Self-accelerating processing workflows
(GPU vs CPU with insights to optimize)

- A framework that predicts on which a problem is the best to execute -

1. Introduction

Utilizing resources properly improves the efficiency and performance of a computer system. The GPU was originally invented for graphical processing where a huge number of similar calculations needed to be done parallelly. With the time, it was identified and started to use the GPU general-purpose processing, the GPGPU evolved. The GPU is not good to be assigned with all tasks all the time though it has a higher performance. Because the GPU is busy sometimes for graphical processing purposes, still some problems are limited to the CPU and some problems are less efficient to be executed in GPU compared to the CPU. Our work is to design a function that determines whether it is efficient to execute a given specified problem in a GPU or a CPU. So, the programmer can leave it to the system to select the mode of operation in scripting for a given kind of problem. Some research has already been conducted in this area.

2. Problem Statement

The resources of the GPU and the CPU are wasted and not utilized efficiently if it a problem executed in the wrong one. Also, the programmer sometimes needs to write separate programs for both the CPU and the GPU in two projects which wastes their time too. There were already a few related kinds of research conducted previously. Programmers do not have a way to do the selection dynamically and execute in two different modes for the CPU and GPU for a specific processing problem.

3. Research Objectives

A framework that essentially contains a function that determines whether a problem of which context given as arguments to the function should be executed in the GPU or CPU to utilize its resources and use them efficiently.

4. Literature Review

The main CPU and GPU designing goal is to achieve higher performance in computing. A CPU consists of few cores with the ALUs and the CUs while a GPU consists of thousands of cores and has access to a huge array of memory which enhances parallel processing. The GPUs are used for general purposes (GPGPU) computing such as grid computing, machine learning, data mining, cryptography (neural networks), bioanalysis molecular dynamics. As authors of the paper “CPU - GPU Processing” [1] states, GPGPU vector processing is not the solution to everything and CPU still do much better than GPU for certain problems.

The results of the experiments, mentioned in Anna Syberfeldt and Tom Ekblom s’ paper [2], showed that the amount of data and the number of available parallel instances have a decisive influence on which platform is more efficient. With larger amounts of data, the CPU is more efficient when a limited number of parallel instances are available, as using the GPU is associated with a significant overhead that negates the possible improvements gained by parallelization. With small amounts of data, the CPU is more efficient than the GPU, but only up to a relatively low number of parallel instances.

In order to get benefit out of accelerators such as GPUs, one has to find computationally expensive (calculation dominated) parts of the program which can run independently and separate them into so-called kernels. These kernels are then executed by the GPU. This process is not always possible. Some programs have very little computation and a lot of copying memory around. These applications are bandwidth limited (data dominated) and will not perform well on external accelerators compared to the CPU. [3]

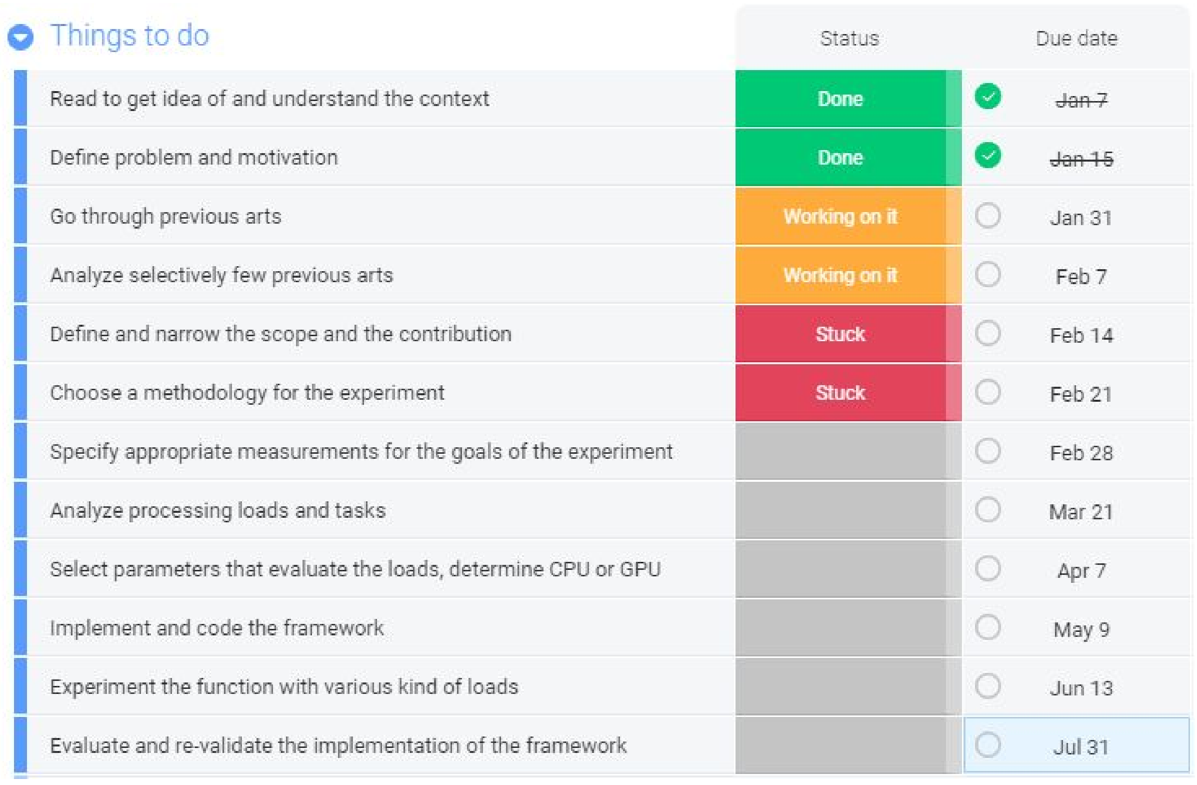
A system composed of computing units, with different characteristics and strategies for data processing is usually called a hybrid system (H-system) due to the presence of heterogeneous computing units. [4] Usually the decision that whether a problem to be executed in the GPU or the CPU is hard-coded by the programmer based on the problem to be solved. This is resulting in inappropriate or inefficient scheduling of jobs and processes and to an unoptimized use of the hardware resources.

