor, V_i an offspring individual, X_i mutation target, X_r^1 and X_r^2 two random individuals chosen from the current population, X^* and $X^\#$ the individuals randomly chosen from top-k best individuals in the population ordered by their estimated reliability and makespan, respectively, k is set to 15% in this paper. This top-k strategy can accelerate the convergence speed and in the meantime avoiding trapping into local optima. As shown in Algorithm 1, mutation operator first ranks individuals in a population order by its makespan and reliability and get the top 15% best group. Then it chooses base individuals X_r^1 and X_r^2 from current population and target individuals X^* and $X^\#$ from top 15% best group randomly. Finally, mutated individuals are generated by Eq. (15) based on base individuals and target individuals.

The time complexity of ranking all individuals and selecting top 15% best group (as shown lines 1–3 in Algorithm 1) are $O(y \log y)$, mutating all individuals (as shown lines 4–16 in Algorithm 1) is $O(my)$. Generally, m is large than $\log y$, therefore the time complexity of mutation operator is $O(my)$.

5.3 Crossover

The crossover operator simulates the genetic activity that an individual obtains characteristics from other individuals controlled by a crossover rate. As shown in Algorithm 2, the dynamic changing crossover rates are employed to avoid useless crossover operations. The crossover rate in the i-th generation, C_i, is randomly generated from a Gaussian distribution as:

$$C_i = \mathbf{G}(C_m, 0.1) \tag{16}$$

where C_m is calculated from the historical value of C_i, C_m in its first generation is 0.6. We use $\mathbb{C}$ to indicate the set of crossover rates which used in previous generations. C_m thus can be calculated as follows:

$$C_m = w_C \times C_m + (1 - w_C) \times mean_{Pow}(\mathbb{C}) \tag{17}$$

Algorithm 1. Mutation operator

Input: Population X; Task count m; Scale factor F; Resource pool P
Output: Mutated population V;

```
rank individuals in X order by its makespan and reliability
Top_Rel ← get top 15% best individuals according to estimated reliability
Top_Ms ← get top 15% best individuals according to estimated makespan
for each individual X_i in population X do
    X_r^1 ← choose one individual from X randomly
    X_r^2 ← choose one individual from X randomly
    X^* ← choose one individual from Top_Rel randomly
    X^# ← choose one individual from Top_Ms randomly
    V_i ← X_i + F(X^* − X_i) + F(X^# − X_i) + F(X_r^1 − X_r^2)
    for j = 1 to m do
        if V_i[j] < P.LowBounds[j] or V_i[j] > P.UpperBounds[j] then
            V_i[j] ← choose one provider between low bounds and upper bounds of
            providers randomly
        end if
    end for
    add V_i into mutated population V
end for
return V
```

where

$$mean_{Pow}(\mathbb{C}) = \sum_{i=1}^{|\mathbb{C}|}[\frac{(C_i)^n}{|\mathbb{C}|}]^{\frac{1}{n}} \tag{18}$$

where w_C is a real value randomly generated from $[0.9, 1]$ and n is set to 1.5 in this paper. The time complexity of crossover operator is $O(my)$.

5.4 Complexity Analysis

The overall computational complexity of our proposed approach can be analyzed by examining its position prediction, population initialization, reliability and makespan evaluation, mutation, crossover and dominance selection. Suppose there are k available mobile resource providers in a MRSC, the time complexity of forecasting all providers' future position is $O(k^2)$. The time complexity of initializing an individual is $O(m)$, and thus population initialization requires $O(my)$ where y is the size of initial population. The reliability and makespan evaluation for each individual has the time complexity of $O(mlog|E|)$ and thus reliability and makespan evaluation for initial population of size y with ω generations has the time complexity of $O(y\omega mlog|E|)$. The time complexity for mutation, crossover, and dominance selection operations are $O(my)$, $O(my)$, and $O(y^2)$, respectively. Consequently, the total time complexity of motion, crossover and dominance selection with ω generations is $O(\omega my) + O(\omega my) + O(\omega y^2)$. Finally, the total time complexity of position prediction, population initialization, reliability and makespan evaluation, mutation, crossover and dominance

