Int j simul model 19 (2020) 1, 146-156
ISSN 1726-4529 Original scientific paper

A BI-OBJECTIVE OPTIMIZATION ALGORITHM FOR AUTOMOBILE MANUFACTURING SCHEDULING

Alatangaowa, B.[*,**]; Batbileg, S.[***,#] & Enkhbat, R.[***]
[*] Normal School, Chifeng University, Chifeng 024000, China
[**] Department of Mathematics, Mongolian National University of Education, Baga toiruu-14, Ulaanbaatar 48, Mongolia
[***] National University of Mongolian, Baga toiruu-1, Ulaanbaatar 46, Mongolia
E-Mail: batbileg@seas.num.edu.mn ([#] Corresponding author)

Abstract
In automobile manufacturing, it is a great challenge to select and execute the services with the optimal value, cost and time out of complex processes. Most traditional algorithms only optimize one objective. In this paper, an optimization algorithm for service value and time is proposed under the constraint of deadline, and denoted as SRVT. The proposed algorithm reversely derives the service with the maximum value at each time point, and adds it to the set of candidate solutions in the next iteration. Then, the optimal solutions were selected iteratively from the set. In the end, the maximum service value of the entire workflow was obtained. The proposed SRVT was compared with two traditional algorithms through a case study. The comparison shows that our algorithm can outperform the contrastive algorithms to a certain extent, and strike a balance between service time, service cost and service quality.
(Received in September 2019, accepted in January 2020. This paper was with the authors 1 month for 1 revision.)

Key Words: Automobile Manufacturing, Workflow, Scheduling Optimization, Maximal Service Quality, Deadline

1. INTRODUCTION

Workflow, a computer-aided full- or semi-automation technique, has been widely adopted in business and production processes. In general, a workflow model covers the start and termination conditions of the entire process, the specific information on each task, as well as the logical sequence between tasks and between tasks and activities.

The rational scheduling lies at the core of workflow. Luo et al. [1] enumerated various scheduling strategies, and thoroughly demonstrated that a rational scheduling algorithm can optimize the cost and time of workflow. Every workflow consists of multiple tasks, each of which can be completed by executing the corresponding service. This service is usually selected based on the quality of service (QoS) and the business demand of the target user. Hence, the key of scheduling strategy is to choose the optimal service that satisfies user demand [2-3].

Currently, the QoS is mainly measured by two reference indices: execution time and execution cost. The workflow is often modelled, and divided into a set of loose tasks in different domains. Then, the path with the least cost is searched for by heuristic scheduling algorithms before the deadline [4].

Similar to query optimization problem, the workflow problem is non-deterministic polynomial-time (N-P) hard [5-7]. Time and cost are the main objectives of workflow optimization. In other words, the quality of service must be ensured within a short time at a low cost.

2. LITERATURE REVIEW

Most workflows are control-oriented and aimed at reducing labour time and labour cost. To minimize the time and cost, the scheduling strategy of a workflow should dynamically select the service to be executed before the deadline, striking a balance between the goals of all parties.

Xu et al. [8] explored deep into the cloud service reliability of workflow scheduling, and developed a heuristic time-cost trade-off algorithm, in which the workflow execution is optimized to realize the dynamic equilibrium between the minimal total makespan and the minimal total execution cost.

Ahmad et al. [9] studied the multi-objective optimization of workflow scheduling in heterogeneous computing systems, and combined genetic operators and heuristic algorithm into an optimization algorithm. The proposed algorithm can converge to the optimal solution in a short time, and optimize the multiple objectives of cloud workflow scheduling in heterogeneous computing environment.

Alkhanak et al. [10] designed a cost-based met classification method to minimize the workflow execution cost, and satisfy the user-defined QoS. Based on scheduling stages, the metrics were divided into monetary and temporal cost parameters. Their method sheds new light on multi-objective optimization of workflow scheduling in various fields.

Based on multi-agent system (MAS), Hsieh [11] put forward a scalable and sustainable scheduling system, which relies on workflow scheduling and Petri net to optimize the distributed organization of hospital, the dynamic medical workflow and the variable medical resources.

Al-Kiswany et al. [12] developed a workflow-optimized storage system (WOSS), which exploits application hints to provide per-file optimized operations, and exposes data location to enable multi-objective optimization of location-aware scheduling.

Shishido et al. [13] described various metaheuristic scheduling techniques for cloud and their applications, optimized workflow scheduling with particle swarm optimization (PSO) [14] and genetic algorithm (GA) [15], and achieved the dynamic balance between safety, cost and risk. Considering the complexity of cloud computing, Verma and Kaushal [16] designed a hybrid particle swarm optimization (HPSO) algorithm based on non-dominance sorting. The HPSO is a hybrid of budget and deadline constrained heterogeneous earliest finish time (BDHEFT) algorithm and multi-objective PSO. The proposed algorithm gives a set of Pareto Optimal solutions, and effectively schedules the workflow with multiple conflicting objectives in infrastructure-as-a-service (IaaS) clouds.

Considering the difficulty in balancing the multiple conflicting objectives in service-orientated grids (SOG) environments, Ambursa et al. [17] created a multi-QoS workflow scheduling algorithm based on the PSO and a look-ahead heuristic (LAPSO), which selects the best scheduling solutions based on the proposed constraint-handling strategy and improves the quality of the best solutions.

