Improving Max-Min scheduling Algorithm for Reducing the Makespan of Workflow Execution in the Cloud

Ali S. A. Al-Haboobi
Computer Science Dept., Faculty of Computer Science and Mathematics, University of Kufa, Najaf, Iraq

ABSTRACT
Cloud computing has become a new paradigm that provides IT resources as a service based on the basis: pay-per-use over the internet. Scientific workflow applications can gain the advantage by running on cloud resources. However, the optimized of workflow scheduling algorithms is a challenge and still needed further work. This paper presents an improved Max-Min algorithm based on the Max-Min algorithm. It can minimize the makespan of workflow execution and increases the resource utilization. The algorithm tested using WorkflowSim with five workflows from the Pegasus workflow management system. The results show that this algorithm can achieve better than Max-Min in most of the cases.

General Terms
Distributed Computer Systems, Algorithms.

Keywords
Cloud Computing, Scientific workflow, Scheduling algorithms, Max-Min, Makespan.

1. INTRODUCTION
Over the recent years, cloud computing [1] has become an important technology in the IT systems. It delivers IT resources for scientific and commercial users in an effective way and plays a vital role in processing scientific workflow applications in distributed systems [2]. Companies can easily rent resources from cloud for computational purposes and storage to run their analyses and consequently their cost will be significantly reduced. The scientific workflow applications [3] such as Cyber Shake and Montage need large processing of data which is possible by resources of cloud computing.

Scheduling mechanism is a crucial issue in cloud computing that requires to design scheduling algorithms for achieving effective mapping of workflow tasks to cloud resources. Many algorithms are proposed such as Max-min, Min-Min, and MCT algorithms, but they are still needed improving to reduce the workflow execution time [4].

In this paper, an improved Max-Min algorithm has proposed based on the Max-Min algorithm. It performs better than Max-Min in most of the cases by minimizing the total task execution time (makespan) of a workflow running in the cloud and increasing the resource utilization.

The remaining parts of this paper are organized as follows: section 2 presents the scientific workflows representation. Section 3 contains the background and related work. Section 4 presents the proposed algorithm and section 5 involves the scheduling experiment and experimental results. Finally, the conclusion and future work is presented in section 6.

2. SCIENTIFIC WORKFLOWS
Scientific workflow consists of a set of atomic tasks which follow precedence-constrained in processing. It is represented as Directed Acyclic Graph (DAG), G= (V, E), where V is a set of tasks = {T1, T2, … , Tn} and E is a set of edges which each edge links two tasks and represents their data dependency [5]. In the workflow each task can be executed if all its parents have completed processing.

3. BACKGROUND AND RELATED WORK
Generally, the problem of scheduling is known as a NP-Complete problem [6] which cannot solve it as an optimal solution in polynomial time by any algorithm. However, some algorithms perform in polynomial time that can provide good results. For example, the Max-Min algorithm is one of the heuristic algorithms that can minimize the makespan.

3.1 Max-Min Scheduling Algorithm
Max-Min algorithm assigns a task to the resource based on the minimum completion time. The idea of Max-Min algorithm is to map the largest task among all the tasks to the fastest machine among all the machines. Then, it assigns the largest task among all the remaining tasks to the fastest machine among all the remaining machines and so on until all the tasks are scheduled [7]. The algorithm may assign a task with larger execution time to a slower machine, and as a result, the makespan will increase. Consequently, the aim of this paper wants to improve the Max-Min algorithm for workflow scheduling to minimize the overall makespan of the workflow execution.

3.2 Related Work
The workflow scheduling is an important component in the applications of cloud computing or distributed environment, and researchers have shown an increased interest in it.

Parsa and Maleki [8] presented the RASA algorithm that is combined the Max-Min and Min-Min algorithms. It composed these algorithms to obtain their advantages and eliminate their disadvantages. For example, if the first task is mapped to a machine based on the Max-Min algorithm, the next task will be mapped with the Min-Min algorithm and it repeats until all tasks being scheduled.

Bhoi and Ramanuj [9] proposed an enhanced Max-Min algorithm. It computes the estimated completion time of the scheduled tasks on each machine. Next, by assigning the task

that is greater than the average execution time to the slowest machine which has the overall minimum completion time.