Algorithm 2. Crossover operator

Input: Population X; Mutated population V; History crossover rate $\mathbb{C}$; Task count m;

Output: Population after crossover operation X';

calculate $mean_{Pow}$ according to history crossover rate $\mathbb{C}$ by (18)
calculate C_m by (17)
calculate crossover rate C_i by (16)
for each individual X_i in population X **do**
  for $j = 1$ **to** m **do**
    if $rand() < C_i$ **then**
      $Cv[j] \leftarrow 1$
    else
      $Cv[j] \leftarrow 0$
    end if
  end for
  for $j = 1$ **to** m **do**
    $X_i'[j] \leftarrow X_i \wedge (1 - Cv[j]) + V_i \wedge Cv[i]$
  end for
  add X_i' into X'
end for
return X'

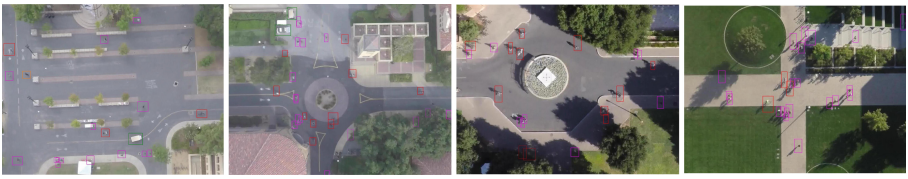
selection is thus $O(k^2) + O(y\omega mlog|E|) + O(\omega my) + O(\omega my) + O(\omega y^2)$. Generally, $mlog|E|$ is large than y, thus the total time complexity of our approach is $O(k^2 + y\omega mlog|E|)$, and such complexity suggests good scalability.

6 Experiments and Analysis

To evaluate the effectiveness of our approach, we conducted experiments on a real-world user trajectory dataset and multiple scientific workflow templates in a wide range of application scenes.

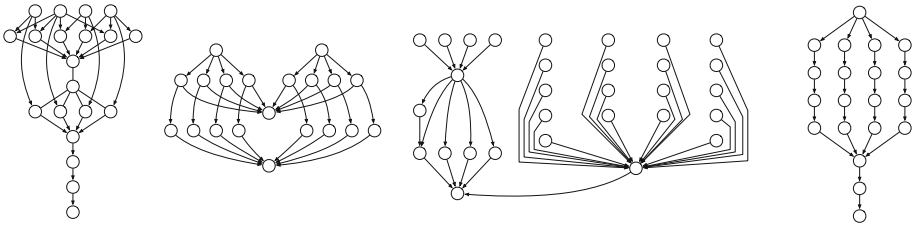
The Stanford Drone dataset [19] is a user trajectory dataset collected from Stanford campus. In this dataset, all pedestrians' movement trajectories in a certain scene are recorded for consecutive periods. We choose *bookstore*, *gates*, *deathcicle*, and *hyang* these four scenes with varying crowd density to conduct our experiments. The aerial views of four scenes are shown in Fig. 8, and we assume that pedestrians within each others' D2D communication distances in the same scene establish an MRSC.

Pegasus project [6] has released a real-world scientific workflow dataset which includes Montage, CyberShake, Sipht and so on. In this dataset, the structure of DAG, size of tasks and data transferring are well recorded. Besides, it also provides a reference execution time of each task. In this paper, we use Montage, CyberShake and Sipht these three most common scientific workflow templates and one randomly generated workflow template to conduct our experiments. The structures of these workflows are shown in Fig. 9.



(a) bookstore (b) deathcircle (c) gates (d) hyang

Fig. 8. The aerial view of experiment scenes.



(a) Montage (b) Cybershake (c) Sipht (d) Random

Fig. 9. Workflow templates used for experiment.