Considering the heterogeneous nature, complex billing models and infinite resources requirements of workflow scheduling in cloud platform, Prathibha et al. [18] presented a high-performance method for scheduling workflow on cloud computing platform, called non-dominated sorting particle swarm optimization (NSPSO). The NSPSO provides better workflow scheduling solution for cloud than the existing workflow scheduling algorithms.

Abdi et al. [19] proposed a novel greedy randomized adaptive search procedure (GRASP) called GRASP-FC, with the aim to reduce the financial cost like the fees for running virtual machines (VMs) and the fees for data transfer, and to fulfil deadline and resource constraints in the clouds. The GRASP-FC provides a valuable tool to obtain an approximate optimal solution to resource allocation for bag-of-tasks (BoT) workflows.

Based on evolutionary states, Zhang et al. [20] formulated an adaptive individual-assessment scheme to handle the constraints in multi-objective optimization problems. Experimental results show that the proposed algorithm outperformed other state-of-the-art methods in convergence and diversity, and achieved better optimization ability in solving cloud workflow scheduling problem.

Almi'ani et al. [21] presented a resource demand aware scheduling (RDAS+) algorithm, which maximizes the resource utilization by allocating the minimum number of resources. This optimization eventually leads to cost efficiency for pay-per-use cloud resources.

Focusing on the total makespan and execution cost of workflow, Ghafouri et al. [22] proposed two heuristic algorithms, namely, minimum time and decreased cost (MTDC) and constrained time and decreased cost (CTDC). Through comparative analysis, the two algorithms were found to perform better than heterogeneous earliest finish time (HEFT) algorithm and the PSO in scheduling results and performance.

Whereas most of the heuristics or meta-heuristics may not fulfil certain optimum criterion and produce near optimal solution, Choudhary et al. [23] set up a meta-heuristic based algorithm for workflow scheduling that considers minimization of makespan and cost. Simulation results show that the proposed algorithm outperformed traditional single-objective scheduling optimization algorithms in all scenarios.

Targeting the workflow scheduling in cloud environment, Naidu and Bhagat [24] put forward a modified PSO with scout adaptation (MPSO-SA) algorithm. Compared with conventional scheduling algorithms, the MPSO-SA scheduling mechanism lowers the probability of security risk on job scheduling.

Luo et al. [25-26] developed a virtual iterative reduction algorithm for the manufacturing of complex products. Through reverse iteration, the algorithm specifies a scheduling path that balances makespan and production accuracy, and optimizes the multi-objective scheduling of manufacturing businesses for complex products.

Qureshi [27] investigated the role of application profiles in addressing the trade-off between performance and energy efficiency of small- to medium-scale data centres, and proposed a power-aware framework for efficient placement of application workloads in the data centre. Simulation results show that the proposed framework was 19 % and 38 % more energy efficient than robust time cost (RTC) and HEFT, respectively for medium to large sized workloads.

Kalra and Singh [28] created an approach based on intelligent water drops (IWD) algorithm to minimize the execution time of workflows while balancing the resource utilization of VMs in cloud computing environment. Experimental results demonstrate that proposed algorithm performed better than these existing techniques in terms of makespan and load balancing.

Taking dynamic workflow scheduling problem as a dynamic multi-objective optimization problem (DMOP), Ismayilov and Topcuoglu [29] developed a prediction-based dynamic multi-objective evolutionary algorithm, which achieves the optimal scheduling of six objectives: minimal makespan, minimal cost, minimal energy, minimal degree of imbalance, maximal reliability and maximal utilization.

Drawing on the above literature, this paper attempts to offer a scheduling optimization strategy that can select and execute a suitable service in each local process, and thus obtain the optimal solution: maximizing the service quality when the time is fully utilized. The service quality was measured by the QoS per unit time (i.e. service value)

3. PROBLEM DESCRIPTION

3.1 Relevant definitions

Definition 1. Service value S_{val} refers to the ratio of service quality S_{qlty} to service price S_{pc} per unit time t, i.e. $S_{val} = S_{qlty} \cdot t / S_{pc}$.

Definition 2. Task package W is the set of all tasks in a workflow. Let $n \in \mathbf{Z}^+$ be the number of tasks. Then, task package can be expressed as $W = \{w_1, w_2, \ldots, w_i, \ldots, w_n\}$.

Definition 3. Service package F_i is the set of services available for a specific task w_i. Let $m \in \mathbf{Z}^+$ be the number of services. Then, service package can be expressed as $F_i = \{f_{i,1}, f_{i,2}, \ldots, f_{i,j}, \ldots, f_{i,m}\}$.

Definition 4. Service details $P_{i,j}$ refer to the specific parameters of a service, including service quality, service price and service time. Let $S_{qlty<i,j>}$, $S_{pc<i,j>}$, and $t_{i,j}$ be the service quality, service price and service time of the j^{th} service of task w_i, respectively. Then, service details can be expressed as $P_{i,j} = (S_{qlty<i,j>}, S_{pc<i,j>}, t_{i,j})$.

