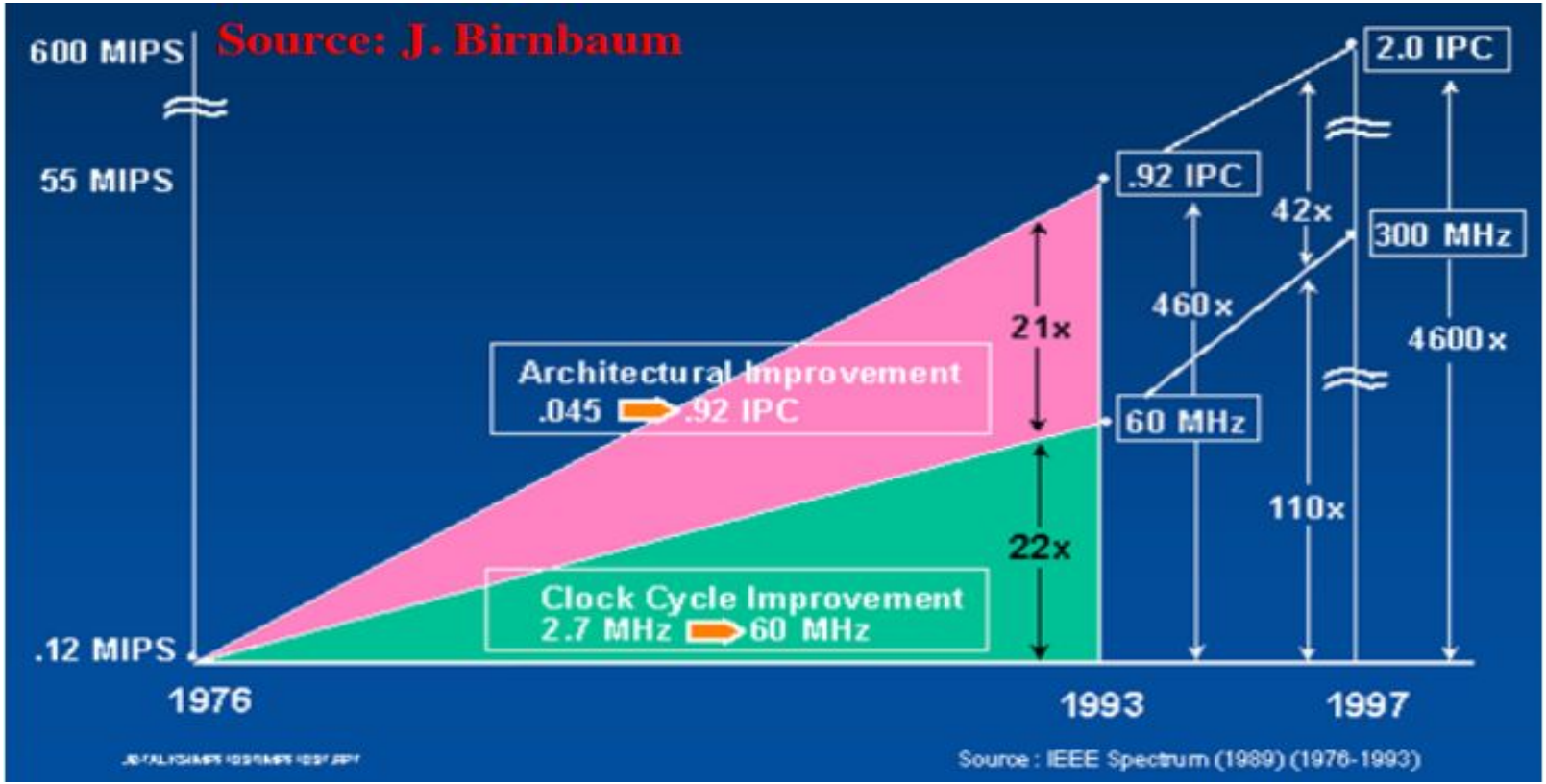
CS 5005
Parallel Programming