We first evaluate the exploitation ability of MDEWS and its peers towards the problem of workflow scheduling. Figure 10 shows the trade-off between reliability and makespan (i.e., pareto fronts) get by MDEWS and it peers. We consider NSGA-II, MOPSO, MOEA\D and SPEA2 as baseline algorithms because they are most widely used methods in solving multi-object workflow scheduling problem. It can be clearly seen that MDEWS can yield better pareto fronts in most cases, NSGA-II performs close to ours, then SPEA2 and MOPSO, MOEA\D cannot get a full pareto front in some cases such as *deathcicle* scene with Sipht workflow and *hyang* scene with Montage workflow and Sipht workflow.

To make a more clear comparison, HV values (a comprehensive evaluation index used to judge a multi-objective optimization method, the higher the better) are used to show the differences between MDEWS and it peers. As shown in Table 1, the ratios are used to offer a clearer comparison, for example, the HV improvement ratio can be calculated as follows:

$$\frac{HV(MDEWS)}{HV(Peer)} - 1 \tag{19}$$

Similarly, the comparison ratio of algorithms' runtime can be calculated as follows:

$$\frac{RunTime(Peer)}{RunTime(MDEWS)} \tag{20}$$

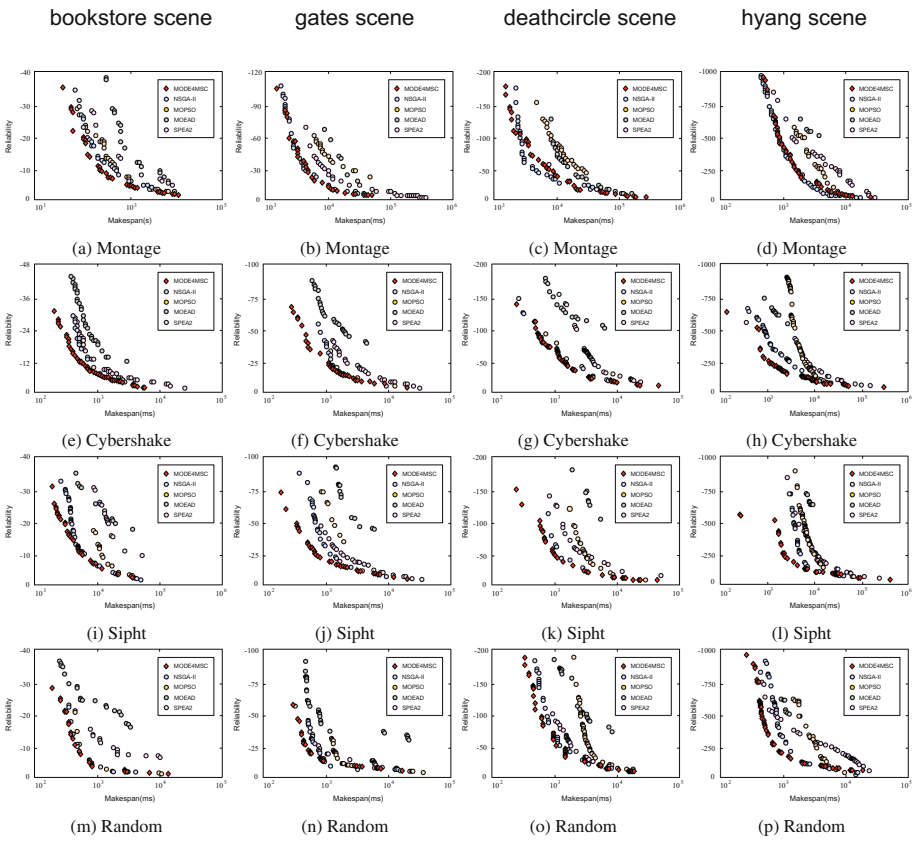


Fig. 10. Trade-off between reliability and makespan for different multi-objective optimization methods.

It can be seen that, MDEWS achieves a higher HV value in most cases. This advantage is achieved in a way that the individuals, with the help of MDEWS, are more likely to learn from a group of other individuals with high reliability and low makespan estimates, rather than learning from a single individual with seemingly highest optimality achieved by traditional algorithms. It also shows that MDEWS achieves higher time-efficiency in all cases (2 times faster than MOPSO on average, 3 times faster than MODE and SPEA2 on average).

