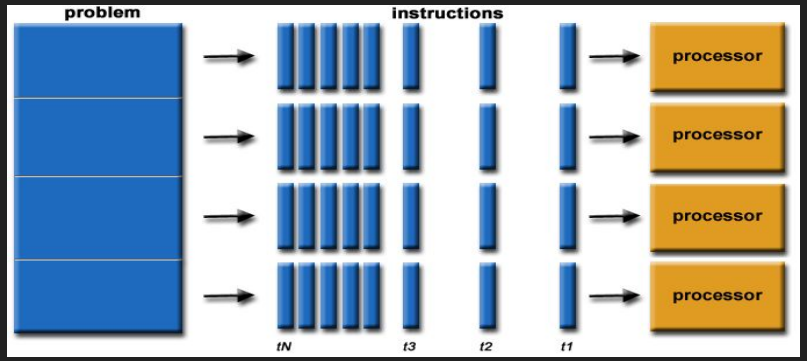

Parallel Computation
By: Zach Nguyen, Matthieu Rouxel, Yuzhou Shen

What is Parallel Computation?

Definition: when a computer uses more than one or multiple processor units to make more than one computations simultaneously. In other words, problems are broken down into smaller instructions and then these sub-problems are all solved concurrently working together to solve the bigger problem.



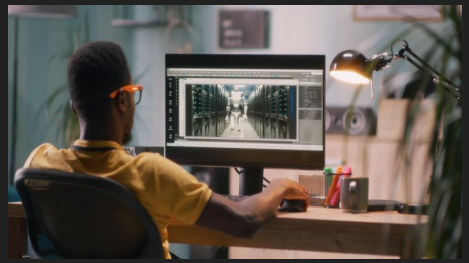
Parallel Computation plays an important role in many fields such as:

1. Artificial Intelligence (AI)

2. Financial and Weather modeling

3. Video processing/Animation in the entertainment industry

4. Oil and Gas data processing



Boolean Circuits

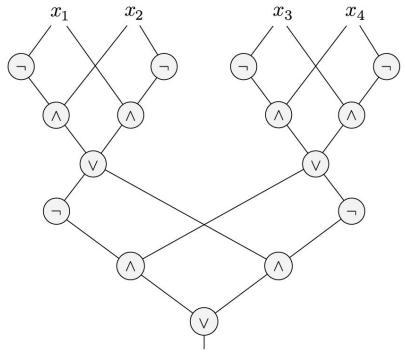
Boolean circuits are mathematical models that represents of boolean functions which operate on binary inputs and give us binary outputs.

Key aspects:

- Boolean functions
- Logic gates

EXAMPLE 9.25

The n-input ***parity function*** $parity_n : \{0,1\}^n \longrightarrow \{0,1\}$ outputs 1 if an odd number of 1s appear in the input variables. The circuit in Figure 9.26 computes $parity_4$, the parity function on 4 variables.



Types of Parallelism

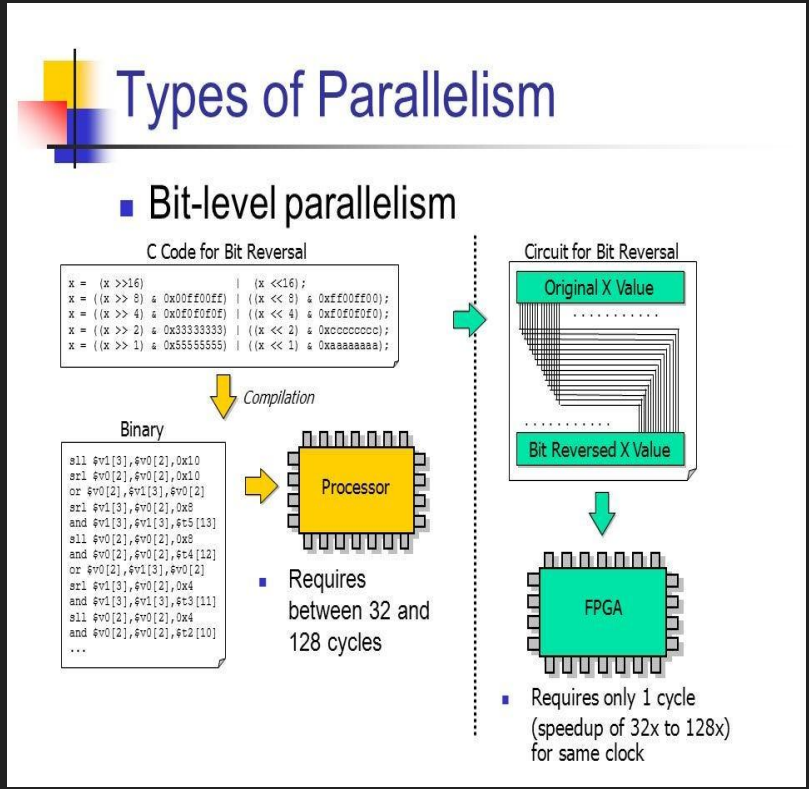
.There are many different types of parallelism but we will cover three main ones:

