ISSN XXXX XXXX © 2018 IJESC

Research Article Volume 8 Issue No.10

# Confuse Workflow Preparation with Deadlines and Instance Period Accessibility

R.Jayasri [1], K.Sathya [2]
M.Phil., Research Scholar[1], Assistant Professor[2]
Department of Computer Science
Padmavani Arts and Science College for Women, Salem, India

**Abstract:**
Permitting cloud capacities in darken computing is based on the postulation that they are unrestricted and can be used at any time. However, obtainable service capacities modify with workload and cannot convince users' desires at any time from the cloud provider's standpoint because cloud air force can be collective by several tasks. Cloud service provider affords obtainable occasion slots for new user's desires based on obtainable capacities. We consider workflow preparation with time limit and moment slot accessibility in cloud computing. A dynamic heuristic construction is obtainable for the problem under study which mainly consists of primary explanation invention, perfection, and perturbation. Three primary solution assembly strategies, two voracious- and fair-based upgrading strategies and a perturbation strategy are anticipated. Different strategies in the three phases result in numerous heuristics. Tentative outcome show that different primary result and perfection strategies have different special effects on clarification behavior.

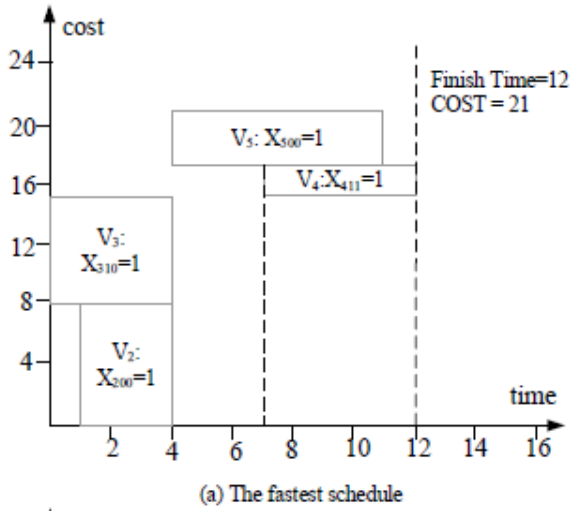
**Index Terms:** Workflow, preparation, Time slots, darken compute.

## I. INTRODUCTION

The fundamental air force is borrowed by users for their difficult applications with diverse reserve necessities which are generally modeled as workflows. Improved air force involves superior outlay. Navy are inspired based on Service elevation Agreements, which classify arguments of superiority of repair in conditions of the pay-per-use guidelines. Although present are loads of arguments or constraints occupied in sensible workflow preparation settings, time limit and instance period are two critical ones in confuse computing, a latest advertise leaning commerce reproduction, which offers soaring superiority and short price in sequence air force. Essential to believe both of the constraints mutually since: (i) Last time of the schedule applications wants to be meet. (ii) Unconditional instance schedule is critical for supply deployment from the standpoint of examination service. (iii) exploitation of moment slots in held in reserve free time should be enhanced to circumvent renting new property. In this paper, we judge the workflow preparation difficulty with deadlines and instance slot accessibility in cloud computing.

The considered WSDT problem is similar to the Discrete Time/Cost Trade-off Problem (DTCTP) [3] to some extent. We can change presented algorithms for the final to the solution below learning with fewer than 200 behavior and no further than 20 applicant air force in the examiner team spending thousands of seconds .The number of activities is usually far more than 200 in practical workflow applications which makes the modified versions are not suitable for the problem under study. Normally, longer implementation instance implies cheaper price in darken computing for the DTCTP.

The outlay is in the converse fraction to the implementation point shows the fastest schedule and a non-fastest one where xijk=1means the kth available time slot of Mj i is selected for vi (Mj I is the jth available service list of vi). In the fastest schedule _1=fx100=1, x200=1, x310=1, x411=1, x510=1, x600=1g, each activity vi chooses the service to finish as early as possible.



