0.1 parallel computing

0.1.1 What is parallel computing

The problem when facing large calculations is that they require a lot of computing power and time. Performing these calculations can be done in different types of computation but the most common ones are serial and parallel computation. Serial computing means, you have one compute unit (e.g. a single core CPU) available to deal with all the calculations that have to be done on a certain set of data. The problem will be broken into multiple smaller subproblems that will be solved by a certain instruction. The single compute unit has to perform the instruction on every subproblem to solve the main problem as shown in figure 1.

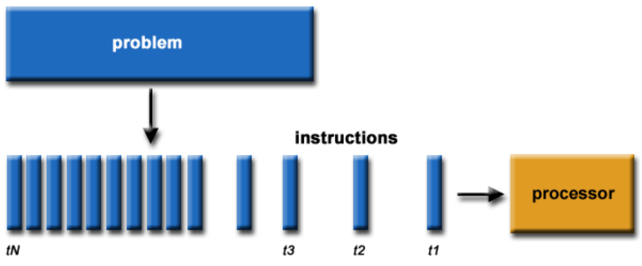


Figure 1: Workflow serial computing

Parallel computing, on the other hand, is the simultaneous use of multiple compute units (CPU+GPU), or a compute resource, to solve a computational problem. We break apart the main problem in smaller subproblems as we did with serial computing. Each part is further broken down to a series of instructions again.Now, since there are multiple compute units, we can distribute the subproblems among all these compute units. Every unit can now perform the instruction on their given subproblems simultaneously as shown in figure 2. One compute resource can constist of multiple compute units, for example it can include multiple processors or it's just a number of computers connected by a network.
With multiple compute units for one task, we will shorten the completion time aswell have potential cost savings. It also allows us to solve larger/complex problems since a single computer could suffer from limited memory. And last, we are able to access non-local resources in a network that wouldn't be accessable from a local computer.
It is easy to conclude that the concept of parallel computing was to have a more efficient way to handle with large sets of data such as images or simulations that involve multiple parameters. It's also easier to handle with complex data for example in algorithms.

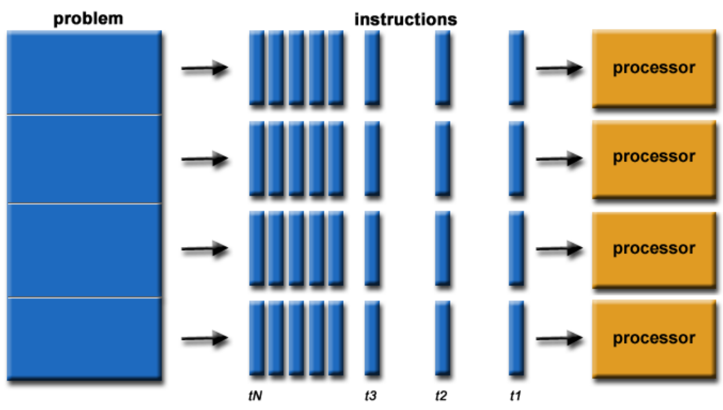


Figure 2: Workflow parallel computing

0.1.2 Classification of parallel computing

Parallel computing systems can be seperated into different classes. According to Flynn's taxonomy we can roughly place any of these systems in one of the four classes he defined. The classifications are determined by two factors: instruction stream and data stream which both have two possible states being single or multiple. Figure 3 represents a matrix of the four possible classes from the Flynn's taxonomy.

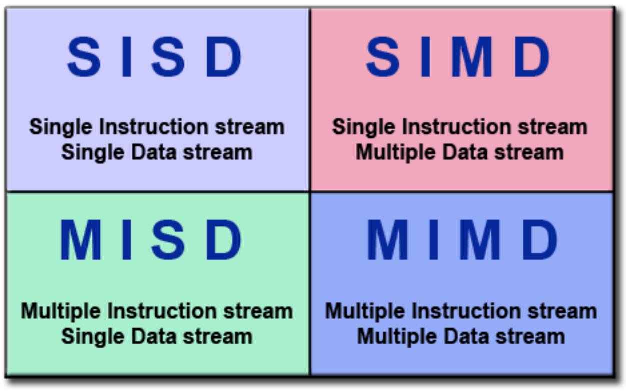

S I S D
Single Instruction stream Single Data stream
S I M D
Single Instruction stream Multiple Data stream
M I S D
Multiple Instruction stream Single Data stream
M I M D
Multiple Instruction stream Multiple Data stream

Figure 3: Four possible classifications according to Flynn's Taxonomy

- SISD

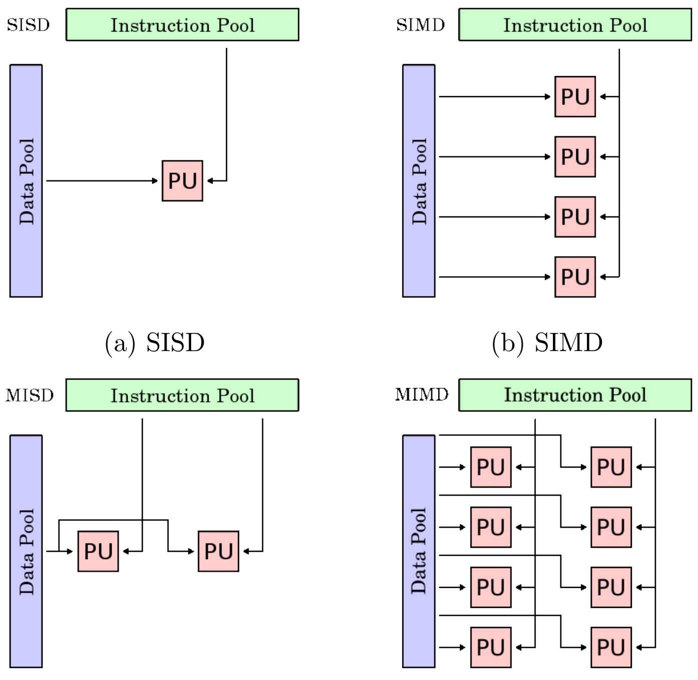
The SISD class will have a single processor executing a single data stream to operate on data stored in a single memory (figure 4a). This means that a parallel compute system cannot be classified as an SISD sytem, but Flynn's taxonomy wasn't made for just classifying parallel compute systems. It's usually old computers and other older compute units that can be classified as SISD.

- SIMD

In this class, the data is disctributed amongst multiple processors who all execute the same instruction (figure 4b). Since we have access to multiple compute units, parallel computing can be categorized in this class. Furthermore, this is the class where we can categorize our subject of the thesis in since we have a large set of data divided over multiple data streams (MD) and only one instruction stream (SI) since all processing units will perform the same instruction on the data. The SIMD class contains most modern computers, particularly those with a graphics processor unit (GPU).

- MISD

- MIMD



(a) SISD (b) SIMD

(c) MISD (d) MIMD

Figure 4: Architecture classes from Flynn's taxonomy

0.1.3 Amdahl's law and Gustafson's law

- Amdahl's Law

Amdahl's Law

0.1.4 OpenCL

Bibliography