1. Bit-level parallelism
2. Task parallelism
3. Data-level parallelism

Bit-level parallelism

. Bit-level parallelism essentially means doubling the word size, by doing so, it will give the processor of the computer less instructions to carry out, therefore, greatly increasing the speed of the computation.

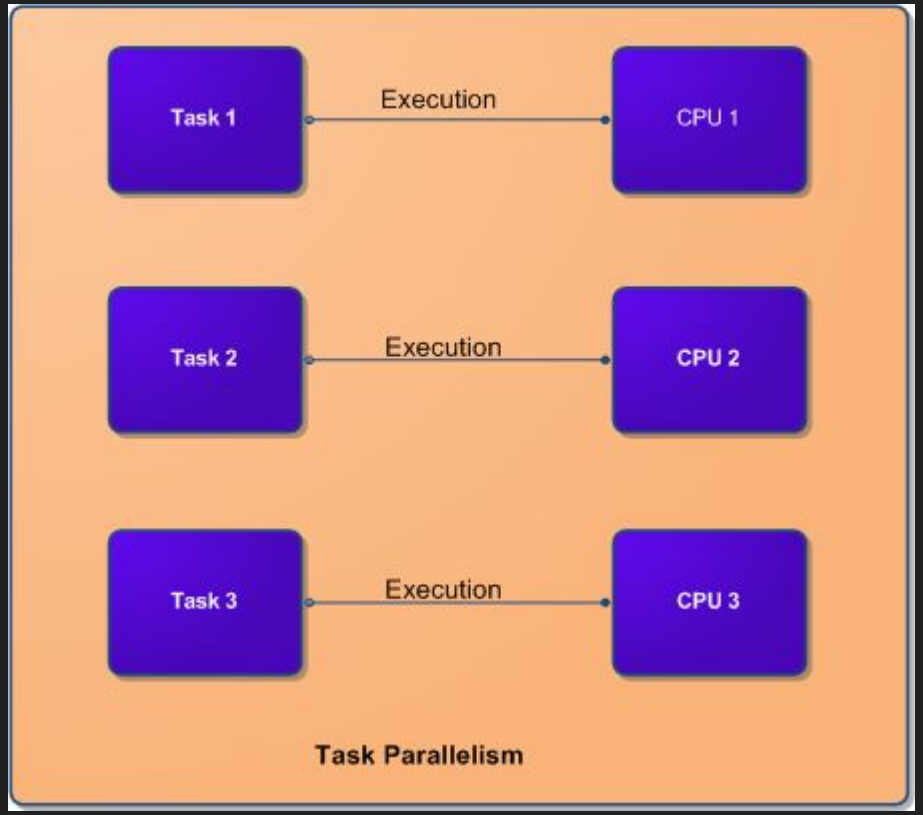
. By using bit-level parallelism, we lower the time of computing a specific instruction and increase the efficiency of the machine.



Task parallelism

. Task parallelism involves breaking a big task down into smaller, more manageable tasks. Consequently, it gives different processors these smaller tasks and allows them to compute the executions at the same time.