(a) The fastest schedule

The non-fastest schedule is _2=fx100=1, x200=1, x300=1, x401=1, x510=1, x600=1g with the finish time f6=13 and the total cost 27. The total cost of the fastest schedule is less than that of the non-fastest one.

## II. RELATED WORK

Time limit unnatural workflow arrangement is one of the majority accepted preparation problems in confuse computing. Suitable navy are selected for all the activities in the complicated workflow to exploit objectives. Minimizing time

holdup, expenses and time-cost trade-off are ordinary objectives. repair assortment for workflow scheduling bearing in mind both instant and charge was modeled as the separate Time/Cost Trade-off Problem (DTCTP). Every workflow or scheme movement can be satisfied by extra than one repair, which consists of a examination group. normally, workflows are represented by undeviating Acyclic Graphes. Deadlines are common constraints for workflow arrangement. The DTCTP with deadline or financial plan constraint constraints was called DTCTP-D. It follows that the DTCTP-D is NP-hard as good.

The exact algorithms are usually heavily time consuming for complex workflow instances when the number of activities is more than 200. Consideration on deadline partition, numerous heuristics was urbanized for the DTCTP-D in circulated computing environments. Yu et al. [15] upcoming the workflow Markov Decision Process (DMDP) method for effectiveness workflow preparation. DMDP partitioned the DAG into sub graphs and assigned each partition a sub-deadline according to the minimum execution time of activities and the whole workflow deadline. Yuan et al. [16] obtainable the target Early Tree (DET) scheme where a solitary serious path was constructed to distinguish dangerous and non-critical behavior. Dangerous behavior was scheduled primary and the non-critical ones be planned according to their priorities. The priorities were strong-minded by floats or slacks. Abraham et al.

The entire actions were partition into diverse incomplete dangerous paths. Every incomplete dangerous path was planned according to priorities of its behavior. They demonstrated that PCP outperforms DMDP and DET. Liu et al. Proposed the Path impartial based outlay Optimization algorithm which accustomed the duration of every one path in a workflow. A deadline was position for every task. The outstanding occasion was circulated proportionally to responsibilities according to the quantity of responsibilities at every level the depth of a pathway join from the resource or drop node using the foundation level approach. PBCO confirmed better presentation than DET and DBL. Though it seems that there is no existing work on the DTCTP-D, a few meta heuristics were proposed for workflow scheduling in Grid computing or only for the DTCTP. Based on the proposed Critical Path based Iterative heuristic (CPI) in, they considered shareable service provisioning for workflows in public clouds.

## III. METHODOLOGY

In the DTCTP, each service can be used at any time, namely the available time slot for each service is [0;+1. Both of time 0, the cost 0, and the available time slot [0;+1). Activity v2 has two available services M0 2 and M1 2 . The execution time and cost of M0 2 are 3 and 8 and those of M1 2 are 4 and 5 he DTCTP can be seen as a special case of the WSDT when the available time slot is [0;+1) for each service. The DTCTP was proven to be NP-hard [13]. It is natural that the WSDT is NP-hardV =fv1;v2;:::;vng is the set of activities and E=f(vi;vj)jvi;vj2V g is the set of arcs (precedences between activities). v1 and vn are two dummy activities. Each arc (vi;vj)2E (i6=j) indicates that vj cannot start until vi finishes. The service pool Mi=fM0 i ;M1 i ;:::;MNc i 1 i g of vi2V is composed of Nc i candidate services provided by different CSPs. Mj i is the jth service of vi. The available time slot list of Mj i with length Ns ij is represented as Sij=(Sij0;Sij1;:::;Sij(Ns ij□1)). The kth available time slot Sijk in Mji is denoted as [Bijk;Fijk), where Bijk and Fijk represent the start and end times of Sijk respectively. Mj I with the execution time eij and cost cij is represented by (Sij;eij ;cij). fi denotes the finish time of vi. D is the given deadline. Since the problem under study is similar to those in the mathematical model is constructed using the mixed integer programming. The following binary decision variables are needed:

Only one service is chosen for each activity from its service pool. Only one available time slot is selected from the corresponding service time slot list. Each activity cannot be interrupted during execution. The cost is in inverse proportion to the execution time. The execution time includes computing and data transmission times. No service is shared by any two activities, i.e., services for different activities are independent. Service pools of all the activities in workflow instances are constant.