The Heterogeneous Earliest Finish Time algorithm (HEFT) has proposed by Topcuoglu et al. [5]. The idea of this algorithm is to calculate the average of execution time of each task as well as the connection time between the machines of two tasks. Next, the rank function is calculated based on the summation of average execution time and communication time. Then, the rank function sorts its values in non-ascending order and a task with higher rank value set higher priority. In the selection phase, the scheduling of tasks are based on their priorities. Furthermore, each task is mapped to the machine to minimize the makespan.

Sun et al. [10] proposed the Period Ant Colony Optimization algorithm (PACO) depends on an ACO algorithm. The result of PACO's simulations achieved better than the Min-Min algorithm.

Ming and Li [11] investigated an enhanced algorithm MMST based on the Max-Min algorithm because of Max-Min is not achieved better results. MMST minimizes the waiting time and increases the resource utilization of tasks.

4. IMPROVING MAX-MIN

The basic Max-Min algorithm determines the completion times for each task on all VMs. It sets the priority to the task that require longest execution time instead of the smallest execution time. It assigns the first longest task among all tasks to the first fastest VM among all VMs. Then, the second longest task assigns to the second fastest VM. The same procedure is repeated until all task being scheduled. When using Max-Min in workflow scheduling, there is a possibility of assigning a larger task to a slower machine and consequently the overall total execution time will increase. Therefore, an improved Max-Min algorithm is wanted to avoid this issue.

To tackle this problem, an improved Max-Min algorithm has been proposed based on Max-Min algorithm. This algorithm achieves better results than Max-Min in most of the cases. It calculates the average of execution time for all tasks in the workflow. Next Max-min is used when receiving a task with execution time is smaller than the average. Otherwise a task with execution time greater than or equal to the average is assigned to the VM with minimum completion time among all the VMs regardless of VM availability. Where the completion time represents machine's ready time with task's execution time. Figure 1 shows the flowchart of an improved Max-Min algorithm for workflow scheduling.

