List-Based Workflow Scheduling Utilizing Deep Reinforcement Learning

Wei-Cheng Tseng and Kuo-Chan Huang(✉)

Department of Computer Science, National Taichung University of Education, No.227, Minsheng Rd., West Dist., Taichung City 403012, Taiwan (R.O.C.)
kchuang@mail.ntcu.edu.tw

Abstract. Workflow scheduling is a well-known NP-complete research problem with wide applications and increasing importance. Traditionally, heuristic and guided random search methods, e.g. genetic algorithm, are the two major categories of scheduling approaches developed to tackle this challenging problem. With the rise of deep reinforcement learning (DRL), this paper tries to apply it to solve the workflow scheduling problem in two different ways. The first way utilizes DRL as an iterative optimization method to find the best schedule for a specific workflow. In the second way, DRL is used to train a neural network which could be adopted to schedule new workflows not in the training set. Our DRL-based workflow scheduling method is based on the policy gradient (PG) reinforcement learning algorithm and utilizes a convolutional neural network (CNN). Experimental results show that our DRL-based method can produce more efficient workflow execution schedules, compared to the state of the art of heuristic-based scheduling algorithms. The superior performance of the DRL-based method indicates a promising direction for future research work on workflow scheduling.

Keywords: Deep Reinforcement Learning · Workflow Scheduling · Policy Gradient · Convolutional Neural Network

1 Introduction

Workflow is a common task-parallel model for parallel computing [5] with various applications, ranging from instruction scheduling within parallel compilers in earlier days to modern high performance computing on cloud platforms. A workflow is usually represented by a Directed Acyclic Graph (DAG), G = (V,E), where V is the set of nodes representing computational tasks in the workflow, and E is the set of edges describing precedence relations among tasks. Figure 1 shows an example of workflow DAG structure. Each node, i.e. a computational task, is annotated with a value indicating the required execution time. The number next to an edge represents the required data transfer time between the two connected tasks.

The workflow scheduling problem on a parallel computing platform is to determine the start time and processor allocation for each task in order to optimize a specific goal, e.g. the execution makespan between the start time of the first task to the finish time of the last task. Workflow scheduling is a well-known NP-complete problem [5]. Therefore, instead of finding optimal solutions, various practical algorithms were developed to produce good-enough schedules in reasonable time. Most previous workflow scheduling methods can be roughly categorized into two types [1]: heuristic-based and guided random search approaches, e.g. genetic algorithm.

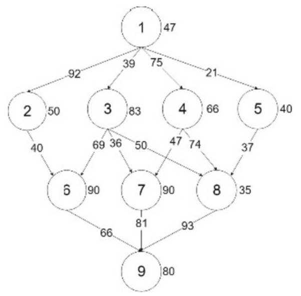

Fig. 1. Workflow structure represented by a DAG

Recently, deep reinforcement learning (DRL) has shown great achievements and potential in many application fields, e.g. game playing [6]. In this paper, we try to apply DRL to the workflow scheduling problem. Reinforcement learning (RL) [7] is a subfield of machine learning where an agent learns to make the best decisions by trial and error, through interaction with the environment, in order to maximize a specific kind of cumulative reward. DRL combines reinforcement learning and deep learning, allowing agents to make decisions from unstructured input data and large complex state space. Our DRL-based scheduling method is based on the policy gradient (PG) [7] reinforcement learning algorithm and utilizes a convolutional neural network (CNN) [8].

The proposed DRL-based workflow scheduling method could be used in two different ways. The first way utilizes DRL as an iterative optimization method to find the best schedule for a single workflow from scratch. In the second way, DRL is adopted to train a neural network based on a set of workflows first. The neural network model could then be used to schedule new workflows not in the training set. Experimental results show that our DRL-based method can produce more efficient workflow execution schedules in terms of makepsan, compared to the most famous heuristic-based workflow scheduling algorithm HEFT [1] and the state of the art of heuristic-based scheduling algorithms IPPTS [2]. The superior performance of the proposed DRL-based method indicates a promising direction for future research work on workflow scheduling.

2 Related Work

Heuristic-based workflow scheduling methods rely on empirical rules or heuristics to quickly find good schedules, while guided random search methods, e.g. genetic algorithm, usually involve extensive trial and error to search for the best schedules, and thus generally require more scheduling time. Heuristic-based methods can be further divided into three categories: list-based, clustering-based, and duplication-based [1]. Among them, list-based methods are the most commonly used ones because of their simplicity and wide applicability.