We also compared MDEWS with traditional non-prediction-based workflow scheduling algorithms (EMS-C [24], PSO [15], and CEGA [12]) which assume stable resource reliability, in terms of the success rate of workflow execution. As shown in Fig. 11, our proposed approach clearly outperforms non-prediction-based approaches in most cases. To be specific, the success rate achieved by our method is 3.74%, 9.25%, 12.35% and 20.51% higher success rate than EMS-C on average in four scenes, respectively; 5.20%, 9.51%, 12.33% and 23.40% higher than PSO; and 6.05%, 10.94%, 15.02% and 23.07% higher than CEGA. Note that

Table 1. HV and runtime comparison between MDEWS and traditional MOO Algorithms

Scene	Workflow	NSGA-II		MOPSO		MOEA\D		SPEA2	
		HV	Runtime	HV	Runtime	HV	Runtime	HV	Runtime
Bookstore	Montage	−5.81%	1.21	−8.27%	1.51	121.48%	2.43	14.65%	3.04
	Cybershake	−0.47%	1.27	−17.85%	1.62	19.54%	2.68	8.57%	3.28
	Siph	−4.90%	1.31	117.54%	1.69	83.80%	2.93	5.03%	3.43
	Random	1.03%	1.28	13.10%	1.75	117.25%	3.01	61.59%	3.62
Gates	Montage	8.58%	1.13	27.69%	1.51	23.03%	2.06	11.57%	2.73
	Cybershake	10.12%	1.24	21.95%	1.59	61.14%	2.31	8.84%	2.89
	Siph	5.70%	1.25	64.75%	1.65	127.19%	2.94	73.91%	3.41
	Random	8.39%	1.31	36.12%	1.69	117.53%	3.03	42.62%	3.63
Deathcircle	Montage	3.91%	1.07	23.96%	1.54	77.42%	2.13	112.93%	2.63
	Cybershake	7.34%	1.14	77.49%	1.63	132.18%	2.31	16.40%	2.91
	Siph	12.63%	1.19	113.15%	1.65	61.91%	2.53	53.67%	3.13
	Random	6.01%	1.22	124.37%	1.71	143.27%	2.61	22.01%	3.28
Hyang	Montage	−0.24%	1.05	4.76%	1.52	141.57%	2.15	71.87%	2.77
	Cybershake	9.09%	1.12	23.97%	1.53	188.79%	2.26	13.75%	2.90
	Siph	11.75%	1.15	131.06%	1.55	131.22%	2.40	23.77%	3.15
	Random	17.03%	1.20	140.38%	1.59	117.80%	2.51	27.31%	3.32

these four experiment scenes are with vary density of mobile users (*bookstore* about 3428 per km^2, *gates* about 4981 per km^2, *deathcicle* about 6571 per km^2, and *hyang* about 8714 per km^2). The experimental results also show that the more crowded of the mobile users, the better MDEWS performs than traditional non-prediction-based methods.

When the density of mobile users in an MRSC is sparse, there are few resource providers for requester to choice, therefore baseline algorithms perform close to ours. But when the density of mobile users become dense, there will be more resource providers available. Under such condition, traditional non-prediction methods trend to schedule tasks into providers with high reliability and low makespan to meet deadline constrain and QoS requirement. However, as we mentioned earlier, mobile users are keep moving during the resource provisioning, which means that providers which are reliable at the current state are not guaranteed to keep its reliability in the future. In contrast, with the help of user position prediction and reliability evaluation methods, MDEWS prefers to choice those service providers with good expected reliability and acceptable response time. In this way the knowledges and patterns behind users' movement are properly mined to generate more reliable schedules.