(1). exactly one service is assigned to each activity according to constraint (2). The actual start time of each activity cannot be earlier than the start time of the selected time slot according to constraint (3). Constraint (4) guarantees that the actual finish time of each activity cannot later than the finish time of the selected time slot. Constraint (5) controls precedence constraints. The deadline is met according to constraint (6). Constraint (7) defines binary decision variables.

For each remaining activity vi, the two parameters are calculated by the following way with time complexity O(Ns ij_Nc i +jPij).

$$E_{st}(i)=\min_{j=0,\ldots,\mathcal{N}_i^c-1}\left\{B_{ij}^A(\max_{p\in\mathcal{P}_i}\{E_{ft}(p)\})\right\}$$

$$E_{ft}(i)=\min_{j=0,\ldots,\mathcal{N}_i^c-1}\left\{B_{ij}^A(\max_{p\in\mathcal{P}_i}\{E_{ft}(p)\})+e_{ij}\right\}$$

**ILAH consists of four components:**
Time Slot Filtering, Initial Solution Construction, Solution Improvement and Perturbation. ILAH starts from an initial solution _. Improving and perturbing operations are performed on _ iteratively until the termination criterion is satisfied. The termination criterion is set as _, the number of consecutive iterations without improvement. Let C(_) be the total cost of _. The high level procedure of ILAH is described in Algorithm 1.

```
begin
    Time Slot Filtering;
    Generate the initial solution π by an initial solution
    construction strategy;
    π_best ← π, C(π_best) ← C(π);
    while (termination criterion not met) do
        π ← Improve(π);
        if (C(π_best) > C(π)) then
            π_best ← π, C(π_best) ← C(π);
        Perturbation(π);
    return π_best.
```

**Time Slot Filtering**

Available time slots might not be available for an activity vi even before the service assignment. For example, Eft(n)>D in the fastest schedule, or the duration of a time slot is less than the execution time of the activity, or the start or finish time is beyond the earliest start or the latest finish time of the activity. By filtering out all impossible time slots, remaining time slots are eligible for activities of the instance, which make workflow scheduling much more efficient. The Time Slot Filtering procedure is given in Algorithm 2.

**begin**
  **for** *(each* $v_i \in V$*)* **do**
    Calculate $E_{st}(i)$, $E_{ft}(i)$, $L_{ft}(i)$, $L_{st}(i)$ using equations (8), (9), (10), (11);
    **if** $(E_{ft}(n) > D)$ **then**
      **return** *NULL*;
      /* infeasible problem */
  **for** *(each* $v_i \in V$*)* **do**
    **for** *(each service* $M_i^j \in \mathbb{M}_i$*)* **do**
      **for** $k = 0$ **to** $\mathcal{N}_{ij}^s - 1$ **do**
        **if** $F_{ijk} - B_{i,j,k} < e_{ik}$ *or* $B_{i,j,k} > D$ *or* $B_{ijk} > L_{ft}(i)$ *or* $F_{ijk} < E_{st}(i)$ **then**
          Remove $s_{ijk}$ from $\mathbb{S}_{ij}$;
      **if** $(\mathcal{N}_{ij}^s = 0)$ **then**
        Remove $M_i^j$ from $\mathbb{M}_i$;
    **for** *(each* $v_i \in V$*)* **do**
      Generate the service pool $\mathbb{M}_i$ by sorting all candidate services in non-increasing order of costs;
  **return** $\{\mathbb{M}_i\}$.

