Vietnam Journal of Science and Technology ***56*** *(2) (2018) 246-256*

DOI: 10.15625/2525-2518/56/2/10672

A BRANCH AND BOUND ALGORITHM FOR WORKFLOW SCHEDULING

Phan Thanh Toan[1, *] , Nguyen The Loc[2]

[1]*Faculty of Technology Education, HNUE, 136 Xuan Thuy, Ha Noi, Viet Nam*

[2]*Faculty of Information Technology, HNUE, 136 Xuan Thuy, Ha Noi, Viet Nam*

[*]Email: *pttoan@hnue.edu.vn*

Received: 13 September 2017; Accepted for publication: 11 January 2018

Abstract. Nowadays, people are connected to the Internet and use different Cloud solutions to store, process and deliver data. The Cloud consists of a collection of virtual servers that promise to provision on-demand computational and storage resources when needed. Workflow data is becoming an ubiquitous term in both science and technology and there is a strong need for new tools and techniques to process and analyze large-scale complex datasets that are growing exponentially. Scientific workflow is a sequence of connected tasks with large data transfer from parent task to children tasks. Workflow scheduling is the activity of assigning tasks to execution on servers and satisfying resource constraints and this is an NP-hard problem. In this paper, we propose a scheduling algorithm for workflow data that is derived from the Branch and Bound Algorithm.

Keywords: workflow scheduling, Branch and Bound Algorithm, cloud computing.

Classification numbers: 4.7.1; 4.7.4

1. INTRODUCTION

With the development of the network technology, Cloud Computing used to solve larger scale complex problems becomes a focus technology. Scheduling for big data workflow is a challenging problem in Cloud environment. Data scientists develop workflows by modeling their complex scientific applications as a set of data processing tasks with a set of data dependencies between the tasks and there are many scientific applications that use workflow data such as Montage [1], CyberShake [2], Epigenomics [3], LIGO [4, 5]. The goal of these applications is to minimize the total cost for executing the workflow.

The workflow scheduling problem in a cloud environment is essentially mapping of tasks in the workflow to cloud servers that satisfy the order of the tasks in the workflow and the total costs of executing the workflow is minimum. The calculated volume and data requirements of the tasks are given. The computation cost of each task on the server and the data communication costs between the servers are given by the cloud service providers. There are many approaches to solving workflow scheduling problems. Evolution algorithms have a fast execution time, but the solution is not optimal. The branch and bound algorithm has a longer execution time, but this algorithm gives an optimal solution.

The rest of the paper is organized as follow. Section II reviews some of the related works about the workflow scheduling algorithms. Section III briefly describes the computation platform on which our algorithm operates. Section IV represent a new scheduling algorithm for data workflow in the cloud environment based on branch and bound algorithm. Section V describes the experiments we have conducted by using some scientific workflows. Section VI concludes our paper and sketches the future works.

2. RELATED WORK

Workflow is a sequence of connected tasks. Workflow scheduling is a big issue in the era of Cloud Computing. Basically it is the issue related to the mapping of each task to an appropriate server and allowing the task to satisfy some performance constraints. The mapping of tasks to the computation resources such as servers is an NP-complete problem [6]. So, past works have proposed many heuristics based approach to scheduling Cloud's workflows.

A. Mohan [7] proposed a scheduling algorithm in heterogeneous cloud computing environment which minimize the makespan of workflow. B. Lin [8] proposed a scheduling algorithm for big data application in Cloud environments. Guo-Ning and Ting-Lei [9] represented an optimized algorithm for task scheduling based on Hybrid Genetic Algorithms. The authors covered in their study the QoS requirements like completion time, bandwidth, cost, distance, reliability of different types of tasks. L. Guo [10] represented a model for task scheduling in Cloud to minimize the overall time of execution and transmission. L. Guo proposed the PSO algorithm which is based on small position value rule. R. Rajkumar [11] proposed an hierarchical scheduling algorithm which helps satisfy service level agreement with quick response from the service provider. S.J. Xue [12] proposed the hybrid PSO algorithm to minimize the cost execution of the workflow. Crossover and mutation of genetic algorithm are embedded into the PSO algorithm to improve the global search. J. Liu In et al. [13] represented the components of an intelligent job scheduling system in cloud computing. Pandey [14] represented a scheduling algorithm (PSO_H) to minimize the total cost of the execution at servers, instead of finding the schedule which has a minimum cost, PSO_H looked for the schedule that minimizes the execution cost of the server which has greatest cost.