Definition 5. Workflow model M is usually described by a directed acyclic graph (DAG), in which the nodes and edges represent the tasks in the workflow and the sequence between the tasks, respectively. The workflow model can be expressed as $M = (W, E)$, where E is a set of directed edges ($E = \{(i, k) \mid$ task k is the direct successor of task $i\}$. For convenience, two virtual nodes w_{start} and w_{end} were introduced to indicate the start and the end of a service. Then, the service packages at the start and end of a service can be denoted as F_{start} and F_{end}, respectively. The core of workflow optimization is to select services from service packages, according to the specific situation.

Definition 6. Task execution domain $RA_i = [ET_i, LT_i]$ refers to a closed period allowed to execute task w_i, where ET_i and LT_i are the earliest time and latest time allowed to execute task w_i, respectively. The actual start time of task i falls between the earliest time and latest time, i.e. $I_i \in [ET_i, LT_i]$.

$$\begin{cases} ET_i = \sum_{k=1}^{i-1} t_{k-\min}, ET_1 = 0 \\ LT_i = DL - \sum_{k=i}^{n} t_{k-\min} \\ t_{k-\min} = \min\left\{t_{k,1}, \ldots, t_{k,j}, \ldots, t_{k,m}\right\}, t_{k,j} \in F_i \end{cases} \tag{1}$$

where, DL is the deadline of the entire workflow.

3.2 Workflow diagram generation algorithm (WDG)

Workflow diagram is a popular way to illustrate how a workflow operates. Based on the previous literature, this paper designs a workflow diagram generation algorithm (WDG), which considers the sequence between tasks and the parallel relationship between services:

Step 1. Determine the execution sequence of the tasks in the task package, according to the logical sequence of businesses.

Step 2. Take F_{start} as the direct predecessor of all services in the first task, and F_{end} as the direct successor of all services in the last task.

Step 3. Traverse the service package of the next task, take each service in the package as the direct successor of the corresponding service in the package of the current task, and specify the details of all these services.

Step 4. Repeat Step 3 until the task package is traversed, creating the workflow diagram.

The pseudo code of our WDG is as follows:

Inputs: $W \leftarrow w_i (i = 1, 2, \ldots, n)$// The set of tasks in the workflow

$F_i \leftarrow f_{i,j}$ ($j = 1, 2, \ldots, m$)//The set of services of task i
$P_{i,i} \leftarrow S_{qlty<i,j>}, S_{pc<i,i>}, t_{i,i}$// The details of each service
F_{start}, F_{end}// The service packages at the start and end of a service
$G \leftarrow \varnothing$

Output: Workflow diagram

1. FOR (i = 1; i<= W.length; i++)// Traverse task package W.
2. FOR (j = 1; j<= F_i.length; j++) // Traverse service package F_i.
3. IF (i ==1) // Judge if it is the first task.
4. addEdge(F_{start}, $f_{i,j}$)// Add service $f_{1,j}$ to the diagram as the direct successor to F_{start}.
5. ELSE IF (i ==W.length) // Judge if it is the last task.
6. addEdge($f_{i,j}$, F_{end})// Add service F_{end} to the diagram as the direct successor to $f_{m,j}$.
7. ELSE
8. FOR(k = 1; F_{i-1}.length; k++)// Add each service in the next service package to the diagram
9. addEdge($f_{i-1,k}$, $f_{i,j}$)//as the direct successor to the corresponding service of the current service package.
   The services in the next service package are used as the previous one in turn.
10. addProperty($f_{i,j}$, $P_{i,j}$)// Add the details corresponding to the services to the diagram.
11. RETURN Workflow diagram

The time complexity of the WDG is $O(n \cdot m^2)$.

3.3 Constraints

In addition to the above definitions, the workflow must obey the following constraints before being applied in actual production:

$$\begin{cases} I_i + t_{i,sel} \le LT_{i+1} \\ t_{total} = \sum_{i=1}^{n} t_{i,sel} \le DL \\ 1 \le i \le n, 1 \le sel \le m \end{cases} \tag{2}$$

where, I_i is the start time for the execution of task i; $t_{i,sel}$ is the execution time of a specific service in task i; t_{total} is the total execution time of the workflow.

3.4 Deadline-constrained service value-time optimization algorithm (SRVT)

This paper puts forward a service value-time optimization algorithm under the constraint of deadline, and denotes it as the SRVT. The core idea is to divide the workflow into multiple phases, and select the best quality service in the current phase, without surpassing the deadline DL. In other words, the workflow optimization aims to maximize the service value within the task execution domain. The maximum service value of the workflow can be reversely derived from the maximum service values of sub-workflows:

$$\begin{cases} \theta(n, I_n) = \max\{S_{val<n,1>}, \ldots, S_{val<n,j>}, \ldots, S_{val<n,m>}\} \\ t_{n,sel} + I_n \le DL \end{cases} \tag{3}$$

where, $S_{val<n,j>}$ is the service value of service j in task n; $\theta(n, I_n)$ is the maximum total service value of the last task n with the actual start time I_n:

$$\theta(n\text{-}1, I_{n-1}) = \max\left\{\theta\left(n, I_{n-1} + t_{n-1,sel}\right) + S_{val<n-1,sel>}\right\} \tag{4}$$

The SRVT is implemented in the following steps:

Step 1. Transform the input data into the nodes and edges in the DAG, using the WDG algorithm.

Step 2. Compute the execution domain [*ET*, *LT*] of all tasks in the input task package.