The time complexity of Steps 2_5 is O(n_(Ns ij_Nc i + jPij)), that of Steps 6_12 is O(n_Ns ij_Nc i ), and that of Steps 13_14 is O(n_Nc i _lnNc i ). Therefore, the time complexity of the Time Slot Filtering procedure is O(n_((Nsij+lnNc i )_ Nc i +jPij)).

**Initial solution construction strategies**

Different service assignments to activities result in different initial solutions. In this paper, we present three priority rules, based on which three initial solution construction strategies are developed. The Minimum Average Cost First (MACF) method focuses on the cost of an activity and those of its immediate successors, i.e., costs among activities. The Maximum Cost Ascending Ratio First (MCARF) strategy considers the two services with the lowest costs for of same activity.

**Minimum Average Cost First (MACF)**

MACF generates initial solutions according to the minimum average cost on an activity and its immediate successors. A smaller average cost implies a higher priority of activity choosing a service. After an available service Mj i is selected for vi, fij=BA ij(Est(i))+eij and the earliest start times of all immediate successors of vi are updated by Est(q)= maxffij;Est(q)g (8q2Qi). Let Msj q be the cheapest available service for activity vq when Mj i is selected for vi. W(i;j) denotes the average cost on activity vi and all of its successors and W_ i represents the Minimum Average Cost W(i;j) among all available services of vi. W(i;j) and W_ i are calculated by

$$\overline{W}(i,j) = \left(c_{ij} + \sum_{q \in \mathcal{Q}_i} c_{qs_j}\right) / (|\mathcal{Q}_i| + 1)$$

$$\overline{W}_i^* = \min_{j=0,\ldots \mathcal{N}_i^c - 1} \left\{\overline{W}(i,j)\right\}$$

Let S and U be sets of scheduled and unscheduled activities, which are initialized as S=f1;ng and U=f2;:::;n□1g respectively. Services M0 1 and M0 n are assigned to the two dummy activities v1 and vn respectively. Est(1)=0 and Eft(1)=0.

## IV. EXPERIMENTAL SETUP

We calibrate components and parameters first, based on which three heuristics are compared. All involved algorithms are coded in Java and run on Intel(R) Core(TM) i7-4770 CPU @3.40GHz with 8GB RAM on Windows Server 2008 R2 standard. The termination criterion is set as the maximum computation time n2 t (t2f5;10;20g) milliseconds.

- *parameter and Component Calibration* condition parameter t=5, i.e., the computation time is limited to n2 5 milliseconds for all combinations. WSDT calibration instances are randomly generated. Seven instance factors result in different workflows. These are: N, M, OS, CF, CP, Loadnum and DF. N2 f200;400;600;800;1000g is the number of activities in a workflow.
- *The deadline* D=Dmin+ (TH□Dmin)_DF. Therefore, there are 5_3_3_3_3_3_3 = 3645 instance combinations. One calibration instance is randomly generated for each combination, i.e., 3645 instances are test to calibrate the components and parameters.

Though there are three initial solution generation strategies (EFTF, MACF and MCARF), there is a parameter _ in MACF. In this paper, we test all values of _2f0:25;0:5;0:75;1g, i.e., four MACFs are tested. By replacing Equation (14) of MCARF with RI i=yi( cij cik )2+(1□yi) cij (n□2) Tmin , a new rule MCARF0 is obtained. In addition, the RANDOM strategy is involved, which generates initial solutions randomly. In other words, we calibrate eight variants of the initial solution construction component. Three variants of the improvement component are evaluated: GIH, FIH and NONE (no improvement strategy). _ takes values from f0, 0.2, 0.4, 0.6, 0.8, 1g, i.e., 6 candidates for _ are considered. There are two operations for time slots: filtering and non-filtering.