3. HETEROGENEOUS COMPUTATION PLATFORM

3.1. Problem formulation

Briefly, CLOWS problem is identified as: Given a set of servers S-the computation resource of the Cloud Center-and a set of workflow tasks T. How to determine a schedule of minimal total cost for T on S.

We denote the workflow as a Directed Acyclic Graph (DAG) represented by G = (V, E), where

- V is set of vertex, each vertex represent a task,
- $T = \{T_1, T_2,\ldots,T_M\}$ is the set of tasks, *M* is the number of tasks,

- E represents the data dependencies between these tasks. The edge $(T_i, T_j) \in E$ means the task T_i is the father of the task T_j, $tdata^k = (T_j, T_k) \in E$ is the data produced by T_j will be consumed by the task T_k. (see Figure 1),
- The Cloud's computation resource, set of servers $S = \{S_1, S_2, \dots, S_N\}$. N is the number of servers.
- The computation of task T_i denoted by W_i (flop-floating point operations).
- $P(S_j)$: the computation power of the server S_j (unit MI/s : million instructions/second).
- The bandwidth $B(S_i,S_j)$ between server S_i and server S_j represents by the function B(): $S \times S \rightarrow R^+$. We assume that $B(S_i,S_i) = \infty$ and $B(S_i,S_j) = B(S_j,S_i)$.
- Each scheduling plan can be represented by the function $f()$: $T \rightarrow S$ where $f(T_i)$ is the server which handle the task T_i.
- x_j^k characterizes where task T_k is processed. $x_j^k = 1$ iff task T_k is processed on server S_j.
- $d_{i,j}^k$ denotes the amount of data to be transferred from server S_i to S_j for task unit iff $x_j^k = 1$. $d_{i,j}^k = 40.1$, denotes 40.1 units of data are to be transferred from S_i to S_j for task T_k
- $tfcost_{i,j}$ characterizes the cost of data transfer for a link per data unit. It is added to the overall cost iff $d_{i,j}^k > 0$ and $x_j^k = 1$
- $excost_j$ characterizes the cost of computation of a Server. $excost_j = 1$ denotes the cost of using a Server S_j. It is added to the overall cost iff $x_j^k = 1$
- $tftime_{i,j}$ denotes the time for transferring amount data from sever S_i to S_j for task T_k iff $d_{i,j}^k > 0$ and $x_j^k = 1$

$$tftime_{i,j} = \frac{d_{i,j}^k}{B(S_i,S_j)} \tag{1}$$

- $extime_j^k$ denotes the time for executing a task T_k on server S_j. It is added to execution time of Server S_j iff $x_j^k = 1$ and calculated as equation (2).

$$etime_j^k = \frac{W_k}{P(S_j)} \tag{2}$$

- Execution time of task T_k denotes as ET^k

$$ET^k = \sum_{i=1}^{N} \sum_{j=1}^{N} d_{i,j}^k \times tftime_{i,j} \times x_j^k + \sum_{j=1}^{N} extime_j^k \times x_j^k \tag{3}$$

- We denote the cost of the workflow as CT :

$$CT = \sum_{i,j \in S, k \in T} d_{i,j}^k \times tf\cos t_{i,j} \times x_j^k + ex\cos t_j \times extime_j^k \times x_j^k \tag{4}$$

- Formally, we need to minimize the cost of the workflow ; find CT_{min} = Min{CT= $\sum_{i,j \in S, k \in T} d_{i,j}^k \times tf\cos t_{i,j} \times x_j^k + ex\cos t_j \times extime_j^k \times x_j^k$ }

