Optimizing 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*—This is the second report of the research project. This report discusses our methodology for the proposed problem and our potential experiments.**

I. Introduction

THE scheduling and resource allocation policies for a distributed system are very hard to achieve. This becomes even harder when the system is heterogeneous, where you have different architerctures of processors. The problem is classified as *NP Hard* and it's [1] an open problem for research. In this work we will discuss several possible solutions to the problem and introduce yet a new idea that could potentially work better.

II. Formulation

To simulate the real heterogeneous systems, we have to create a simulation stub, where you have some defined machines with their specifications and a list of jobs that need to be executed on such machines. We will work with *Global Static* scheduling scheme and the goal is to assign the jobs to the machines at each time step to maximize overall performance.

Although resources [2] utilization isn't our main concern, we may include it in our objective.

III. Methodology

A. Baseline

We will consider *Heterogeneous Earliest Finish Time* (HEFT) as our baseline and will compare its performance against our work.

HEFT is an old and well-established algorithm for scheduling in heterogeneous systems. Most research work compares their results with it because of its near-optimality. 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. HEFT isn't practical for real-world systems, as no-one can know in advance the finish time of a process.

B. Neural Networks (Deep Learning)

The main novel idea of this work will be the usage 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.

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

C. Genetic Algorithms

We will use genetic algorithms as a recursive optimization to solve our problem. The targeted *fitness function* will be the average waiting time for all jobs.

Genetic algorithms have proven to be very efficient in solving some of the hardest optimization problem. Few similar research projects have been conducts in using genetic algorithms. We hope we can produce better results if the simulation and the algorithm were both well-tuned.

D. Reinforcement Learning

Another possible solution is *Reinforcement Learning* (RL), which is one of the most dominant fields in optimization and decision making problems.

The main advantage of RL is that it can act in a very dynamic environment, so it can adapt, for instance, to priority changes and deadlocks. Moreover, we can add extra layers of complexity to our problem, such as: adding a status to each machine which can change with time or machine-specific tasks. However, this comes at a cost of complicated and unstable environment.

If RL is to be used, one can consider using *Q-Learning* to learn a scheduling policy using an environment of some machines.

[1] to the time of writing this paper, hopefully.
[2] the other resources of machine other than CPUs.

1) Architecture: Our architecture consists of two encoders and one decoder, as shown in Fig.1.

The first encoder is a simple feedforward fully connected network, 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)*, to which the jobs are passed one by one ordered by the arrival time. This network learns a representation of the jobs preserving the arrival order.

The representation of the jobs is later passed to an RNN decoder, which produces N timesteps, at each step the processes currently running on each machine. The output is supported by attention modules to be able to take the machines representation into account when distributing the jobs.

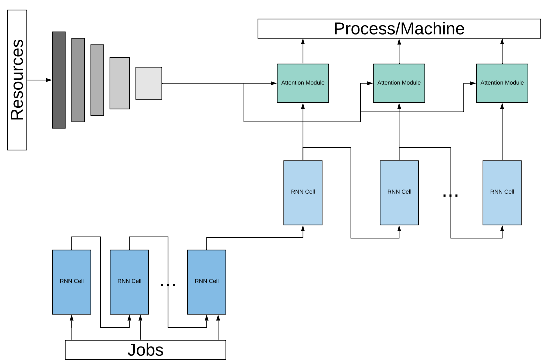

Fig. 1. The architecture of the proposed neural network for Heterogeneous systems scheduling.

2) Objective Function: The cost function defined for this problem will be a simple *mean square error* (MSE) between the output *(processes/machine)* and some optimally solved examples.

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 jobs to be able to define the network. However, the network scheduling inference can be faster than many algorithms.

IV. Experiments

This work will many focus of performance, specifically average waiting time and average turnaround time.

- Measure the performance of *HEFT* on our simulation stub.
- Experiment with Genetic Algorithms to explore possible improvements.
- Try the proposed neural network and explore the results and possible improvements.
- Experiment with *RL* on our simulation stub. *[OPTIONAL]*

V. References

[1] Kumar, Ranjeet & Dr.Rakesh, & Er.Rajiv, & Gill, Er & Kaushik, Ashwani. (2010). Genetic Algorithm approach to Operating system process scheduling problem. International Journal of Engineering Science and Technology.

- Why this paper: To explore the formulation of a scheduling problem using *Genetic Algorithms*.
- Paper research problem: Operating systems process scheduling.
- Paper goal: To improve scheduling with *Genetic Algorithms*.
- Conclusion: *Genetic Algorithms* has a potential in such problems.

[2] Alexandru Iulian Orhean, Florin Pop, Ioan Raicu, New scheduling approach using reinforcement learning for heterogeneous distributed systems, Journal of Parallel and Distributed Computing.

- Why this paper: To understand the scheduling problem as an *RL* model.
- Paper research problem: *RL* modelling for scheduling.
- Paper goal: Explore all layers of complexity of a scheduling problem using *RL*.
- Conclusion: *RL* can be very useful in dynamic scheduling.

[3] Chillet, Daniel & Eiche, Antoine & Pillement, Sébastien & Sentieys, Olivier. (2011). 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. 10.1016/j.sysarc.2011.01.004.

- Why this paper: To explore the previous work made about the usage of neural networks in scheduling tasks.
- Paper research problem: Process scheduling with *ANNs*.
- Paper goal: To improve the usage of neural networks in scheduling.
- Conclusion: Neural networks can be a good candidate for scheduling.