List-based workflow scheduling methods consist of two major parts: task prioritization and processor allocation. Different list-based methods differ in the mechanisms adopted in these two parts. HEFT [1] is the most famous list-based workflow scheduling algorithm, which prioritizes tasks according to their bottom ranks [1] and allocates processors based on the earliest-finish-time principle. Many later algorithms made improvement based on it, such as a lookahead variant of HEFT [10] and PEFT [9]. IPPTS [2] is a most recent state-of-the-art work on list-based workflow scheduling, which was shown to outperform many previous methods, including HEFT [1], its lookahead variant [10], and PEFT [9]. At the task prioritization stage, IPPTS adopts a Predict Cost Matrix (PCM) to calculate each task's rank, and takes into consideration the task's out-degree in the workflow structure. In the processor allocation phase, IPPTS uses a principle called looking ahead earliest finish time, which is a bi-directional downward and upward approach to allocating a task and its heaviest successor onto the most optimistic processor.

List-based workflow scheduling blends well with a Markov decision process, i.e. a series of decisions on selecting a task to schedule and a processor to allocate. Therefore, in this paper we try to apply DRL to list-based workflow scheduling and compare its performance to previous typical algorithms, including HEFT [1] and IPPTS [2].

3 Deep Reinforcement Learning for Workflow Scheduling

Reinforcement Learning (RL) [7] is a machine learning approach that learns how to make optimal decisions by continuously interacting with an environment as shown in Fig. 2. Its core principle is to establish a state-action-reward model. The state represents the current environment, the action refers to the operation taken in that state, and the reward is the feedback provided by the environment. The goal of reinforcement learning is to maximize the cumulative sum of rewards.

To apply RL to workflow scheduling, we turn the optimization problem of workflow scheduling, i.e. minimization of execution makespan, into the decision making problem of a Markov decision process following the list-based scheduling methodology, where at each step the best selection of both the task to schedule and the processor to allocate should be made. Specifically, we try to solve the workflow scheduling problem based on deep reinforcement learning (DRL), which incorporates deep learning into RL, i.e. representing the learned policy as a neural network. As shown in Fig. 3, the DRL-based workflow scheduling is an iterative process, which continuously generates new training data by scheduling a set of workflows based on current version of a neural network model, and then improves the neural network model based on the collected training data. The iterative process will continues until the scheduling quality could not be improved further or the training time limit is reached.

In our DRL-based workflow scheduling method, the RL part is governed by the Policy Gradient (PG) algorithm [7], whose basic principle is to adjust the policy based on feedback. Specifically, when receiving positive rewards, the probability of corresponding actions are increased. On the other hand, the probability of corresponding actions will be decreased when receiving negative rewards. In PG, it is necessary to define a policy function, which generates an action based on the current state. Typically, a neural network is used to represent the policy function. In our method, a CNN model is adopted, which accepts current state, including workflow structure and current partial schedule, as input. In this paper, we assume that each workflow contains 60 tasks at most, and is to be scheduled onto five processors for execution. Therefore, the neural network input, i.e. current state, is represented by three matrices of dimension 60×60, 60×5, and 60×2, respectively. The first matrix contains the inter-task communication costs of a workflow, while the second matrix represents the computation costs of each task on different processors in a heterogeneous computing environment. The third matrix records current partial schedule during the scheduling process.

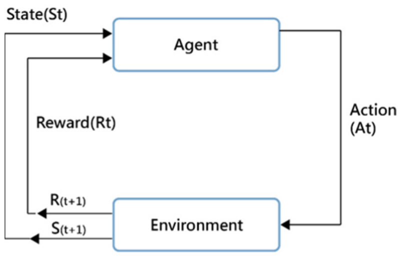

Fig. 2. Reinforcement learning process

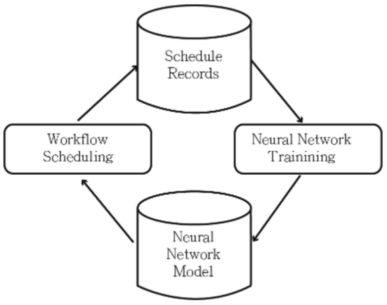

Fig. 3. DRL for workflow scheduling

The proposed DRL-based workflow scheduling method could be used in two different manners. In the first manner, the DRL-based method is used to continuously optimize a single workflow's schedule, similar to what the traditional guided random search methods [2] did. On the other hand, in the second manner, the DRL-based method is used to train a neural network model first based on a training set of workflows. Then, the neural network model is used as a policy function, i.e. returning how to select tasks and allocate processors, to schedule other workflows not in the training set in a list-based way. Both of these two manners will be evaluated in the following section.

4 Performance Evaluation

This section presents the performance evaluation of the proposed DRL-based workflow scheduling method conducted on the commonly used WorkflowSim [3] platform. The workflows used in the performance evaluation were randomly generated from WorkflowGenerator [4]. Each workflow might contain up to 60 tasks to be scheduled onto five processors. Since our DRL-based method follows the list-based scheduling methodology [1, 2], it was compared to the most famous list-based workflow scheduling method HEFT [1] and the state-of-the-art work IPPTS [2] in terms of execution makespan in the following performance evaluation. In the training process of our DRL-based method, the reward of each produced workflow schedule is assigned to the makespan improvement made by our method compared to IPPTS [2].