The constraints can be described as follows:

a) $x_j^k \geq 0;\ \forall k = 1,2,..,M \ and\ j = 1,2,\dots,N$

b) $d_{i,j}^k \geq 0, \forall i,j = 1,2,..,N \ and\ k = 1,2,\dots,M$

c) $tdata_j^k \geq 0, \forall k = 1,2,..,M$

d) $tftime_{i,j} \geq 0;\ \forall i,j = 1,2,\dots,N$

e) $extime_j^k \geq 0;\ \forall k = 1,2,..,M\ and\ j = 1,2,\ldots,N$

f) $\sum_{j=1}^{N} x_j^k = 1$

g) $\sum_{i=1}^{N}\sum_{j=1}^{N} x_j^k \times d_{i,j}^k = tdata^k$

h) $\sum_{k=1}^{M}\sum_{i=1}^{N}\sum_{j=1}^{N} x_j^k \times d_{i,j}^k = \sum_{k=1}^{M} tdata^k$

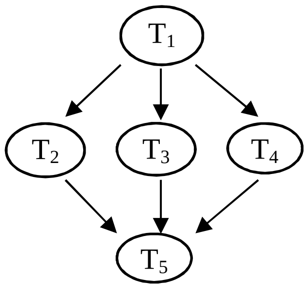

Figure 1. An example of workflow with 5 tasks

3.2. Problem complexity

Theorem 1: CLOWS is NP-Hard in strong sense.

Proof.

Let's consider the SCHED problem, which have described and proved by O. Sinnen that to be NP-Hard in strong sense [15].

Table 1. The comparison between SCHED and CLOWS problem.

Assumptions, constraints and the objective	SCHED problem	CLOWS problem
Computation power of servers	Homogeneous: the computation power of servers are the same: P(S_i) = P(S_j) ($\forall i,j$)	Heterogeneous: the computation power are not uniform.
The execution progressing of tasks	A task could be executed by an arbitrary server, but by no more than one server. Each server could not execute more than one task at a time.	The same as SCHED
Communication speed between servers	Homogeneous: the bandwidth of connections are the same: B(S_i, S_k) = B(S_u, S_v) $\forall$ *i,k,u,v*	Heterogeneous: the bandwidth of connections are not uniform
Objective function	*Minimize the makspan of workflow*	*Minimize the total cost of workflow*
Data dependencies between tasks	If task T_i was the father of the task T_k, then the data produced by T_i will be consumed by the task T_k	The same as SCHED

Obviously, the main observation from Table 1 is that SCHED problem is a special case of CLOWS problem where computation power of servers and communication speed of connections are uniform.

Assume that there is an algorithm X which could be used to find out the optimal schedule for the CLOWS problem. Since SCHED problem is sub instance of CLOWS problem, so algorithm X could also be used to find out the optimal schedule for the SCHED problem, which mean that SCHED ∞ CLOWS.

As J.D. Ullman showed in [6], if SCHED ∞ CLOWS then CLOWS is NP-Hard in strong sense.

4. PROPOSED ALGORITHM

4.1. Branch and Bound Algorithm

Branch and bound algorithms are a variety of adaptive partition strategies that have been proposed to solve global optimization models. These are based upon partition, sampling, and subsequent lower and upper bounding procedures: these operations are applied iteratively to the collection of active subsets within the feasible set D. Their exhaustive search feature is guaranteed in similar spirit to the analogous integer linear programming methodology. Branch and Bound Algorithm consists of two main procedures:

Branching: splitting the problem into sub-problems.

Bounding: calculating lower and/or upper bounds for the objective function value of the sub-problem.

The branching is performed in the following algorithm by separating the current subspace into two parts using the integrality requirement. Using the bounds, unpromising sub-problems can be eliminated.

```
Procedure Branch(k)
1.  begin
2.  for  $a_k \in A_k$  do
3.   if   $a_k \in S_k$ then
4.   begin
5.     $x_k$ :=  $a_k$;
6.     if(k = n)then <update fopt>
7.    else
8.   if g($x_1$, $x_2$,…,$x_k$) $\le$ fopt then
9.  Branch(k+1)
10.  end;
end;
```

```
Procedure BranchAndBound

  1.  begin
  2.  fopt:= +∞;
  3.  Branch(1);
  4.  if fopt < +∞ then
  5.     return fopt
end;
```

4.2. Proposed Algorithm

Solution representation

In the proposed scheduling algorithm, the solution is represented as a vector of length equal to the number of tasks. The value corresponding to each position *i* in the vector represents the server to which task *i* was executed.