5. Methodology

This will essentially be an experimental methodology since the research is based on the results and outcomes of the predictions returned by the function that is going to be implemented. It will be analyzed the method of execution, the execution flow of processing in the CPU and the GPU and what kind of problems will be more efficient to be executed in the CPU and the GPUs respectively. Research needs to be conducted about the problems processed in a system which will give us knowledge about the dimensions and the attributes resolving the weights of the problems. The parameters accepted by the function which defines the complexity and dimension information of the problems will be extracted with the outcomes and conclusions of the results and further research will be conducted with regards to the parameters.

Also, the framework can be implemented in various ways and has not been selected yet. It could be determined by keeping track of the history such as with few past success counts of the predictions, logging the successes of the predictions and using machine learning or merely just predicting it with the weight and the type of the problems. However, there should be resource gain from the prediction of the framework and it will be useless otherwise.

6. Research Timeline



Things to do	Status		Due date
Read to get idea of and understand the context	Done	✓	~~Jan 7~~
Define problem and motivation	Done	✓	~~Jan 15~~
Go through previous arts	Working on it	○	Jan 31
Analyze selectively few previous arts	Working on it	○	Feb 7
Define and narrow the scope and the contribution	Stuck	○	Feb 14
Choose a methodology for the experiment	Stuck	○	Feb 21
Specify appropriate measurements for the goals of the experiment		○	Feb 28
Analyze processing loads and tasks		○	Mar 21
Select parameters that evaluate the loads, determine CPU or GPU		○	Apr 7
Implement and code the framework		○	May 9
Experiment the function with various kind of loads		○	Jun 13
Evaluate and re-validate the implementation of the framework		○	Jul 31

7. Conclusion/Summary

The research is to evaluate a processing problem based on their characteristic provided by programmers to predict whether the problem should be executed in the CPU or GPU to utilize its resources efficiently. The outcome of the research will be a framework that contains few functions to be used by the programmers. It has also been planned to consider the previous prediction history to make succeeding predictions further accurate. The way how the previous history is stored and analyzed will be decided later.

8. References

[1] Z. Memon, F. Samad, Z. Awan, A. Aziz and S. Siddiqi, "CPU-GPU Processing", IJCSNS International Journal of Computer Science and Network Security, Vol.17, No.9, September 2017. [Accessed on: 30- Dec- 2019] [Online]. http://paper.ijcsns.org/07_book/201709/20170924.pdf

[2] A. Syberfeldt and T. Ekblom, "A comparative evaluation of the GPU vs. The CPU for parallelization of evolutionary algorithms through multiple independent runs", International Journal of Computer Science & Information Technology (IJCSIT) Vol 9, No 3, June 2017. [Accessed: 31- Dec- 2019] [Online]. http://aircconline.com/ijcsit/V9N3/9317ijcsit01.pdf

[3] S. Brinkmann (2020). "ResearchGate" [Accessed:21 Jan. 2020] [Online]. https://www.researchgate.net/post/Anyone_have_experience_in_programming_CPU_GPU_What_is_the_real_benefit_in_moving_everything_possible_from_CPU_to_GPU_programming

[4] Vella, F., Neri, I., Gervasi, O. and Tasso, S. (2012). A Simulation Framework for Scheduling Performance Evaluation on CPU-GPU Heterogeneous System. Computational Science and Its Applications – ICCSA 2012, pp.457-469. [Accessed:21 Jan. 2020] [Online]. https://link.springer.com/chapter/10.1007/978-3-642-31128-4_34