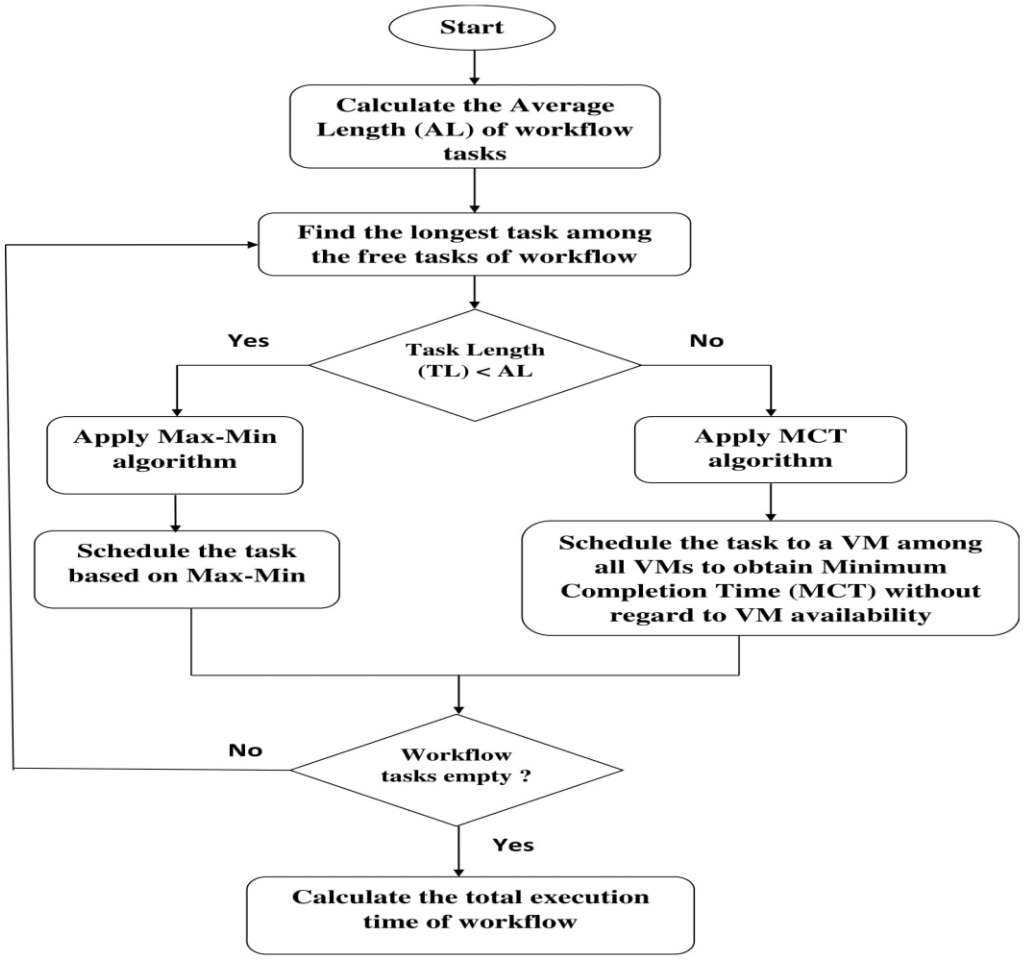

Fig 1: Flowchart of an Improved Max-Min Algorithm.

5. SCHEDULING EXPERIMENT

To evaluate the efficiency of the proposed algorithm, the output of an improved Max- Min algorithm is compared with output of Max-Min. The proposed algorithm was simulated by the WorkflowSim 1.1.0 toolkit [12]. WorkflowSim is an open source that is an extension of the CloudSim framework [13]. It supports dynamic scheduling algorithms.

The experiment was conducted on one data center with six VMs. Each VM has 512 MB of RAM and one CPU as well as Bandwidth of 1000 Mbps. The speed of VMs was (6000, 4000, 2000, 2000, 1000, 500) MIPS respectively. The results were tested with five realistic scientific workflows which are: Montage, CyberShake, Epigenomics, LIGO, and SIPHT as shown in Figure 3. The workflow of real-world applications including Montage is used in astronomy, CyberShake is used in earthquake science, Epigenomics and SIPHT are used in biology, and LIGO is used in gravitational physics [3].

To calculate the average length of the workflow tasks, the following formula is used:

$$\mathbf{AVG} = \sum_{k=1}^{n} \frac{(TL_k)}{n}$$

where TL_k represents the task length of the kth task and n is the total number of workflow tasks. The results obtained (in seconds) for the Improved Max-Min and the Max-Min algorithms as shown in Table l.