Example 1

Consider a workflow with a set of tasks T={T_1, T_2, T_3, T_4, T_5}, a set of servers S = {S_1, S_2, S_3}. So the particle x_i^k = [1 ; 2 ; 1 ; 3 ; 2] gives us the following scheduling plan:

T_1	T_2	T_3	T_4	T_5
S_1	S_2	S_1	S_3	S_2

In that scheduling plan, tasks T_1 and T_3 will be executed by the server S_1, tasks T_2 and T_5 are assigned to the server S_2 and task T_4 is handled by server S_3.

Lower bound function

Each solution of the problem is an M-dimensional vector $x = (x_1, x_2,\ldots,x_M); x_j \in S$

Assuming that $C_{max} = max\{P(S_j)\}; j \in S;$ *S_j is the the greatest computing power*

Consider the partial solution $(x_1, x_2,\ldots,x_L)$, In that scheduling plan $S_L= (S_{x1}, S_{x2},..,S_{xL})$, and the cost of this partial solution is:

$$\delta = \sum_{i,j\in S,k\in T_L} d_{i,j}^k \times tf\cos t_{i,j} \times x_j^k + ex\cos t_j \times extime_j^k \times x_j^k \quad (5)$$

The lower bound function of partial solution will be calculated as the following:

$$g(k) = \delta + \frac{1}{C_{\max}} \times \sum_{j\in S,k\in T-T_L} ex\cos t_j \times W^k \times x_j^k \quad (6)$$

Proof: We have

$$min\{\textstyle\sum_{i,j\in S,k\in T} d_{i,j}^k \times tfcost_{i,j} \times x_j^k + excost_j \times extime_j^k \times x_j^k\}$$

$$= min\left\{\sum_{i,j\in S,k\in T_L} d_{i,j}^k \times tfcost_{i,j} \times x_j^k + excost_j \times extime_j^k \times x_j^k + \sum_{i,j\in S,k\in T-T_L} d_{i,j}^k \times tfcost_{i,j} \times x_j^k + excost_j \times extime_j^k \times x_j^k\right\}$$

$$= min\{\delta + \textstyle\sum_{i,j\in S,k\in T-T_L} d_{i,j}^k \times tfcost_{i,j} \times x_j^k + excost_j \times extime_j^k \times x_j^k\} =$$

$$= \delta + min\left\{\sum_{i,j\in S,k\in T-T_L} d_{i,j}^k \times tfcost_{i,j} \times x_j^k + excost_j \times extime_j^k \times x_j^k\right\}$$

$$= \delta + min\left\{\sum_{i,j\in S,k\in T-T_L} d_{i,j}^k \times tfcost_{i,j} \times x_j^k + excost_j \times \frac{W_k}{P(S_j)} \times x_j^k\right\}$$

$$\geq \delta + min\left\{\sum_{i,j\in S,k\in T-T_L} excost_j \times \frac{W_k}{P(S_j)} \times x_j^k\right\}; because \sum_{i,j\in S,k\in T-T_L} d_{i,j}^k \times tfcost_{i,j} \geq 0$$

$$\geq \delta + \frac{1}{C_{max}}\left\{\sum_{j\in S,k\in T-T_L} excost_j \times W_k \times x_j^k\right\} = g(k)$$

So, g(k) is the lower bound function of partial solution.

