DeepSched: A Deep Representation Of Scheduling Policies For Heterogeneous Distributed Systems

Mohamed Shawky | Remonda Talaat | Mahmoud Adas | Evram Youssef
SEC 2, B.N 16 | SEC 1, B.N 20 | SEC 2, B.N 21 | SEC 1, B.N 9
{mohamedshawky911, remondatalaat21, mido3ds, evramyousef}@gmail.com

***Abstract*—To be conducted, once the results are complete.**

I. Introduction

THE continuous evolution of computing power and growth of widely-distributed applications and *Internet of Things* (IoT) have driven our needs for better resource allocation methods. The performance of any system critically depends on the algorithms used to schedule tasks on its resources. These systems contain limited resources, that should be allocated properly, in order to perform the required tasks with maximum efficiency avoiding starvation and resource exhaustion. The general objective of any scheduling technique is to get the best possible performance with a reasonable resource utilization.

Recently, the heterogeneity of the computational systems has increased tremendously, due to the nature of the applications and the great advances in IoT and distributed systems. A *heterogeneous system* can be defined as a range of resources, different in underlying architecture, targeting different types of computational tasks. The usage of a heterogeneous system has proven to be very efficient in increasing the system performance and reducing the overall power consumption. Recent studies [1] show that *heterogeneous system* can outperform the best homogeneous systems by as much as 21%, with 23% energy savings and a reduction of 32% in Energy Delay Product. These numbers can be improved even more with better task scheduling and resource allocation methods.

However, this comes with the problem of complex task scheduling. The task scheduling problem for a heterogeneous computing system is more complex than that for a homogeneous system, because of the different execution rates of processors and different communication rates among different processors, which add extra layers of complexity to the problem.

The parameters of such problem make it very suitable for various optimization techniques. Recursive optimization techniques are very efficient in solving many optimization problem. Previous work has investigated the usage of techniques such as *genetic algorithms* [2] to solve the task scheduling problem. Also, the usage of a progressive decision making techniques such as *Reinforcement Learning* [3] for task scheduling in heterogeneous systems has been investigated.

In this paper, we investigate the ability of advanced predictive models to learn a representation of static local task scheduling problem. This learned representation will be used to approximate the scheduling performance of other scheduling techniques. *Neural Networks* are our best candidate for this task. Neural networks are being used in many fields replacing traditional methods, such as image classification [4], object detection [5], neural language modelling [5], image generation [6] and others. They have proven to be more efficient than other traditional methods. The complexity of the neural networks comes in the development phases, where the network is designed and trained. However, in inference phase, neural networks can achieve very accurate results at high speed. Recent work has been done on optimizing neural network inference through various techniques, such as quantization [7] and pruning [8].

The recent advances in *Deep Learning* research is considered in designing and training our scheduling network. *Heterogeneous Earliest Time First* (HEFT) [9] is used as a target, where the network learns to approximate its performance. Also, we include a genetic algorithm experiment for the same problem for comparison. We will discuss our approach to the problem and its limitations.

II. Related Work

Since task scheduling in heterogeneous systems is a wide problem, a lot of research has been conducted in this area. Various techniques have been introduced for

the problem. These techniques are categorized based on multiple aspects [10] and some of which are adapted from task scheduling methods for homogeneous systems.

A. Scheduling Categories

In this work, we focus on three basic categorization of heterogeneous task scheduling methods.

1) Application-Specific vs. System-Specific: Heterogeneous scheduling methods can be divided into two main categories based on their target metric.
Application-Specific scheduling targets task execution speed (performance), where scheduling decisions are determined based on many parameters including application performance, task inter-dependency and the availability of resources. *System-Specific* targets resource utilization, where scheduling decisions are based on the percentage of time a resource is available or busy.

2) Global vs. Local: Also, heterogeneous scheduling methods can be divided into global and local scheduling. *Global* scheduling can migrate the tasks from one processor to another, meanwhile *local* scheduling can't migrate, so once a task is assigned to a processor, it stays in its queue.

3) Static vs. Dynamic: Finally, heterogeneous scheduling methods can be divided into static and dynamic scheduling. In *static* scheduling, the tasks required to be scheduled are defined before scheduling. However, in *dynamic* scheduling, new tasks can be introduced during the scheduling process

B. Heterogeneous Earliest Time First (HEFT)

HEFT [9] is one of the most well-established algorithms for task scheduling in heterogeneous systems. It's a local static scheduling algorithms, which deals with a *Direct Acyclic Graph* (DAG) of tasks displaying the dependencies between different processes. Each task has a running time for each different machine and a time for communicating the results to children tasks. A wide range of algorithms [10] follows HEFT further improving the scheduling performance.