Tables 1 and 2 show the experimental results where the proposed DRL-based method was used to optimize a single workflow's makespan iteratively in speed-heterogeneous and speed-homogeneous environments, respectively. Heterogeneous environments were common for traditional distributed computing, while homogeneous environments could be created easily on modern cloud computing platforms. The proposed DRL-based method could produce significantly shorter workflow schedules than HEFT [1] and IPPTS [2] in both heterogeneous and homogeneous environments.

For Tables 3 and 4, a neural network model was trained with 13 randomly generated workflows using the proposed DRL-based method first. Then, the neural network model was used to schedule five new workflows in the list-based manner efficiently. Table 3 shows that the proposed DRL-based method achieves the shortest makespan among the three scheduling methods in four of the five workflows in a heterogeneous environment, indicating promising generalization capability. However, for a homogeneous environment, Table 4 shows that current experimental result is not quite good, where only one of the five workflows could achieve the best performance under the proposed DRL-based method. More training data might be helpful to improve the performance further since current training set contains only 13 workflows which might not be enough for achieving good performance.

Table 1. Optimizing a single workflow's makespan in a heterogeneous environment

Workflows	DRL makespan	IPPTS makespan	HEFT makespan
1	**8700.72**	13974.85	11436.44
2	**8877.99**	11747.95	17260.26

Table 2. Optimizing a single workflow's makespan in a homogeneous environment

Workflows	DRL makespan	IPPTS makespan	HEFT makespan
1	**12892.25**	15747.45	16385.63
2	**12624.57**	14123.23	16252.56

Table 3. DRL-based workflow scheduling in a heterogeneous environment

Workflows	DRL makespan	IPPTS makespan	HEFT makespan
1	**10800.56**	12301.70	10983.5
2	**10971.97**	14887.44	22393.05
3	11564.96	**11324.79**	14287.08
4	**11750.57**	12054.5	12565.47
5	**11204.16**	12149.25	11827.92

Table 4. DRL-based workflow scheduling in a homogeneous environment

Workflows	DRL makespan	IPPTS makespan	HEFT makespan
1	15099.79	**14434.20**	14619.49
2	**15133.12**	15816.7	16450.7
3	18869.33	**15415**	19474.79
4	18382.52	**18028.24**	18691.32
5	19128.88	**13774.70**	15790.1

5 Conclusions and Future Work

This paper presents a new workflow scheduling method based on deep reinforcement learning. The proposed method could be applied in two manners: (1) iteratively optimizing a single workflow's makespan from scratch, (2) training a neural network model to efficiently schedule new workflows never seen before following the list-based scheduling methodology without relying on human heuristics. Experimental results show significant performance improvement in both manners, compared to the most famous and state-of-the-art list-based workflow scheduling methods, i.e. HEFT [1] and IPPTS [2].

The preliminary results presented in this paper indicates a promising future research direction for workflow scheduling based on deep reinforcement learning. In addition, more research efforts are needed to investigate some issues and improve the scheduling quality further. For example, it will be interesting to find out why a heterogeneous environment could benefit more from the proposed DRL-based method as shown in Tables 3 and 4. Moreover, all of the adopted neural network model, reinforcement learning algorithm, state representation, and reward assignment method, involved in the proposed DRL-based method, deserve more research work to improve their designs for further performance improvement in workflow scheduling.

References

1. Topcuoglu, H., Hariri and Min-You Wu, S.: Performance-effective and low-complexity task scheduling for heterogeneous computing. IEEE Trans.Parall. Distrib. Syst.**13**(3), 260–274 (2002)
2. Djigal, H., Feng, J., Lu, J., Ge, J.: IPPTS: an efficient algorithm for scientific workflow scheduling in heterogeneous computing systems. IEEE Trans. Parallel Distrib. Syst. **32**(5), 1057–1071 (2021)
3. Chen, W., Deelman, E.: WorkflowSim: a toolkit for simulating scientific workflows in distributed environments. In: IEEE 8th International Conference on E-Science, pp. 1–8 (2012)
4. Juve, G., et al.: Workflow Generator. https://github.com/pegasus-isi/WorkflowGenerator
5. Sinnen, O.: Task Scheduling for Parallel Systems, John Wiley (2007)
6. Silver, D., Hubert, T., Schrittwieser, J.: Mastering chess and shogi by self-play with a general reinforcement learning algorithm. arXiv:1712.01815v1 [cs.AI] (2017)
7. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, MIT Press, (2018)
8. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning, (MIT Press, 2016)

9. Arabnejad, H., Barbosa J.G.: List scheduling algorithm for heterogeneous systems by an optimistic cost table. IEEE Trans. Parall. Distrib. Syst. **25**(3), 682–694 (2013)
10. Bittencourt, L.F., Sakellariou, R., Madeira, E.R.M.: DAG scheduling using a lookahead variant of the heterogeneous earliest finish time algorithm. In: 18th Euromicro Conference on Parallel, Distributed and Network-based Processing, pp. 27–34 (2010)