Step 3. Reversely derive the maximum service value of the entire workflow, after the start time of each service within the execution domain of a task.

Step 4. Compare the service values of different services at the end of iteration, and take the maximum value as the maximum service value.

4. CASE STUDY

The workflow of automobile manufacturing was abstracted into a task package *W*. The ten tasks in the package include casting w_1, forging w_2, pressing w_3, welding w_4, plastic processing w_5, mechanical processing w_6, hot treatment w_7, electroplating w_8, painting w_9, and assembly w_{10}.

Each process was abstracted into a service, which differs with the processing methods and prices. The available services were thus aggregated to service packages. The service packages for the ten tasks are $F_1 = \{f_{1,1}, f_{1,2}\}$, $F_2 = \{f_{2,1}, f_{2,2}, f_{2,3}, f_{2,4}\}$, $F_3 = \{f_{3,1}, f_{3,2}, f_{3,3}\}$, $F_4 = \{f_{4,1}, f_{4,2}, f_{4,3}\}$, $F_5 = \{f_{5,1}, f_{5,2}\}$, $F_6 = \{f_{6,1}, f_{6,2}, f_{6,3}\}$, $F_7 = \{f_{7,1}, f_{7,2}, f_{7,3}, f_{7,4}\}$, $F_8 = \{f_{8,1}, f_{8,2}\}$, $F_9 = \{f_{9,1}, f_{9,2}\}$ and $F_{10} = \{f_{10,1}, f_{10,2}, f_{10,3}\}$, respectively.

Two virtual nodes, w_{start} and w_{end}, were introduced to represent the supply and delivery of components.

If the details of a service are $P_{1,1} = (0.97, 5, 5)$, the quality, price and time of the first service in the first task package are 0.97, 5 and 5, respectively. The details of services are listed in Table I below.

Table I: Details of services.

Service package F_i	**Service value $S_{val<i,j>}$**	**Service details $P_{i,j}$**	**Service package F_i**	**Service value $S_{val<i,j>}$**	**Service details $P_{i,j}$**
F_1	0.553, 0.533	(0.83,6,4), (0.96,9,5)	F_6	0.486, 0.474, 0.581	(0.85,7,4), (0.79,5,3), (0.93,8,5)
F_2	0.474, 0.539, 0.491	(0.79,5,3), (0.97,9,5), (0.86,7,4)	F_7	0.474, 0.560, 0.550	(0.79,5,3), (0.84,6,4), (0.99,9,5)
F_3	0.462, 0.540, 0.533	(0.77,5,3), (0.81,6,4), (0.96,9,5)	F_8	0.462, 0.533	(0.77,5,3), (0.8,6,4)
F_4	0.544, 0.456, 0.533	(0.98,9,5), (0.76,5,3), (0.8,6,4)	F_9	0.456, 0.547	(0.76,5,3), (0.82,6,4)
F_5	0.474, 0.563	(0.79,5,3), (0.9,8,5)	F_{10}	0.503, 0.462, 0.544	(0.88,7,4), (0.77,5,3), (0.98,9,5)

4.1 Algorithm analysis

The duration of automobile manufacturing, i.e. the *DL* of the workflow, was assumed as 35. Under this assumption and the conditions in Table I, the results of three different algorithms are explained below:

(1) Shortest time algorithm (STA)

The STA aims to complete the workflow in the shortest time, without considering the service value of specific tasks. According to the data in Table I, the services were executed in the following sequence: $f_{1,2} \rightarrow f_{2,3} \rightarrow f_{3,3} \rightarrow f_{4,2} \rightarrow f_{5,2} \rightarrow f_{6,2} \rightarrow f_{7,3} \rightarrow f_{8,2} \rightarrow f_{9,2} \rightarrow f_{10,2}$.

Thus, the total execution time:

$t_{total} = t_{1,2}+t_{2,3}+t_{3,3}+t_{4,2}+t_{5,2}+t_{6,2}+t_{7,3}+t_{8,2}+t_{9,2}+t_{10,2} = 4+3+3+3+3+3+3+3+3+3 = 31 < DL$, and the total service value:

S_{val} = 0.553+0.474+0.462+0.456+0.474+0.474+0.474+0.462+0.456+0.462 = 4.747.

(2) Maximum service value algorithm (MVAL)

The maximum service value algorithm (MVAL) gives priority to the service with the maximum value, but its execution time might exceed the deadline. According to the data in Table I, the services were executed in the following sequence:
$f_{1,1}\rightarrow f_{2,2}\rightarrow f_{3,2}\rightarrow f_{4,3}\rightarrow f_{5,1}\rightarrow f_{6,1}\rightarrow f_{7,2}\rightarrow f_{8,1}\rightarrow f_{9,1}\rightarrow f_{10,1}$. Thus, the total service value:
S_{val} = 0.553+0.539+0.540+0.544+0.563+0.581+0.560+0.533+0.547+0.544 = 5.504, and the total execution time:
$t_{total} = t_{1,1}+t_{2,2}+t_{3,2}+t_{4,3}+t_{5,1}+t_{6,1}+t_{7,2}+t_{8,1}+t_{9,1}+t_{10,1}$ = 5+5+4+5+5+5+4+4+4+5 = 46 > *DL*. The algorithm is not feasible, for failing to meet the required duration.