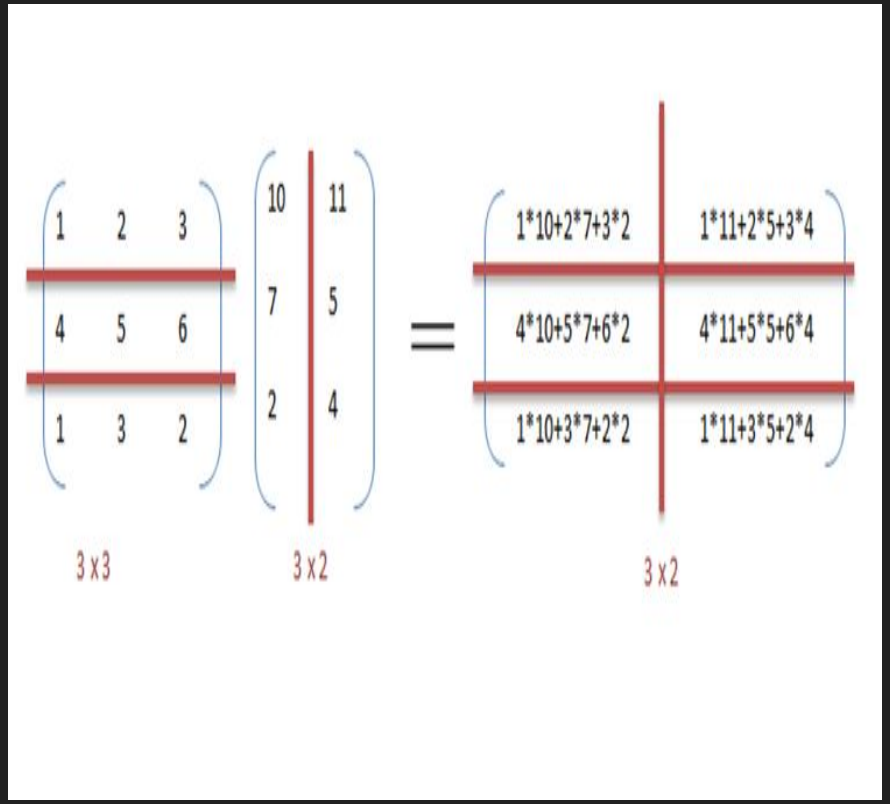
. By using task parallelism, we do not have to entirely depend on a single processor to finish a big task all by itself; that would take way too much time to process data and information, leading to more risks of system failures.



Data-level parallelism

. Data-level parallelism is a concept in which instead of processing only individual sets of data at a time, multiple sets would be processed at once.

. This increases the computational efficiency and in turns, decreases the amount of time it takes to fully process information from the data being given.



Parallel Architectures

Shared memory:

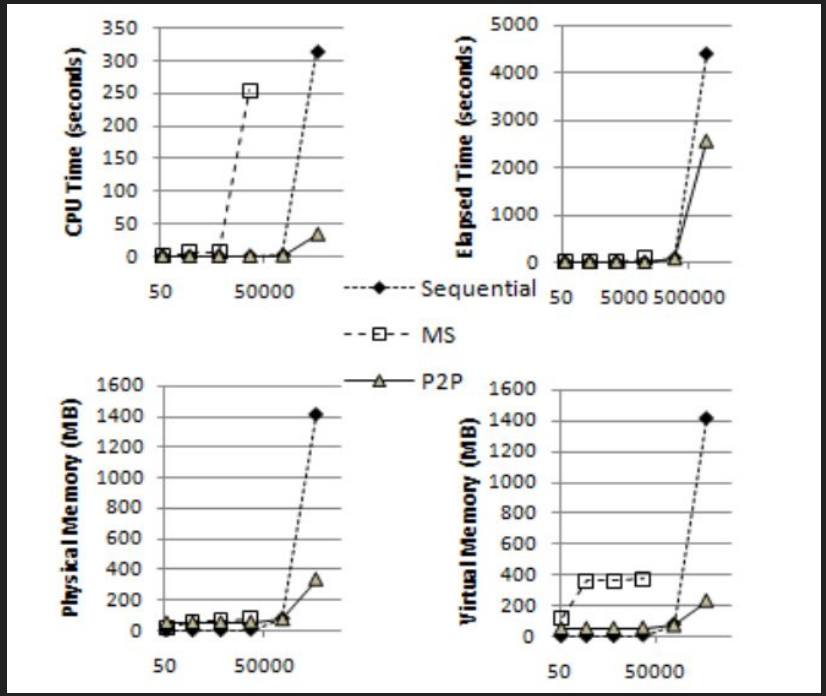
- Common, global memory space

Distributed Memory:

- Independent operations/computing and distinct memory space.