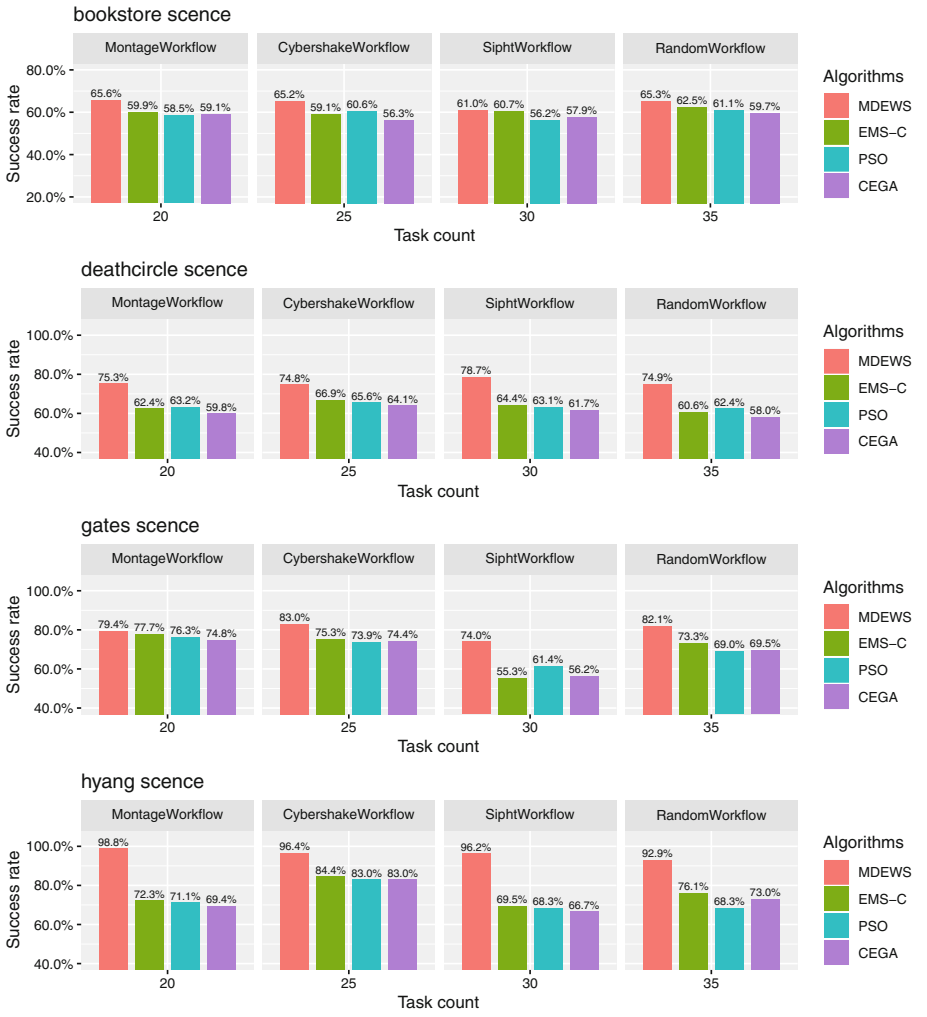


Fig. 11. Success rate comparison between MDEWS and non-prediction-based approaches.

7 Conclusion and Further Work

This paper targets at the problem of unreliable workflow scheduling under the mobile computing environment, and proposed a position-prediction-based mobile workflow scheduling approach in the context of MANET. We evaluate the reliability of mobile resource providers dynamically based on predicted user positions through a Gaussian mixture prediction model. Mobile providers are selected and schedule plans are generated by an evolutionary multi-objective optimization algorithm.

We consider hard deadline in this paper, as future work, we plan to consider soft deadline constraints (where makespan is allowed to exceed a threshold value with a bounded given rate) and introduce corresponding algorithms to generate run-time schedules. Besides, some learning-based method will be employed to achieve a smarter scheduling.