A number of hypotheses have to be ideally met by the experimental data. The main three hypotheses (in order of importance) are the independence of the residuals, homoscedasticity or homogeneity of the factor's level variance and normality in the residuals of the model. Apart from a slight non-normality in the residuals, all the hypotheses are easily accepted. The response variable in the experiments is RPD (Relative Percentage Deviation) for each algorithm in every instance. Let Vi(H) be the solution of instance i obtained by algorithm H and V _ i be the optimum solution for i. The RPD is defined as

$$RPD = \frac{V_i(H) - V_i^*}{V_i^*} \times 100\%$$

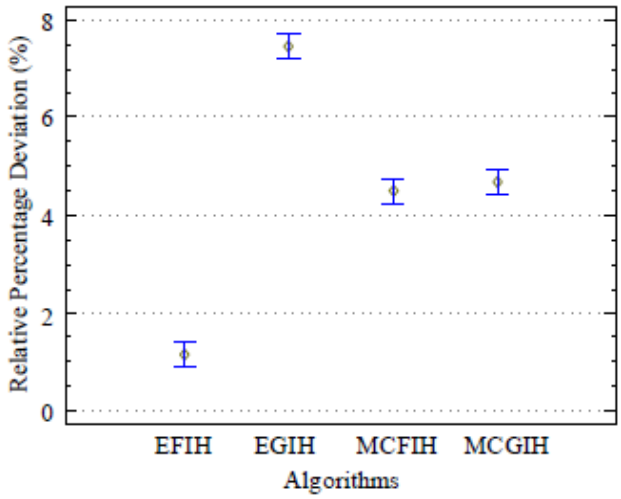
The NONE has the largest RPD which implies that the FIH and the GIH are effective improvement strategies. We choose both FIH and GIH as improvement strategies.

Any pair of the four types of initial solution generation strategies (EFTF, MACF, MCARF and RANDOM) are significant. RPDs of the MACF with different parameters have no statistically significant differences. MCARF and MCARF0 have the lowest RPD. RPDs of MCARF and MCARF0 are slightly less than those of MACFs but much smaller than those of the EFTF and the RANDOM, which indicates that MCARF and MCARF0 are effective for the problem under study. However, it is strange that the EFTF has the largest RPD, even worse than RANDOM. The main reason lies in that the EFTF chooses service time slots with the earliest finish times which always results in poor initial solutions. Both the MCARF and the EFTF are selected as initial solution generation strategies for the proposed algorithm in the following comparing experiments.

**Algorithm Comparison**

For the problem studied in this paper. Except the Time Slot Filtering component of ILAH (the statistical difference is not significant between ILAHs with and without the Time Slot Filtering component in the above calibration), there are two variants of the Initial Solution Construction (MCARF and EFTF), two candidates of the Solution Improvement (FIH and GIH) and one Perturbation operator (_=0.4). Therefore, we compare four ILAH-based algorithms: MCFIH, MCGIH, EFIH and EGIH. The MCARF are selected by the MCFIH and the MCGIH while the EFTF are chosen by the EFIH and the EGIH to generate initial solutions.

The termination criterion is set as the maximum computation time n2 20 (i.e., t=20) milliseconds. To avoid bias in the result, we do not take results from the previous calibration experiments but rather we run all algorithms again. For each of the 3645 instance combinations, 10 instances are randomly generated, i.e., totally 3645_10_4=145800 tests are conducted for the four algorithm comparisons. For each algorithm, the average RPD on 10 instances of every combination is used to measure the algorithm's effectiveness.




To demonstrate the performance of the compared algorithms in a sound and statistical way, we first analyze the experimental results using the same ANOVA tool as before. The means plot of compared algorithms and 95.0% Tukey HSD intervals. The EFIH has the smallest RPD and the EGIH has the largest, the difference is statistically significant. The RPD difference between the MCFIH and the MCGIH is not statistically significant.