(3) The SRVT

The execution domains of the tasks in the task package were derived by Eq. (1) as: RA_1 = [0,4], RA_2 = [4,8], RA_3 = [7,11], RA_4 = [10,14], RA_5 = [13,17], RA_6 = [16,20], RA_7 = [19,23], RA_8 = [22,26], RA_9 = [25,29] and RA_{10} = [28,32].

The maximum service value in each phase was reversely derived by Eqs. (1) and (2) as follows:

Phase 1: $\theta(10,28) = \max\{0.544, 0.462, 0.503\} = 0.544$,
$\theta(10,29) = \max\{0.544, 0.462, 0.503\} = 0.544$,
$\theta(10,30) = \max\{0.544, 0.462, 0.503\} = 0.544$,
$\theta(10,31) = \max\{0.462, 0.503\} = 0.503$,
$\theta(10,32) = \max\{0.462\} = 0.462$;

Phase 2: $\theta(9,25) = \max\{\theta(10,28)+0.456, \theta(10,29)+0.547\} = 1.091$,
$\theta(9,26) = \max\{\theta(10,29)+0.456, \theta(10,30)+0.547\} = 1.091$,
$\theta(9,27) = \max\{\theta(10,30)+0.456, \theta(10,31)+0.547\} = 1.050$,
$\theta(9,28) = \max\{\theta(10,31)+0.456, \theta(10,32)+0.547\} = 1.009$,
$\theta(9,29) = \max\{\theta(10,32)+0.456\} = 0.918$;

Phase 3: $\theta(8,22) = \max\{\theta(9,25)+0.462, \theta(9,26)+0.533\} = 1.624$,
$\theta(8,23) = \max\{\theta(9,26)+0.462, \theta(9,27)+0.533\} = 1.583$,
$\theta(8,24) = \max\{\theta(9,27)+0.462, \theta(9,28)+0.533\} = 1.542$,
$\theta(8,25) = \max\{\theta(9,28)+0.462, \theta(9,29)+0.533\} = 1.471$,
$\theta(8,26) = \max\{\theta(9,29)+0.462\} = 1.380$;

Phase 4: $\theta(7,19) = \max\{\theta(8,22)+0.474, \theta(8, 23)+0.560, \theta(8, 24)+0.550\} = 2.143$,
$\theta(7,20) = \max\{\theta(8,23)+0.474, \theta(8, 24)+0.560, \theta(8, 25)+0.550\} = 2.102$,
$\theta(7,21) = \max\{\theta(8,24)+0.474, \theta(8, 25)+0.560, \theta(8, 26)+0.550\} = 2.031$,
$\theta(7,22) = \max\{\theta(8,25)+0.474, \theta(8, 26)+0.560\} = 1.945$,
$\theta(7,23) = \max\{\theta(8,26)+0.474\} = 1.854$;

Phase 5: $\theta(6,16) = \max\{\theta(7,19)+0.474, \theta(7, 20)+0.486, \theta(7, 21)+0.581\} = 2.617$,
$\theta(6,17) = \max\{\theta(7,20)+0.474, \theta(7, 21)+0.486, \theta(7, 22)+0.581\} = 2.576$,
$\theta(6,18) = \max\{\theta(7,21)+0.474, \theta(7, 22)+0.486, \theta(7, 23)+0.581\} = 2.505$,
$\theta(6,19) = \max\{\theta(7,22)+0.474, \theta(7, 23)+0.486\} = 2.419$,
$\theta(6,20) = \max\{\theta(7,23)+0.474\} = 2.328$;

Phase 6: $\theta(5,13) = \max\{\theta(6,16)+0.474, \theta(6,18)+0.563\} = 3.091$,
$\theta(5,14) = \max\{\theta(6,17)+0.474, \theta(6, 19)+0.563\} = 3.050$,
$\theta(5,15) = \max\{\theta(6,18)+0.474, \theta(6, 20)+0.563\} = 2.979$,
$\theta(5,16) = \max\{\theta(6,19)+0.474\} = 2.893$,
$\theta(5,17) = \max\{\theta(6,20)+0.474\} = 2.802$;

Phase 7: $\theta(4,10) = \max\{\theta(5,13)+0.456, \theta(5, 14)+0.533, \theta(5, 15)+0.544\} = 3.583$,

$\theta(4,11) = \max\{\theta(5,14)+0.456, \theta(5,15)+0.533, \theta(5,16)+0.544\} = 3.512,$
$\theta(4,12) = \max\{\theta(5,15)+0.456, \theta(5,16)+0.533, \theta(5,17)+0.544\} = 3.435,$
$\theta(4,13) = \max\{\theta(5,16)+0.456, \theta(5,17)+0.533\} = 3.349,$
$\theta(4,14) = \max\{\theta(5,17)+0.456\} = 3.258;$

Phase 8: $\theta(3,7) = \max\{\theta(4,10)+0.462, \theta(4,11)+0.533, \theta(4,12)+0.540\} = 4.045,$
$\theta(3,8) = \max\{\theta(4,11)+0.462, \theta(4,12)+0.533, \theta(4,13)+0.540\} = 3.974,$
$\theta(3,9) = \max\{\theta(4,12)+0.462, \theta(4,13)+0.533, \theta(4,14)+0.540\} = 3.897,$
$\theta(3,10) = \max\{\theta(4,13)+0.462, \theta(4,14)+0.533\} = 3.811,$
$\theta(3,11) = \max\{\theta(4,14)+0.462\} = 3.720;$