Multiplicity

P2P evaluation in relation to Master/Slave and Sequential implementation



Parallel Programming Models/Interfaces

Message Passing Mode/ Distributed Memory: MPI (Message Passing Interface)

Shared Memory: OpenMP(Open Multi-Processing)

CUDA: A GPGPU(General-purpose computing on graphics processing units) Api from Nvidia

MPI (Message Passing Interface)

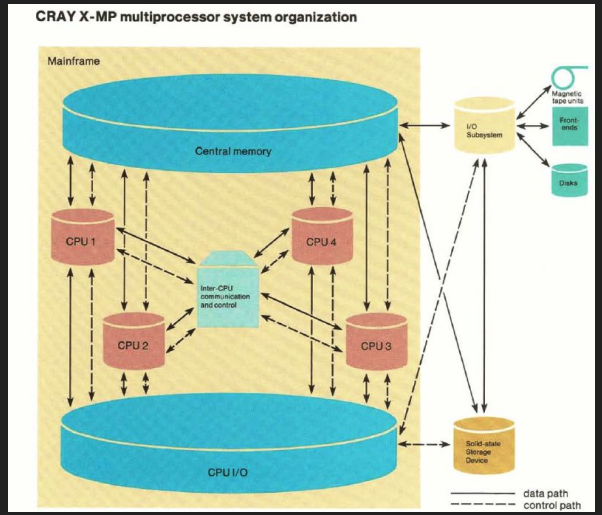
Before MPI:

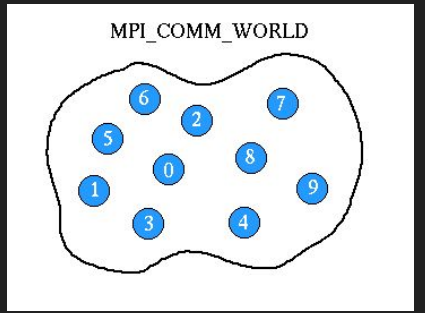
It is challenging to write portable and scalable parallel applications.

MPI:

A interface or standard for parallel programmers, enhancing the **portability** and **scalability** of parallel applications.

Functional Block diagram of the Cray X-MP (Mogill, 2010)





MPI (Message Passing Interface)

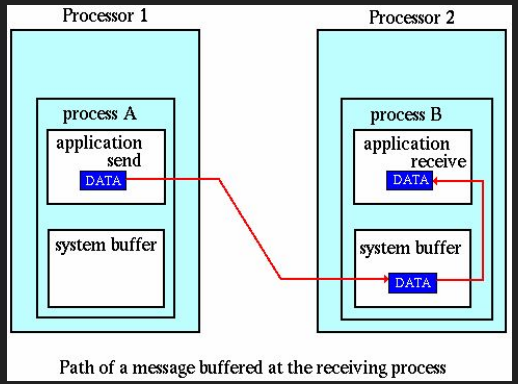
Point to Point:

MPI_Send, MPI_Recv, MPI_Sendrecv

Collective Communication:

MPI_Bcast, MPI_Gather, MPI_Scatter, MPI_Allgather, MPI_Reduce, MPI_Allreduce

command	send buffer				receive buffer			
	P_1	P_2	P_3	P_4	P_1	P_2	P_3	P_4
MPI_Send()+ MPI_Recv()		A					A	
MPI_Sendrecv		A	B			B	A	
MPI_Bcast		A			A	A	A	A
MPI_Gather	A	B	C	D		A,B,C,D		
MPI_Scatter		A,B,C,D			A	B	C	D
MPI_Allgather	A	B	C	D	A,B,C,D	A,B,C,D	A,B,C,D	A,B,C,D
MPI_Reduce	A	B	C	D		r(A,B,C,D)		
MPI_Allreduce	A	B	C	D	r(A,B,C,D)	r(A,B,C,D)	r(A,B,C,D)	r(A,B,C,D)