**Perturbation**

The ILAH is likely to get trapped into local optima after some iterations. To enhance the diversification of the search process, a Perturbation method is introduced. Starting from a solution _, bn_c (0<_<1) activities are randomly selected and their current service assignment are changed by more expensive services, which can result in bigger adjustment intervals. The new solution is then improved by the FIH or the GIH with the hope of finding a better solution than before. Substituting the current service Mj i of vi with a more expensive available service, Mj0 i might lead to changes in the start and finish times of predecessors and successors. Let `p and `s be the increased slack or float time of immediate predecessors and successors of vi respectively with the substitution. `p=stij0□stij , where stij0 is the start time of vi with service Mj0 i and stij is the start time of vi with the current service Mj i . `s=ftij0□ftij , where ftij0 is the finish time of vi with service Mj0 i and ftij is the finish time of vi with the current service Mj i . If the service substitution delays the start time of vi (`p>0), more float time will be available for all of its immediate predecessors. Otherwise, no more float (`p=0) is available for its immediate predecessors. If the finish time ftij0 with service Mj0 i becomes smaller than that ftij with the current service Mj i , i.e., `s>0, more float time will be available for all of its immediate successors. Otherwise, no more float (`s=0) is available for its immediate successors.

The cost increase ratio per float unit RF ijj0 of substituting Mj I with Mj0 i for vi is defined by

$$R_{ijj'}^{F}=\begin{cases}+\infty & \ell_p=0 \text{ and } \ell_s=0 \text{ or no such } M_i^{j'} \\ \frac{c_{ij'}-c_{ij}}{|P_i|\times\ell_p+|Q_i|\times\ell_s} & \text{Otherwise}\end{cases}$$

A smaller RF ij=minfRF ijj0g means a less cost increasing ratio per float unit and vi has a higher priority to replace its current service with a more expensive one. To avoid repeating substitution on the same activity with the same available service, we perform replacement on the activity with the mth smallest RF ij if there is no improvement in consecutive m iterations (m).

The MCFIH and the MCGIH are significantly different from those of the EFIH and EGIH. It is surprising that the EFIH outperforms the other three algorithms though the performance of EFTF is even worse than that of RANDOM to generate initial solutions. This implies that the FIH is much more effective than the GIH for improving poor solutions. To further demonstrate the performance of the four compared algorithms, ARPDs (Average Relative Percentage Deviation) on all instance combinations are shown in Table 2. From Table 2, we observe that the EFIH has the smallest ARPDs on all parameter combinations.

The EGIH has the largest ARPD on each parameter combinations. The MCFIH and the MCGIH have similar ARPDs on all the instances. The average ARPDs of the four algorithms supports the results shown in Figure 8. The average

ARPD of EFIH is only 1.17% which is much better than that of EGIH with 7.44%. The average ARPDs of MCFIH and MCGIH are 4.52% and 4.68%, which are worse than that of EFIH.

**Influences of instance parameters on the compared algorithms' performance:**

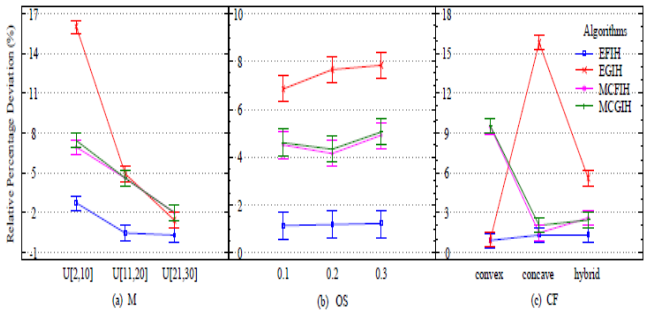
ARPDs of the compared algorithms are different when an instance parameter takes different values. We analyze influences of six instance parameters (M, OS, CF, CP, Loadnum and DF) on the compared algorithms' performance using the ANOVA tool as before. Interactions between each parameter and the compared algorithms with 95.0% Tukey HSD intervals are depicted in Figure 9All algorithms obtain similar RPDs in the U[21,30] case. The reason lies in that a bigger service pool gives more service candidates which leads to smaller total costs.