Phase 9: $\theta(2,4) = \max\{\theta(3,7)+0.474, \theta(3,8)+0.491, \theta(3,9)+0.539\} = 4.519,$
$\theta(2,5) = \max\{\theta(3,8)+0.474, \theta(3,9)+0.491, \theta(3,10)+0.539\} = 4.448,$
$\theta(2,6) = \max\{\theta(3,9)+0.474, \theta(3,10)+0.491, \theta(3,11)+0.539\} = 4.371,$
$\theta(2,7) = \max\{\theta(3,10)+0.474, \theta(3,11)+0.491\} = 4.265,$
$\theta(2,8) = \max\{\theta(3,11)+0.474\} = 4.194;$

Phase 10: $\theta(1,0) = \max\{\theta(2,4)+0.553, \theta(2,5)+0.533\} = 5.072,$
$\theta(1,1) = \max\{\theta(2,5)+0.553, \theta(2,6)+0.533\} = 5.001,$
$\theta(1,2) = \max\{\theta(2,6)+0.553, \theta(2,7)+0.533\} = 4.924,$
$\theta(1,3) = \max\{\theta(2,7)+0.553, \theta(2,8)+0.533\} = 4.786,$
$\theta(1,4) = \max\{\theta(2,8)+0.553\} = 4.747;$

The maximum among $\{\theta(1,0), \theta(1,1), \theta(1,2), \theta(1,3), \theta(1,4)\}$, i.e. $\theta(1,0) = 5.072$, was taken as the maximum service value. In this case, the services were executed in the following sequence: $f_{1,2} \rightarrow f_{2,3} \rightarrow f_{3,3} \rightarrow f_{4,1} \rightarrow f_{5,2} \rightarrow f_{6,2} \rightarrow f_{7,2} \rightarrow f_{8,1} \rightarrow f_{9,1} \rightarrow f_{10,2}$. The total execution time was $t_{total} = t_{1,2}+t_{2,3}+t_{3,3}+t_{4,1}+t_{5,2}+t_{6,2}+t_{7,2}+t_{8,1}+t_{9,1}+t_{10,2} = 4+3+3+4+3+3+4+4+4+3 = 35 = DL$.

4.2 Comparative analysis

The execution paths of the STA, MVAL and SRVT are plotted as Fig. 1, providing an intuitive way to compare the performance of the three algorithms.

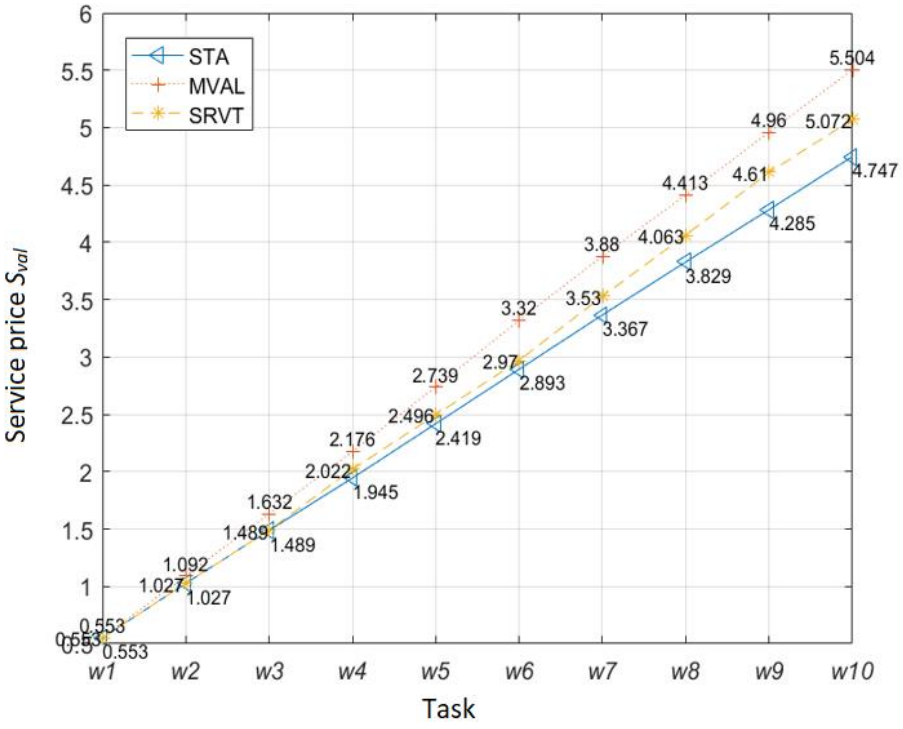

Figure 1: The execution paths of the STA, MVAL and SRVT.

As shown in Fig. 1, the MVAL achieved a much higher total service value than the other two algorithms. However, this algorithm was not adoptable, because its makespan far exceeded the deadline. The SRVT improved the total service value by (5.072–4.747)/4.747 ≈ 6.85 % from the level of the STA. Compared with other single-objective algorithms, the SRVT boasts a certain optimization effect.