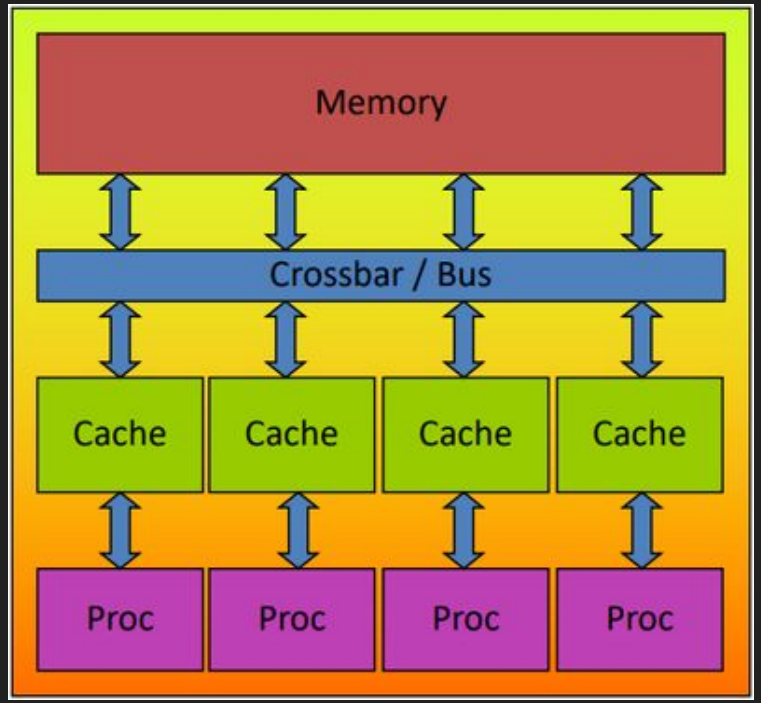
Path of a message buffered at the receiving process

OpenMP(Open Multi-Processing)

Shared Memory

High-Level Interface:

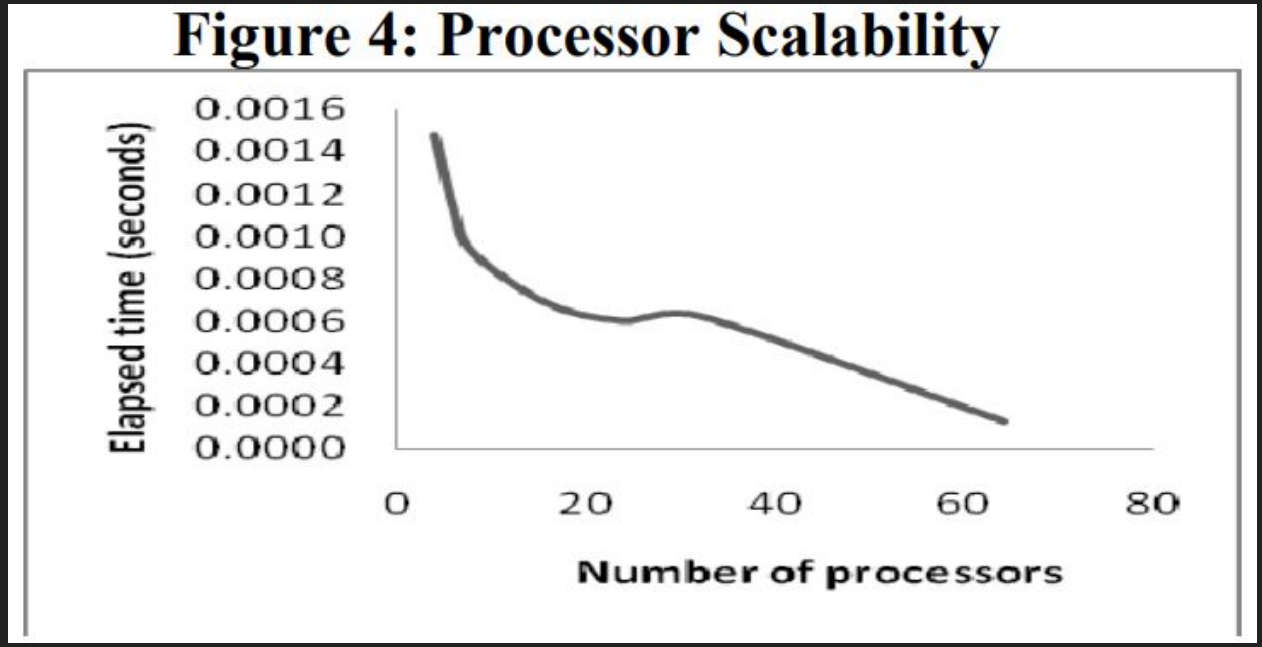
```
#include <stdio.h>
#include <omp.h>

int main(int argc, char** argv){
    int partial_Sum, total_Sum;

    #pragma omp parallel private(partial_Sum)
shared(total_Sum)
    {
        partial_Sum = 0;
        total_Sum = 0;

        #pragma omp for
        {
            for(int i = 1; i <= 1000; i++){
                partial_Sum += i;
            }
        }
    }
    return 0;
  }
```



Functions Diagram for OpenMP (Witham, File:openmpsharedmem.png - HPC wiki)