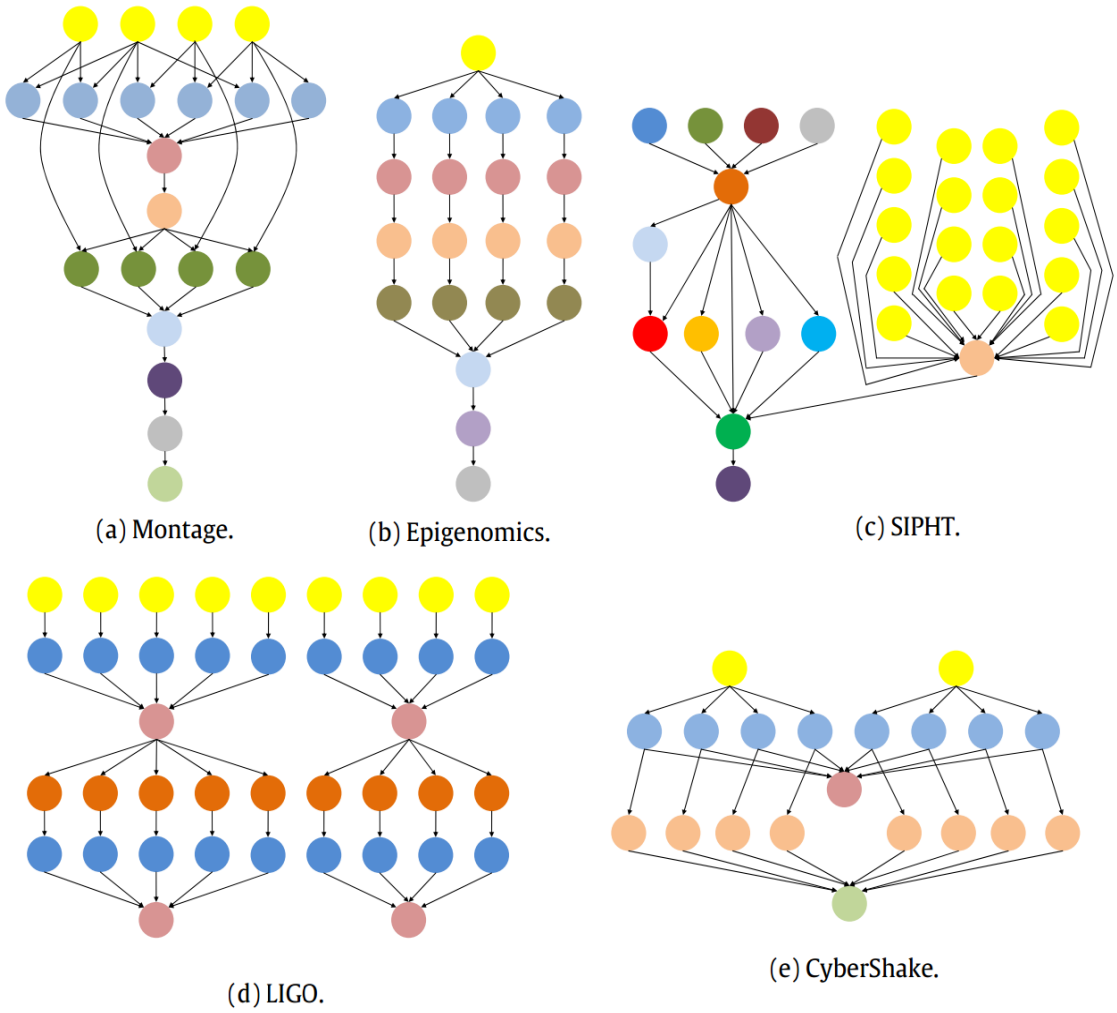

Fig 2: The structure of five realistic scientific workflows [3].

Table l: The makespan's comparison in seconds of scheduling algorithms using five scientific workflows with different numbers of workflow tasks.

Workflow	Improved Max-Min	Max-Min
Montage_25	22.35	39.86
Montage_50	38.69	72.25
Montage_100	85.67	96.89
Montage_1000	783.63	791.91
Epigenomics_24	1627.28	2864.4
Epigenomics_46	4548.28	7610.39
Epigenomics_100	32047.38	36128.67
Sipht_30	734.94	734.94
Sipht_100	1468.46	3149.59
LIGO_30	647.91	1815.76
LIGO_50	920.38	1932.48
LIGO_100	1613.92	1658.63
CyberShake_30	72.52	81.92
CyberShake_50	112.25	130.04
CyberShake_100	218.32	272.21

Figure 3, 4, 5 and 6 showed the graphical analysis of the results using bar chart. Figure 3, 4, 5 and 6 show the reduction in makespan for all workflows with all datasets excluding the Sipht_30 dataset. While the makespan of Improved Max-Min and Max-Min algorithms is equal for the Sipht_30 dataset. In some datasets, there are a significant decrease in makespan such as Epigenomics_24, Epigenomics_46, Sipht_30, Sipht_100, LIGO_30 and LIGO_50. For example, the makespan of improved Max-Min is shorter about 50% than Max-Min in Sipht_100, LIGO_30 and LIGO_50. Whereas, it is also shorter about 40%, 43%, 46% and 44% in Epigenomics_46, Epigenomics_24, Montage_50 and Montage_25 respectively.