5. PERFORMANCE EFFECT OF DEADLINE

The deadline is a preset time for the completion of a workflow. Any algorithm that completes the workflow beyond the deadline should not be adopted, even if it can achieve a high service value. The proposed SRVT computes the execution domain of each task based on the deadline; the longer the deadline, the larger the execution domain. If there is plenty of time, the service with a high value could be selected for execution, pushing up the total service value of the task. Table II shows how each deadline affects the performance. The deadlines were controlled within 46, the time needed for the MVAL to complete all tasks.

Table II: The impact of deadlines on performance.

DL	**SRVT(S_{val})**	**STA(S_{val})**	**Increment from STA**	**MVAL(S_{val})**	**Decrement from MVAL**
45	5.473	4.747	15.29 %	5.504	——
43	5.415	4.747	14.07 %	5.504	——
41	5.350	4.747	12.70 %	5.504	——
39	5.268	4.747	11.35 %	5.504	——
37	5.179	4.747	9.10 %	5.504	——
35	5.072	4.747	6.85 %	5.504	——

As shown in Table II, the SRVT achieved higher service value than the MVAL, when the deadlines were no greater than 46. Thus, the SRVT performance can be improved by properly extending the deadline.

6. CONCLUSIONS

This paper explores the optimal scheduling of automobile manufacturing services with deadlines, and puts forward a deadline-constrained service value-time optimization algorithm (SRVT). Firstly, the workflow diagram was plotted, and the execution domain of each task was calculated, according to the deadline and the earliest and latest start times of the task. Next, the SRVT was adopted to reversely derive the service with the maximum value at each time point, and adds it to the set of candidate solutions in the next iterations. Finally, an actual workflow was selected to analyse the performance of our algorithm. The results show that the SRVT performance can be improved feasibly, when the deadline is constant.

REFERENCES

[1] Luo, Z.-Y.; Wang, P.; You, B.; Zhu, S.-X. (2016). Serial reduction optimization research of complex product workflow's accuracy under the time constraint, *Advances in Mechanical Engineering*, Vol. 8, No. 10, Paper 1687814016672119, 9 pages, doi:10.1177/1687814016672119

[2] Deldari, A.; Naghibzadeh, M.; Abrishami, S. (2017). CCA: a deadline-constrained workflow scheduling algorithm for multicore resources on the cloud, *The Journal of Supercomputing*, Vol. 73, No. 2, 756-781, doi:10.1007/s11227-016-1789-5

[3] Xie, Z. L.; Yin, H. K. (2018). Selection of optimal cloud services based on quality of service ontology, *Ingénierie des Systèmes d'Information*, Vol. 23, No. 6, 127-141, doi:10.3166/ISI.23.6.127-141

[4] Ajeena Beegom, A. S.; Rajasree, M. S. (2019). Non-dominated sorting based PSO algorithm for workflow task scheduling in cloud computing systems, *Journal of Intelligent & Fuzzy Systems*, Vol. 37, No. 5, 6801-6813, doi:10.3233/JIFS-190355

[5] Ajmi, F.; Zgaya, H.; Othman, S. B.; Hammadi, S. (2019). Agent-based dynamic optimization for managing the workflow of the patient's pathway, *Simulation Modelling Practice and Theory*, Vol. 96, Paper 101935, 21 pages, doi:10.1016/j.simpat.2019.101935

[6] Xu, W.; Yin, Y. (2018). Functional objectives decision-making of discrete manufacturing system based on integrated ant colony optimization and particle swarm optimization approach, *Advances in Production Engineering & Management*, Vol. 13, No. 4, 389-404, doi:10.14743/apem2018.4.298

[7] Xu, L. Z.; Xie, Q. S.; Yuan, Q. N.; Huang, H. S. (2019). An intelligent optimization algorithm for blocking flow-shop scheduling based on differential evolution, *International Journal of Simulation Modelling*, Vol. 18, No. 4, 678-688, doi:10.2507/IJSIMM18(4)CO16

[8] Xu, H.; Yang, B.; Qi, W.; Ahene, E. (2016). A multi-objective optimization approach to workflow scheduling in clouds considering fault recovery, *KSII Transactions on Internet and Information Systems*, Vol. 10, No. 3, 976-995, doi:10.3837/tiis.2016.03.002

[9] Ahmad, S. G.; Liew, C. S.; Munir, E. U.; Ang, T. F.; Khan, S. U. (2016). A hybrid genetic algorithm for optimization of scheduling workflow applications in heterogeneous computing systems, *Journal of Parallel and Distributed Computing*, Vol. 87, 80-90, doi:10.1016/j.jpdc.2015.10.001

[10] Alkhanak, E. N.; Lee, S. P.; Rezaei, R.; Parizi, R. M. (2016). Cost optimization approaches for scientific workflow scheduling in cloud and grid computing: A review, classifications, and open issues, *Journal of Systems and Software*, Vol. 113, 1-26, doi:10.1016/j.jss.2015.11.023

[11] Hsieh, F.-S. (2017). A hybrid and scalable multi-agent approach for patient scheduling based on Petri net models, *Applied Intelligence*, Vol. 47, No. 4, 1068-1086, doi:10.1007/s10489-017-0935-y