Based on the lower bound function and Branch and Bound method we proposed the following algorithm:

Algorithm BBScheduling

```
Input:  set of tasks T, set of servers S, size of workload W[1×M],
     server's execution cost TP[M×N], cost of communication between
     servers PP[N×N], communication data  D[M×M]
Output: best solution
 function cost(x1, x2,..,xM)
 begin
 return  $\sum_{i,j\in S,k\in T_L} d_{i,j}^k \times tx\cos t_{i,j} \times x_j^k + ex\cos t_j \times extime_j^k \times x_j^k$ ;
 end;
 Procedure SchedulingBranch(k)
 begin
  for j:=1 to M do
   if UCV(j,k) then
   begin
    a[i]:=j;
    if i=M then Ghinhan;
    else if g(k)< fopt then
      SchedulingBranch(k+1);
   end;
 end;
 procedure candidates(j,k)
 begin
  var i:integer;
  for i=1 to k-1 do
    if j=ai then
      return false;
    else return true;
 end;
 procedure record
 begin
  double c = cost(x1, x2,..,xM);
  if c < fopt then
    fopt = c;
end
Algorithm BBScheduling
 begin
 1. Calculate average computation cost of all tasks in all compute
    resources
 2. Calculate average cost of (communication/size of data) between
    resources
 3. double fopt = +∞;
 4. SchedulingBranch(1);
 5. writeln(fopt);
 end
```

The BBScheduling algorithm works as following:

To generate an empty schedule with no server sequenced and indicate this by $(t_1*, t_2^*, t_3^*, .., t_M^*)$. Here "*" in the task sequence indicates that no server has yet been assigned to execute task in that position.

To construct a schedule starting from the first position, we move from node $(t_1*, t_2^*, t_3^*, .., t_M^*)$ to one of the M possible nodes $(S_i, t_2^*, t_3^*, .., t_M^*)$; $(S_j, t_2^*, t_3^*, .., t_M^*)$; .. ; $(S_k, t_2^*, t_3^*, .., t_M^*)$. S_i, S_j, $S_k \in S$

To assign the second task in the sequence, we branch from the each of these M nodes to other possibilities. Example branching from $(S_i, t_2^*, t_3^*, .., t_M^*)$ gives $(S_i, S_j, t_3^*, .., t_M^*)$, $(S_i, S_k, t_3^*, .., t_M^*)$,…

Assigning the task to be processed in the third position immediately fixes the last task.

This process is represented by a branching tree. Each node of a tree corresponds to a partial schedule with several server assigned to the first positions. To avoid full enumeration of all task permutations, we calculate in each step the lower bound function by equation (6) of the value of the objective function for each partial schedule.

5. EXPERIMENTAL RESULTS

5.1. Problem Instance

We use both random and real world instances in our experiments using the following data sets:

The cost of unit data transfer between servers and the processing cost of servers are collected from a Cloud provider such as Amazon [16] and its Web site (exp. http://aws.amazon.com/ec2/pricing)

The sets of working data are collected from Montage project [1] and Epigenomics [3], an Epigenomics's workflow is depicted in Figure 2. The instances are divided into 5 groups based on the number of servers N, the number of tasks M and ratio α:

Group 1: M = 10, N = 3, α=0.3; Group 2: M = 10, N = 5, α =0.2 ;
Group 3: M = 10, N = 5, α =0.53 ; Group 4: M = 20, N = 3, α =0.15;
Group 5: M = 10, N = 5, α =0.3

We denote the ratio of the number of edges and the number of vertexes of graph G by:

$$\alpha = \frac{|E|}{M \times (M-1)/2}$$

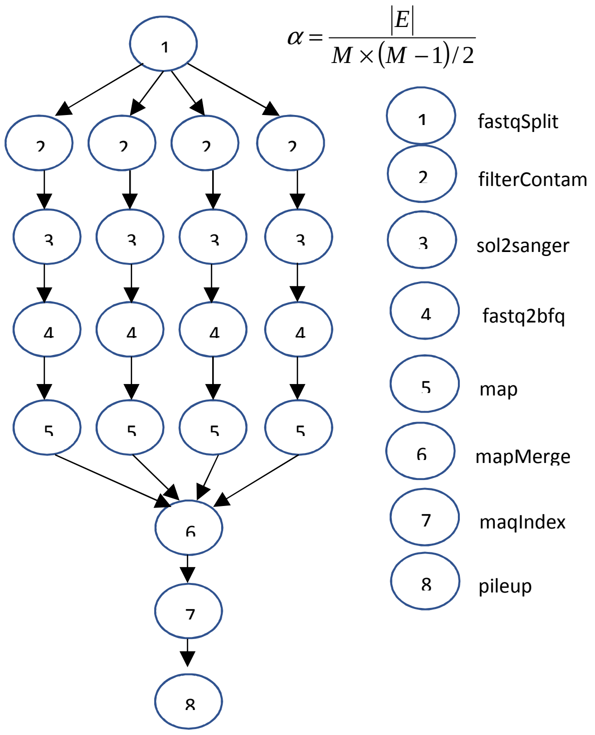

Figure 2. Workflow of Epigenomics [17].