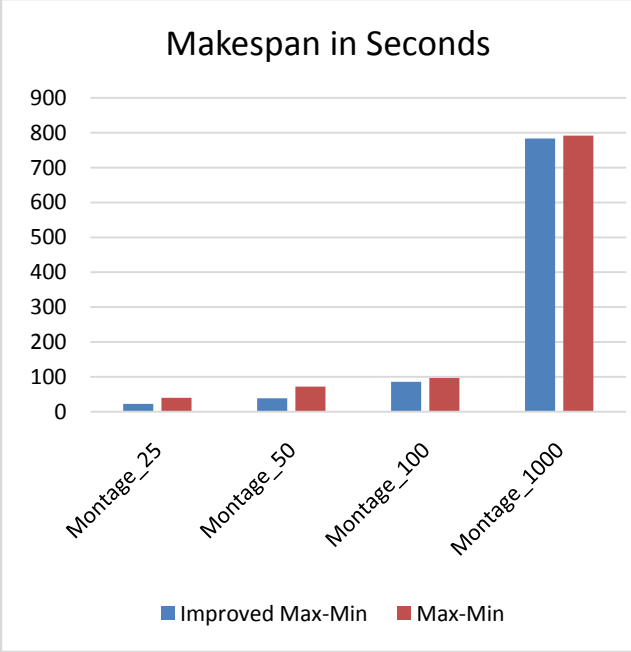

Fig 3: The makespan of Improved Max-Min and the Max-Min algorithms using the Montage workflow.

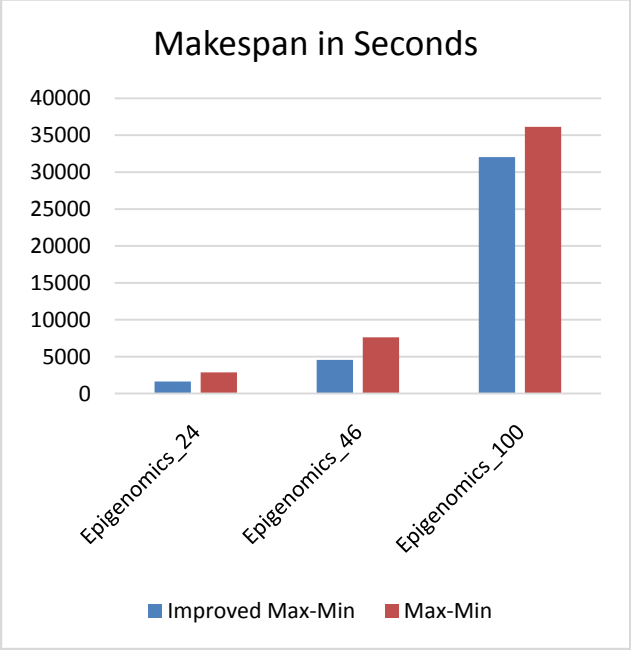

Fig 4: The makespan of Improved Max-Min and the Max-Min algorithms using the Epigenomics workflow.

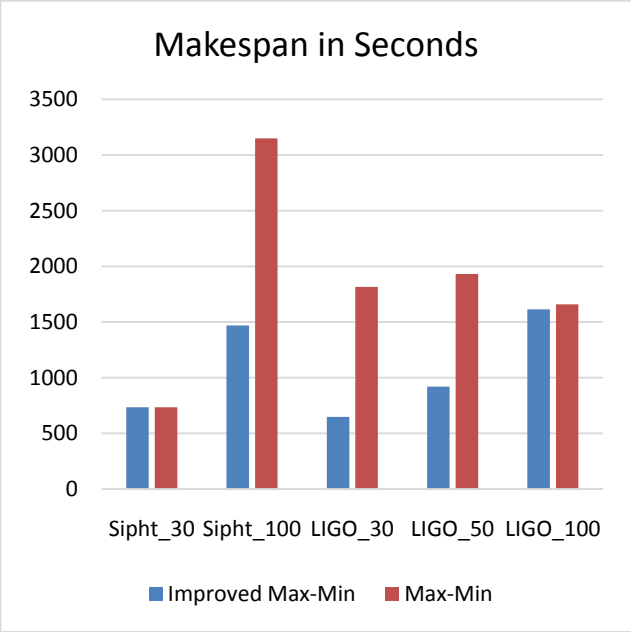

Fig 5: The makespan of Improved Max-Min and the Max-Min algorithms using the Sipht and LIGO workflows.