[12] Al-Kiswany, S.; Costa, L. B.; Yang, H.; Vairavanathan, E.; Ripeanu, M. (2017). A cross-layer optimized storage system for workflow applications, *Future Generation Computer Systems*, Vol. 75, 423-437, doi:10.1016/j.future.2017.02.038

[13] Shishido, H. Y.; Estrella, J. C.; Toledo, C. F. M.; Arantes, M. S. (2018). Genetic-based algorithms applied to a workflow scheduling algorithm with security and deadline constraints in clouds, *Computers & Electrical Engineering*, Vol. 69, 378-394, doi:10.1016/j.compeleceng.2017.12.004

[14] Xu, W.; Yin, Y. (2018). Functional objectives decision-making of discrete manufacturing system based on integrated ant colony optimization and particle swarm optimization approach, *Advances in Production Engineering & Management*, Vol. 13, No. 4, 389-404, doi:10.14743/apem2018.4.298

[15] Özdemir, H.; Sever, R.; Polat, Ö. (2019). GA-based optimization of SURF algorithm and realization based on Vivado-HLS, *Traitement du Signal*, Vol. 36, No. 5, 377-382, doi:10.18280/ts.360501

[16] Verma, A.; Kaushal, S. (2017). A hybrid multi-objective particle swarm optimization for scientific workflow scheduling, *Parallel Computing*, Vol. 62, 1-19, doi:10.1016/j.parco.2017.01.002

[17] Ambursa, F. U.; Latip, R.; Abdullah, A.; Subramaniam, S. (2017). A particle swarm optimization and min-max-based workflow scheduling algorithm with QoS satisfaction for service-oriented grids, *The Journal of Supercomputing*, Vol. 73, No. 5, 2018-2051, doi:10.1007/s11227-016-1901-x

[18] Prathibha, S.; Latha, B.; Sumathi, G. (2017). An improved multi-objective optimization for workflow scheduling in cloud platform, *Journal of Internet Technology*, Vol. 18, No. 3, 589-599, doi:10.6138/JIT.2017.18.3.20161101

[19] Abdi, S.; PourKarimi, L.; Ahmadi, M.; Zargari, F. (2018). Cost minimization for bag-of-tasks workflows in a federation of clouds, *The Journal of Supercomputing*, Vol. 74, No. 6, 2801-2822, doi:10.1007/s11227-018-2322-9

[20] Zhang, M.; Li, H.; Liu, L.; Buyya, R. (2018). An adaptive multi-objective evolutionary algorithm for constrained workflow scheduling in Clouds, *Distributed and Parallel Databases*, Vol. 32, No. 2, 339-368, doi:10.1007/s10619-017-7215-z

[21] Almi'ani, K.; Lee, Y. C.; Mans, B. (2018). On efficient resource use for scientific workflows in clouds, *Computer Networks*, Vol. 146, 232-242, doi:10.1016/j.comnet.2018.10.003

[22] Ghafouri, R.; Movaghar, A.; Mohsenzadeh, M. (2018). Time-cost efficient scheduling algorithms for executing workflow in infrastructure as a service clouds, *Wireless Personal Communications*, Vol. 103, No. 3, 2035-2070, doi:10.1007/s11277-018-5895-y

[23] Choudhary, A.; Gupta, I.; Singh, V.; Jana, P. K. (2018). A GSA based hybrid algorithm for bi-objective workflow scheduling in cloud computing, *Future Generation Computer Systems*, Vol. 83, 14-26, doi:10.1016/j.future.2018.01.005

[24] Naidu, P. S.; Bhagat, B. (2018). Secure workflow scheduling in cloud environment using modified particle swarm optimization with scout adaptation, *International Journal of Modeling, Simulation, and Scientific Computing*, Vol. 9, No. 1, Paper 1750064, 10 pages, doi:10.1142/S1793962317500647

[25] Luo, Z.-Y.; Zhu, Z.-H.; You, B.; Liu, J.-H. (2018). Virtual workflow constrained time-accuracy optimization algorithm scheduling by iterative reduction, *Journal of Electronics and Information Technology*, Vol. 40, No. 8, 2013-2019, doi:10.11999/JEIT171038

[26] Luo, Z.-Y.; Zhu, Z.-H.; You, B.; Liu, J.-H. (2019). A time-quality optimization algorithm of three-layer decision model based on virtual iterative workflow, *Acta Electronica Sinica*, Vol. 47, No. 1, 245-251, doi:10.3969/j.issn.0372-2112.2019.01.033

[27] Qureshi, B. (2019). Profile-based power-aware workflow scheduling framework for energy-efficient data centers, *Future Generation Computer Systems*, Vol. 94, 453-467, doi:10.1016/j.future.2018.11.010

[28] Kalra, M.; Singh, S. (2019). An intelligent water drops-based approach for workflow scheduling with balanced resource utilisation in cloud computing, *International Journal of Grid and Utility Computing*, Vol. 10, No. 5, 528-544, doi:10.1504/IJGUC.2019.101995

[29] Ismayilov, G.; Topcuoglu, H. R. (2020). Neural network based multi-objective evolutionary algorithm for dynamic workflow scheduling in cloud computing, *Future Generation Computer Systems*, Vol. 102, 307-322, doi:10.1016/j.future.2019.08.012