*ARPDS COMPARED ALGORITH*

| Parameter | Value | EFIH | EGIH | MCFIH | MCGIH |
|---|---|---|---|---|---|
| DF | 0.2 | **1.24** | 5.06 | 2.13 | 2.26 |
| | 0.4 | **1.17** | 7.28 | 4.55 | 4.68 |
| | 0.6 | **1.09** | 9.99 | 6.87 | 7.08 |
| N | 200 | **1.68** | 6.45 | 4.15 | 4.35 |
| | 400 | **1.43** | 7.01 | 4.28 | 4.37 |
| | 600 | **1.08** | 7.55 | 4.37 | 4.59 |
| | 800 | **0.96** | 8.07 | 5.02 | 5.21 |
| | 1000 | **0.70** | 8.14 | 4.75 | 4.85 |
| OS | 0.1 | **1.11** | 6.86 | 4.49 | 4.62 |
| | 0.2 | **1.19** | 7.64 | 4.17 | 4.34 |
| | 0.3 | **1.20** | 7.83 | 4.90 | 5.06 |
| LoadNum | 0 | **1.14** | 5.91 | 4.85 | 5.12 |
| | 1 | **0.98** | 7.89 | 4.85 | 4.93 |
| | 2 | **1.38** | 8.53 | 3.84 | 3.98 |
| CF | convex | **0.87** | 0.93 | 9.47 | 9.54 |
| | concave | **1.30** | 15.82 | 1.49 | 2.06 |
| | hybrid | **1.32** | 5.59 | 2.59 | 2.42 |
| M | U[2,10] | **2.71** | 15.99 | 6.92 | 7.43 |
| | U[11,20] | **0.47** | 4.91 | 4.60 | 4.60 |
| | U[21,30] | **0.31** | 1.43 | 2.03 | 2.00 |
| CP | 1 | **1.37** | 6.43 | 2.66 | 2.83 |
| | 2 | **1.07** | 7.78 | 4.34 | 4.55 |
| | 3 | **1.06** | 8.13 | 6.55 | 6.64 |
| Average | | **1.17** | 7.44 | 4.52 | 4.68 |

RPD differences are statistically significant for each algorithm in the three cases except that the RPD difference of the EFIH is not statistically significant in the U[11,20] and U[21,30] cases. OS exerts a little influence on the compared algorithms. RPD differences are not statistically significant for each algorithm in all cases. The cost function CF has a great influence on the EGIH. However, RPD differences are not statistically significant for the EFIH with any cost functions. The MCFIH and the MCGIH obtain much poorer performance when we adopt the convex cost function. RPD differences of EFIH and EGIH for different CP values are not statistically significant while those of MCFIH and MCGIH are statistically significant. With the increase of CP, the RPD of EFIH decreases but those of the other three increase. Loadnum gives impact on the performance of the EGIH. RPD differences of the EGIH are statistically significant for different Loadnum values.

RPD differences of the other three algorithms are not statistically significant for different Loadnum values, i.e., Loadnum exerts a little influence on EFIH, MCFIH and MCGIH. RPD differences of the EFIH are not statistically significant whereas those of the other three are statistically significant for different DF values. In addition, RPDs increase with the increase of DF for all the compared algorithms except the EFIH.




The proposed EFIH is the most robust algorithm among the proposals for all instance parameters (M, OS, CF, CP, Loadnum and DF). In other words, EFIH is suitable for scheduling deadline constrained realistic workflow applications (e.g., Montage, CyberShake, Epigenomics, LIGO, and SIPHT) on rented resources (e.g., EC2 virtual machines from Amazon). In addition, EFIH is applicable to workflow scheduling in cloud manufacturing where virtualized manufacturing resources are rented for workflow applications with deadlines.