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. Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S.: Social LSTM: human trajectory prediction in crowded spaces. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 961–971 (2016)
3. Balasubramanian, N., Balasubramanian, A., Venkataramani, A.: Energy consumption in mobile phones: a measurement study and implications for network applications. In: Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement, pp. 280–293. ACM (2009)
4. Deb, K.: Multi-objective optimization. In: Burke, E., Kendall, G. (eds.) Search Methodologies, pp. 403–449. Springer, Boston (2014). https://doi.org/10.1007/978-1-4614-6940-7_15
5. Giordano, S., Puccinelli, D.: The human element as the key enabler of pervasiveness. In: The 10th IFIP Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net) 2011, pp. 150–156. IEEE (2011)
6. ISI: Pegasus Project. https://confluence.pegasus.isi.edu (2018). Accessed 26 Aug 2018
7. Kharbash, S., Wang, W.: Computing two-terminal reliability in mobile ad hoc networks. In: Wireless Communications and Networking Conference 2007, WCNC 2007. pp. 2831–2836. IEEE (2007)
8. Li, W., Xia, Y., Zhou, M., Sun, X., Zhu, Q.: Fluctuation-aware and predictive workflow scheduling in cost-effective infrastructure-as-a-service clouds. IEEE Access **6**, 61488–61502 (2018)
9. Liu, S., Cao, H., Li, L., Zhou, M.: Predicting stay time of mobile users with contextual information. IEEE Trans. Autom. Sci. Eng. **10**(4), 1026–1036 (2013)
10. Maheswaran, M., Ali, S., Siegal, H., Hensgen, D., Freund, R.F.: Dynamic matching and scheduling of a class of independent tasks onto heterogeneous computing systems. In: Proceedings of the Eighth Heterogeneous Computing Workshop 1999, (HCW 1999), pp. 30–44. IEEE (1999)
11. Mao, M., Humphrey, M.: Auto-scaling to minimize cost and meet application deadlines in cloud workflows. In: 2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC), pp. 1–12. IEEE (2011)
12. Meena, J., Kumar, M., Vardhan, M.: Cost effective genetic algorithm for workflow scheduling in cloud under deadline constraint. IEEE Access **4**, 5065–5082 (2016)
13. Microsoft: RSSI. https://msdn.microsoft.com/en-us/library/windows/desktop/ms706828%28v=vs.85%29.aspx (2018). Accessed 26 Aug 2018
14. Qiao, S., Han, N., Zhu, W., Gutierrez, L.A.: TraPlan: an effective three-in-one trajectory-prediction model in transportation networks. IEEE Trans. Intell. Transp. Syst. **16**(3), 1188–1198 (2015)

15. 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)
16. Sakellariou, R., Zhao, H.: A hybrid heuristic for DAG scheduling on heterogeneous systems. In: Proceedings of the 18th International Parallel and Distributed Processing Symposium 2004, p. 111. IEEE (2004)
17. Schad, J., Dittrich, J., Quiané-Ruiz, J.A.: Runtime measurements in the cloud: observing, analyzing, and reducing variance. Proc. VLDB Endow. **3**(1–2), 460–471 (2010)
18. Song, C., Qu, Z., Blumm, N., Barabási, A.L.: Limits of predictability in human mobility. Science **327**(5968), 1018–1021 (2010)
19. Stanford-CVGL: Stanford Drone Dataset. http://cvgl.stanford.edu/projects/uav_data/ (2018). Accessed 26 Aug 2018
20. 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)
21. Wu, Q., Ishikawa, F., Zhu, Q., Xia, Y., Wen, J.: Deadline-constrained cost optimization approaches for workflow scheduling in clouds. IEEE Trans. Parallel Distrib. Syst. **28**(12), 3401–3412 (2017)
22. Xia, Y., Zhou, M., Luo, X., Pang, S., Zhu, Q.: Stochastic modeling and performance analysis of migration-enabled and error-prone clouds. IEEE Trans. Ind. Inf. **11**(2), 495–504 (2015)
23. Zeng, H., Cheung, Y.M.: A new feature selection method for Gaussian mixture clustering. Pattern Recogn. **42**(2), 243–250 (2009)
24. Zhu, Z., Zhang, G., Li, M., Liu, X.: Evolutionary multi-objective workflow scheduling in cloud. IEEE Trans. Parallel Distrib. Syst. **27**(5), 1344–1357 (2016)