Scalability

Testing scalability



Synchronization

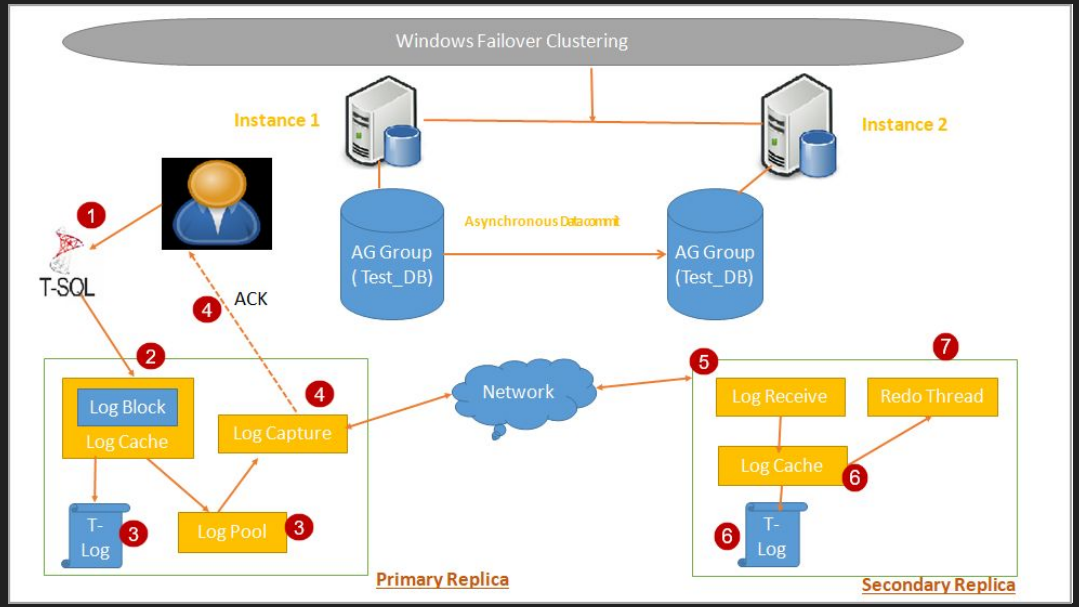
Parallel architecture

Competition

Concurrence

Two Types:

- Mutual exclusion
- Conditional



References

1. “Introduction to Parallel Computing.” *GeeksforGeeks*, GeeksforGeeks, 4 June 2021, www.geeksforgeeks.org/introduction-to-parallel-computing/.
2. “Computer Architecture: Data-Level Parallelism Cheatsheet.” *Codecademy*, www.codecademy.com/learn/computer-architecture/modules/data-level-parallelism/cheatsheet. Accessed 6 Dec. 2023.
3. “Parallel Computing.” *Wikipedia*, Wikimedia Foundation, 29 Nov. 2023, en.wikipedia.org/wiki/Parallel_computing#:~:text=There%20are%20several%20different%20forms,physical%20constraints%20preventing%20frequency%20scaling.
4. “Parallel Computing and Its Modern Uses: HP® Tech Takes.” *Parallel Computing And Its Modern Uses | HP® Tech Takes*, www.hp.com/us-en/shop/tech-takes/parallel-computing-and-its-modern-uses. Accessed 6 Dec. 2023.
5. Author links open overlay panelAtlas A Akhmetzyanov *, and AbstractIn this paper we consider problems of optimization and selection of development systems (technologies) of oil/gas fields. “Parallel Computing in Optimal Design of Development of Multilayer Oil and Gas Fields.” *IFAC Proceedings Volumes*, Elsevier, 14 June 2012, www.sciencedirect.com/science/article/pii/S1474667015372633.
6. Barney, B. (2007). Message Passing Interface (MPI). Message passing interface (MPI). https://wstein.org/msri07/read/Message%20Passing%20Interface%20%28MPI%29.html
7. Using openmp with C¶. Using OpenMP with C - Research Computing University of Colorado Boulder documentation. (n.d.). https://curc.readthedocs.io/en/latest/programming/OpenMP-C.html
8. witham, J. (n.d.). File:openmpsharedmem.png - HPC wiki. https://hpc-wiki.info/hpc/File:OpenMPSharedMem.png
9. Mogill, J., & Haglin, D. (05 2010). A Comparison Of Shared Memory Parallel Programming Models. Cray Users Group Conference (CUG20120).