C. Optimization Algorithms

Recent research has been conducted to use optimization algorithms and predictive models to schedule tasks on heterogeneous system, since these algorithms have proven to be very efficient in solving complex computational problems.

1) Genetic Algorithms (GA): GA [2] can be used to efficiently schedule tasks in different systems. It's used for static scheduling, where a fixed population of tasks are scheduled recursively on multiple processors, based on a specific fitness function.

2) Reinforcement Learning (RL): Also, RL [3] has been a strong candidate for the problem, as the scheduling problem can be formulated as an *RL* problem, solved by various *RL* techniques. RL techniques are effective in solving progressive decision making problems, so they are more likely to be used in dynamic scheduling.

D. Neural Networks

Recent advances in *Deep Learning* have enabled neural networks to dominate many fields, such as computer vision, natural language processing and others. Neural networks are able to perform perception tasks at a human-level accuracy. Also, neural networks are very efficient in function approximation and representation learning. These properties make neural networks a perfect choice for massive scale automation of many tasks. Some studies even investigated the usage of *Artificial Neural Networks* (ANN)[11] in task scheduling in heterogeneous systems. Most of these studies focus on the usage of *Hopfield Nets* [12] and *Inhibitor Neurons* [11].

In this paper, we make use of recent advances in *Deep Learning* and build a deep representation of the scheduling problem and use it to approximate the performance of heterogeneous scheduling methods. *Recurrent Neural Networks*(RNNs) [13] are used for this purpose, as they have the ability to learn complex temporal information from a time-series data, which is exactly our case. Other network operations to be used, too.

III. Methodology

A. Motivation

Heterogeneous task scheduling is a relatively new concept, which lacks developers support. It's an active area of research, where many techniques are being developed through time. Being able to achieve the best possible performance on the given tasks, while keeping good utilization of resources is a tedious problem in heterogeneous systems. Aside from being accurate, the scheduling methods have to be fast in its execution, also they should not consume the system resources.

Heterogeneous scheduling methods are very computationally expensive. However, neural networks are relatively fast in inference and research is being conducted on optimizing inference even more. So, if a neural network can approximate the performance of current scheduling method, it can replace this method in production.

In this work, we study the ability of neural networks to approximate some heuristic methods and explore its accuracy and execution time against heuristic and approximated methods.

B. Formulation

First, let us specify the main target of this work. As the task scheduling in heterogeneous distributed system is a vast field with lots of settings and algorithms [10], we choose our scheduling setting to be local and static. The goal of this work is to explore the ability of neural networks to approximate the performance of some well-established scheduling algorithms. In our setting, the algorithms are provided a list of tasks, each has its own execution time on different machines and a set of heterogeneous machines, on which the tasks are executed. We compare the results neural networks to some other heuristic and approximate methods.

Although, we choose local static scheduling, we assume that the neural approximation methods can work in other settings as well.

C. Baseline

Two baseline methods are considered. We choose *Heterogeneous Earliest Finish Time* (HEFT) [9] to be our heuristic baseline, as it's one of the widely-used scheduling algorithms in heterogeneous systems. Also, genetic algorithms [2] is provided as an approximate baseline.

1) Heterogeneous Earliest Time First (HEFT): HEFT [9] is an old and well-established algorithm for scheduling in heterogeneous systems. It has been used as a baseline in research due to its near-optimal results. Basically, it chooses some process to run on some machine, if the former has the earliest finish time, with considering the special nature of the heterogeneous system. However, HEFT isn't practical for real-world systems, as no-one can know in advance the finish time of a process

2) Genetic Algorithms: Genetic algorithms [2] have proven to be very efficient in solving some of the hardest optimization problem. Previous work on the use of genetic algorithms in task scheduling [2] is adapted to the heterogeneous setting by adding parameters for execution time of different tasks on different machines.

The targeted *fitness function* will be the average waiting time W_t for all tasks. The fitness function for a specific schedule *S* is given by:

$$\mathcal{F}itness(S) = \frac{\sum_{i=1}^{N} W_{ti}}{N} \tag{1}$$

D. Neural Networks (Deep Learning)

The main novel idea of this work is the use of *neural networks* for the scheduling problem. Neural Networks are known to be very good at learning representations and function approximation. In this work, we propose an architecture that can be used to reach a good approximated solution for the scheduling problem. *Offline Optimization* of neural networks is to be used, in order to be able to train the network on scheduling samples and then use the trained network to do inference. We avoid using *online optimization*, as it's slow and we mainly care about execution time of the scheduling method.

Previous work tried to use neural networks to solve scheduling problem through *Hopfield Nets* [12] and *Inhibitor Neurons* [11]. However, we try to make use of the recent advances in *Deep Learning* to redefine the scheduling problem and solve it.