## V.CONCLUSION

We proved that the WSDT had different properties from the DTCTP. The ILAH (iterated local adjusting heuristic) framework was proposed for the NP-hard WSDT. Two improvement strategies, the FIH and the GIH, were introduced which had similar influences on the solution improvement. the impact of different pricing interval lengths on workflow scheduling is worth studying. Instance intensive workflows are also a desirable area of study for future work. We recommend a new algorithm named the SaaS Cloud fractional dangerous pathway (SC-PCP) for workflow preparation in SaaS Clouds, which minimizes the total implementation outlay while assembly a user-defined target. Estimate our algorithm by simulating it with imitation work flows that are based on actual technical workflows with special structures and sizes. We chart to broaden our algorithm to sustain other Cloud computing models, such as IaaS and other pricing models.

## VI. REFERENCES


[1] J. Yu, R. Buyya, and C. K. Tham, "Cost-based scheduling of scientific workflow applications on utility grids," in e-Science and Grid Computing, 2005. First International Conference on. IEEE, 2005, pp. 8–pp.

[2] Bunya, R., Yeo , C.S., Venugopal, S., Broberg, J. and Brandic, I. ''Cloud computing and emerging IT platforms: vision, hype and reality for delivering computing as the 5th utility'', Future Gener. Comput. Syst., 25(6), pp. 599–616 (2009)..

[3] Hoffa, C., Mehta, G., Freeman, T., Deelman, E., Keahey, K., Berriman, B. and Good, J. ''on the use of cloud computing for scientific workflows'', Fourth IEEE Int'l Conference on eScience (e-Science 2008), Indiana, USA, pp.640–645 (2008).

[4] Weinhardt, C., Anandasivam, A., Blau, B. and Stoesser, J. ‘‘Business models in the service world.

[5] Deelman, E. ‘‘Grids and Clouds: making workflow applications work in heterogeneous distributed environments’’, Int. J. High Perform. Comput. Appl., 24(3), pp. 284–298 (2010).

[6] N. Paladi, C. Gehrmann, and F. Morenius, “Domain-Based Storage Protection (DBSP) in Public Infrastructure Clouds,” in Secure IT Systems, pp. 279–296, Springer, 2013.

[7] F. Zhang, J. Chen, H. Chen, and B. Zang, “Cloudvisor: retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization,” in Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, pp. 203–216, ACM, 2011.

[8] Z. Cai, X. Li, and L. Chen, “Dynamic programming for services scheduling with start time constraints in distributed collaborative manufacturing systems,” in Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on, 2012, pp. 803–808.

[9] N. Santos, K. P. Gummadi, and R. Rodrigues, “Towards trusted cloud computing,” in Proceedings of the 2009 Conference on Hot Topics in Cloud Computing, HotCloud’09, (Berkeley, CA, USA), USENIX Association, 2009.

[10] “X. Zou, “A hierarchical attribute-based encryption scheme,” Wuhan Univ. J. Natural Sci., vol. 18, no. 3, pp. 259–264, Jun. 2013.

[11] K. Emura, A. Miyaji, A. Nomura, K. Omote, and M. Soshi, “A ciphertext-policy attribute-based encryption scheme with constant ciphertext length,” in Proc. 5th Int. Conf. Inf. Secur. Pract. Exper., vol. 5451. Apr. 2009, pp. 13–23.

[12] V. Goyal, A. Jain, O. Pandey, and A. Sahai, “Bounded ciphertext policy attribute based encryption,” in Proc. 35th Int. Colloq. Automata, Lang. Program., vol. 5126. Jul. 2008, pp. 579–591.

[13] M. Green, S. Hohenberger, and B. Waters, “Outsourcing the decryption of ABE ciphertexts,” in Proc. 20th USENIX Conf. Secur., Aug. 2011, pp. 1–16.