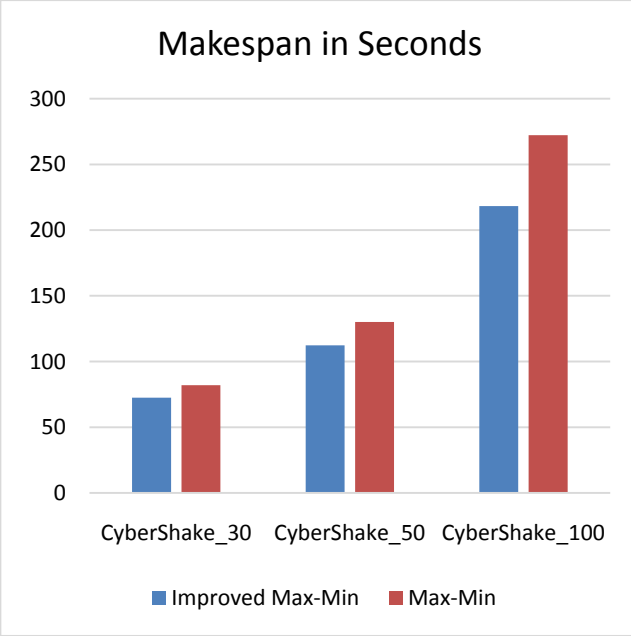

Fig 6: The makespan of Improved Max-Min and the Max-Min algorithms using the CyberShake workflow.

6. CONCLUSION AND FUTURE WORK

Max-Min algorithm is applicable in small scale distributed systems. Whereas, if a task with larger makespan assigns to a machine with slower speed, the overall makespan of workflow execution will increase. As a result, an improved Max-Min algorithm proposed to minimize makespan based on the Max-Min algorithm. The experimental results show that an improved Max-Min algorithm outperforms the Max-Min algorithm in most of the cases. This study is focused on decreasing the overall makespan of the workflow execution. Further study of the issue would be of interest. By considering QoS parameters such as deadline and budget in the way that user can provide constraints in order to improve the performance of workflow execution in the cloud environment.

7. REFERENCES

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," Future Gener. Comput. Syst., vol. 25, pp. 599-616, 2009.

[2] E. Deelman, D. Gannon, M. Shields, and I. Taylor, "Workflows and e-Science: An overview of workflow system features and capabilities," Future Gener. Comput. Syst., vol. 25, pp. 528-540, 2009.

[3] S. Bharathi, A. Chervenak, E. Deelman, G. Mehta, M. H. Su, and K. Vahi, "Characterization of scientific workflows," in 2008 Third Workshop on Workflows in Support of Large-Scale Science, 2008, pp. 1-10.

[4] A. Abraham, R. Buyya, and B. Nath, "Nature's heuristics for scheduling jobs on computational grids," in The 8th IEEE international conference on advanced computing and communications (ADCOM 2000), 2000, pp. 45-52.

[5] H. Topcuoglu, S. Hariri, and W. Min-You, "Performance-effective and low-complexity task scheduling for heterogeneous computing," IEEE Transactions on Parallel and Distributed Systems, vol. 13, pp. 260-274, 2002.

[6] J. D. Ullman, "NP-complete scheduling problems," J. Comput. Syst. Sci., vol. 10, pp. 384-393, 1975.

[7] F. Dong and S. G. Akl, "Scheduling algorithms for grid computing: State of the art and open problems," Technical report2006.

[8] S. Parsa and R. Entezari-Maleki, "RASA: A new task scheduling algorithm in grid environment," World Applied sciences journal, vol. 7, pp. 152-160, 2009.

[9] U. Bhoi and P. N. Ramanuj, "Enhanced max-min task scheduling algorithm in cloud computing," International Journal of Application or Innovation in Engineering and Management (IJAIEM), vol. 2, pp. 259-264, 2013.

[10] W. Sun, N. Zhang, H. Wang, W. Yin, and T. Qiu, "PACO: A period ACO based scheduling algorithm in cloud computing," in Cloud Computing and Big Data (CloudCom-Asia), 2013 International Conference on, 2013, pp. 482-486.

[11] G. Ming and H. Li, "An Improved Algorithm Based on Max-Min for Cloud Task Scheduling," in Recent Advances in Computer Science and Information Engineering: Volume 2, Z. Qian, L. Cao, W. Su, T. Wang, and H. Yang, Eds., ed Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 217-223.

[12] W. Chen and E. Deelman, "Workflowsim: A toolkit for simulating scientific workflows in distributed environments," in E-Science (e-Science), 2012 IEEE 8th International Conference on, 2012, pp. 1-8.

[13] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Software: Practice and experience, vol. 41, pp. 23-50, 2011.