As mentioned, The provided input is a list of tasks with its information and a set of machines. We want the network to learn an implicit representation of the given tasks and machines. Then, the network learn the mapping between the learned representation and the desired schedule. In other words, the target of the network is to take two input list of information of both tasks and machines and then output for each task, the machine to be executed on, the actual start time and the actual finish time.

1) Architecture: Our architecture consists of two encoders and one decoder, as shown in Fig.1.
The first encoder is a fully convolutional network of *1D* convolutional layers, to which is machines specifications are passed. This encoder learns a representation for the machines to be able to use it later in producing the outputs.

The second encoder is an *Recurrent Neural Network* (RNN) [13], to which the tasks are passed one by one ordered by the arrival time. This network learns a representation of the tasks preserving the arrival order. RNN is chosen due to its proven effectiveness in dealing with sequential data, as they can learn complex temporal information from time sequences.

The two representations are then stacked and passed to two modules. A classification module which states the machine on which a specific task is executed. A regression module that identifies the actual start time and the actual finish time. Mainly, the two modules consist of *1D* convolutional layers.

2) Objective Function: The cost function defined for this architecture is a sum of two criteria. Categorical cross entropy *(CE)* on the predicted machine to run a specific job M_p and mean square error *(MSE)* on the predicted actual start time AST_p and the predicted actual finish time AFT_p. So it can be summarized as follows:

$$\mathcal{L}oss(L) = CE(M_p, M_t) + MSE(AT_p, AT_t) \tag{2}$$

where M_p is the predicted machine, M_t is the ground truth machine, AT_p is the predicted actual time range and AT_t is the ground truth actual time range.

3) Downsides: The main downside of the proposed solution is that we have to know the maximum number of machines and the maximum number of tasks to be able to define the network. These numbers are used to define the network and loss function dimensions.

IV. Experimental Setup

V. Discussion and Results

VI. Conclusion

References

[1] Ashish Venkat and Dean M Tullsen. Harnessing isa diversity: Design of a heterogeneous-isa chip multiprocessor. In *2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*, pages 121–132. IEEE, 2014.

[2] Ranjeet Kumar, Dr.Rakesh, Er.Rajiv, Er Gill, and Ashwani Kaushik. Genetic algorithm approach to operating system process scheduling problem. *International Journal of Engineering Science and Technology*, 2, 09 2010.

[3] Alexandru Iulian Orhean, Florin Pop, and Ioan Raicu. New scheduling approach using reinforcement learning for heterogeneous distributed systems. *Journal of Parallel and Distributed Computing*, 117:292 – 302, 2018.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017.

[5] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2015.

[6] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan, 2019.

[7] Yoni Choukroun, Eli Kravchik, Fan Yang, and Pavel Kisilev. Low-bit quantization of neural networks for efficient inference, 2019.

[8] Seul-Ki Yeom, Philipp Seegerer, Sebastian Lapuschkin, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Pruning by explaining: A novel criterion for deep neural network pruning, 2019.

[9] H. Topcuoglu, S. Hariri, and Min-You Wu. Performance-effective and low-complexity task scheduling for heterogeneous computing. *IEEE Transactions on Parallel and Distributed Systems*, 13(3):260–274, March 2002.

[10] Dr. Mamta Padole and Ankit Shah. *Comparative Study of Scheduling Algorithms in Heterogeneous Distributed Computing Systems*, pages 111–122. 01 2018.

[11] Daniel Chillet, Antoine Eiche, Sébastien Pillement, and Olivier Sentieys. Real-time scheduling on heterogeneous system-on-chip architectures using an optimised artificial neural network. *Journal of Systems Architecture - Embedded Systems Design*, 57:340–353, 04 2011.

[12] Saratha Sathasivam and Wan Ahmad Tajuddin Wan Abdullah. Logic learning in hopfield networks, 2008.

[13] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling, 2014.

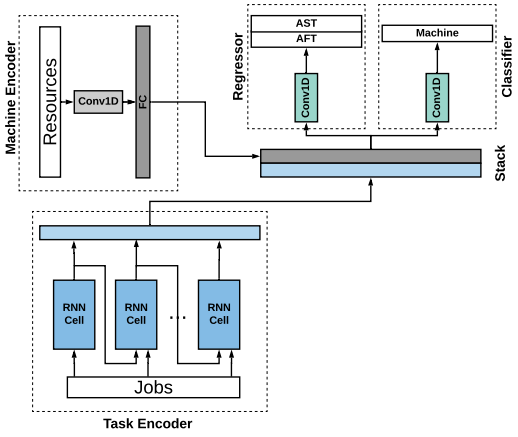

Fig. 1. The architecture of the proposed neural network for distributed heterogeneous systems scheduling