5.2. Experiments

In this paper we perform exhausted search and compare with the result of BBS cheduling algorithm, the results have been illustrated in the Table 2.

Table 2. Experiments results.

Data	M	N	α	BBScheduling		Exhausted Search Algorithm	
				Cost	Execution time	Cost	Excution time
T_1	10	3	0.3	7470.7	2 (s)	7470.7	3 (s)
T_2	10	5	0.2	4866.2	5 (s)	4866.2	28 (m)
T_3	10	5	0.53	5583.9	7 (s)	5583.9	30 (m)
T_4	20	3	0.15	8679	6 (m)	8679	121 (h)
T_5	20	5	0.3	8685.4	6 (m)	8685.4	132 (h)

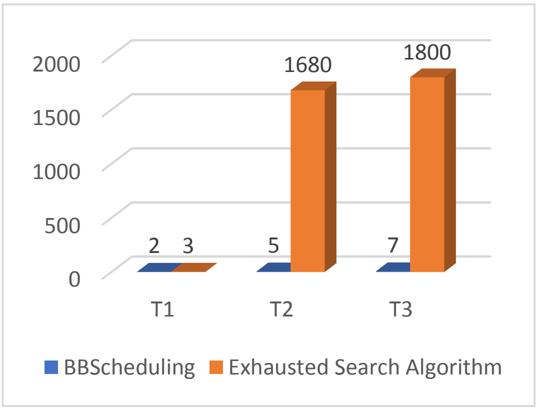

Figure 3. Experiments results of the Instance 1, 2, 3.

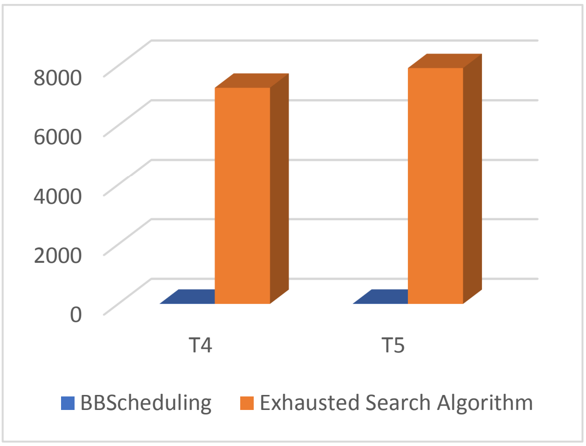

Figure 4. Experiments results of the Instance 4,5.

The results show that both algorithms find the optimal solution, the execution time of BBScheduling algorithm is smaller than the execution time of exhausted algorithm. Especially when the number tasks of workflow increases, the execution time of exhausted algorithm is very large. Example when the number of tasks are 20 and number of Servers are 3, the excution time of exhausted algorithm is 121 hours.

5.3. Results and Discussion

Workflow scheduling is an NP-hard problem, the execution time increase exponentially by the data input, computational complexity of this case is $O(M^N)$, with M is the number of tasks and N is the number of servers. Proposed algorithm that solves the problem with medium and small input data, the execution time of the algorithm is considerably smaller than execution time of the exhausted search algorithm. The results summarized in Table 2 and Figure 3, 4 depict the performance of algorithms where the vertical axis represents the execution time of the algorithms.

6. CONCLUSION

The ultimate goal of any scheduling algorithm is the optimum solution which minimize the execution time. In this paper we proposed a scheduling algorithm based on Branch and Bound method. Our contributions can be summarized as follows:

- Announcing and formulating a new problem about Workflow Scheduling on Cloud Center which called CLOWS (Cloud Workflow Scheduling). We also prove that CLOWS belongs to NP-Hard class
- Proposing a new scheduling algorithm named BBScheduling based on the Branch and Bound method.

The experiment's results show that execution time of BBScheduling is smaller than the execution time of the exhausted search algorithm, especially when it works in a small scale Cloud, i.e. the number of servers and tasks are not very large. In the future, we are planning to improve this algorithm for solving bigger instances by using evolution algorithms.

REFERENCES

1. Berriman G. B, Ewa D., John G., Joseph J., Daniel K.S., Carl K., Anastasia L., Thomas P.A., Gurmeet S., Mei-Hu S. - Montage: A Grid Enabled Engine for Delivering Custom Science Grade Mosaics On Demand, Proceedings of the SPIE **5493** (2004) 221-232 .
2. Maechling P., Ewa D., Li Z., Robert G., Gaurang M., Nitin G., John M., Carl K., Scott C., David O., Hunter F., Vipin G., Karan V., Thomas J., Edward F. - SCEC CyberShake Workflows, Automating Probabilistic Seismic Hazard Analysis Calculations, Springer, 2007, pp.143-163.
3. Hui S., Peter W. L. - Interplay between the Cancer Genome and Epigenome, Cell Elsevier Inc., ISSN: 0092-8674, **153** (1) (2013) 38-55.
4. The Laser Interferometer Gravitational-Wave Observatory and the first direct observation of gravitational waves, The Royal Swedish Academy òf Sciences (2017)
5. Abbott B. P. - LIGO: the Laser Interferometer Gravitational-Wave Observatory, Proceedings of the Conference ICALEPCS, San Jose, USA, 2001, pp. 145-152.

6. Ullman J. D. - NP-complete scheduling problems, Journal of Computer and System Sciences **10** (3) (1975) 384-393.
7. Mohan A., Ebrahimi M., Lu S. and Kotov A. - Scheduling big data workflows in the cloud under budget constraints, Proceedings of the IEEE International Conference on Big Data, Washington, 2016, pp. 2775-2784.
8. Lin B., Guo W., Xiong N., Chen G., Vasilakos A. V. and Zhang H. - A Pretreatment Workflow Scheduling Approach for Big Data Applications in Multicloud Environments, IEEE Transactions on Network and Service Management **13** (2016) pp.581-594.
9. Guo-Ning G. and Ting-Lei H. - Genetic Simulated Annealing Algorithm for Task Scheduling based on Cloud Computing Environment, Proceedings of International Conference on Intelligent Computing and Integrated Systems, 2010, pp. 60-63.
10. Guo L., Zhao S., Shen S., Jiang C. - Task Scheduling Optimization in Cloud Computing Based on Heuristic Algorithm, Journal of Networks, ACADEMY PUBLISHER **7** (3) (2012) 547-552.
11. Rajkumar R., Mala T. - Achieving Service Level Agreement in Cloud Environment using Job Prioritization in Hierarchical Scheduling, Proceeding of the International Conference on Information System Design and Intelligent Application, Springer Verlag Berlin Heidelberg **132** (2012) 547-554.
12. Xue S. J., Wu W. - Scheduling Workflow in Cloud Computing Based on Hybrid Particle Swarm Algorithm, Indonesian Journal of Electrical Engineering **10** (2012) pp.1560-1566.
13. Liu J., Luo X., Li B., Zhang X., Zhang F. - An Intelligent Job Scheduling System for Web Service in Cloud Computing, Indonesian Journal of Electrical Engineering **11** (2013) 2956-2961.
14. Kennedy J., Eberhart R. C. - Particle swarm optimization, Proceeding of IEEE International Conference on Neural Networks, 1995, pp.1942–1948.
15. Sinnen O. - Task scheduling for parallel systems, John Wiley & Sons Inc., ISBN: 978-0-471-73576-2, 2007.
16. Vliet J. V., Paganelli F., - Programming Amazon EC2, O'Reilly Media, ISBN 1449393683, 2011.
17. https://confluence.pegasus.isi.edu/display/pegasus/WorkflowGenerator.