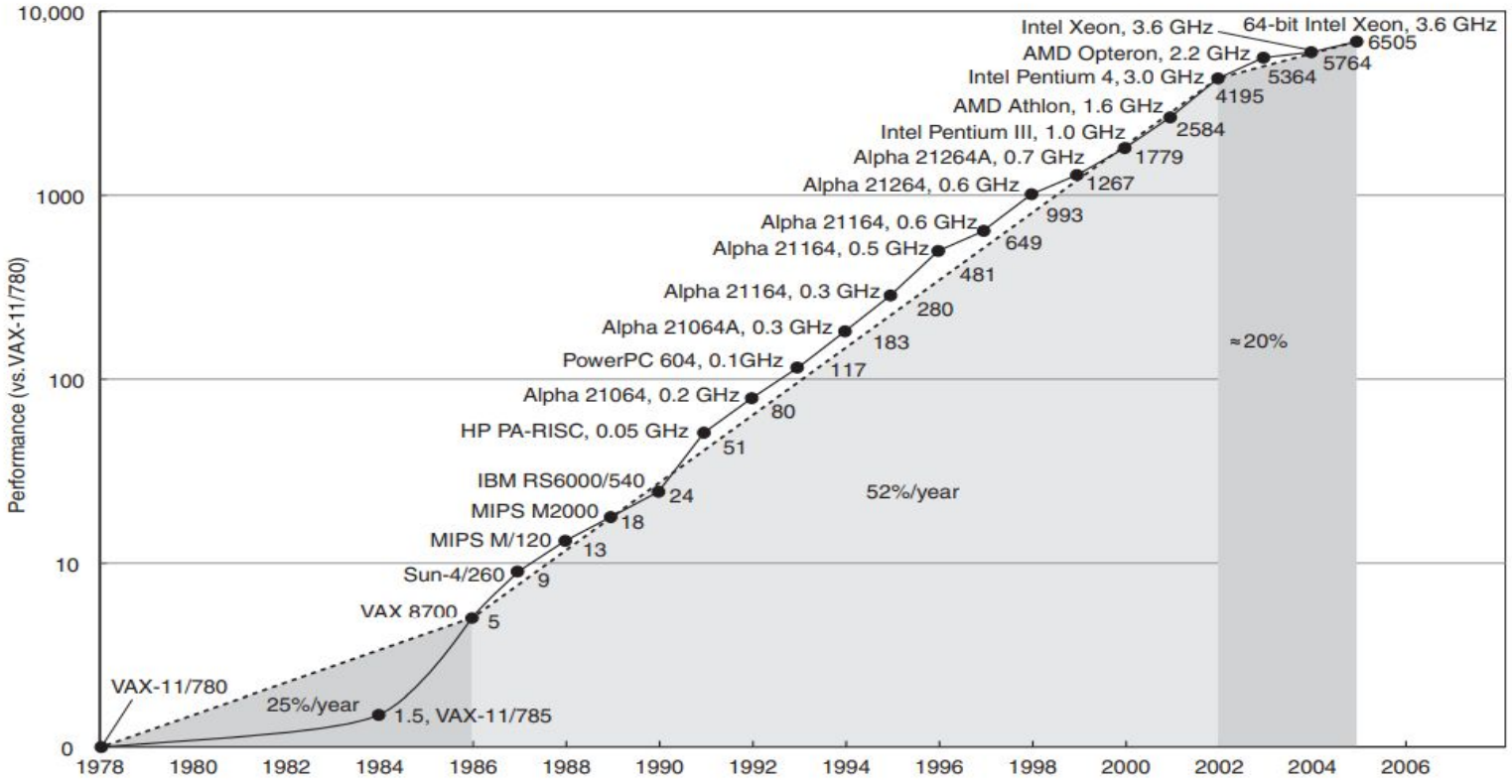
UNNIKRISHNAN C

Course Brief

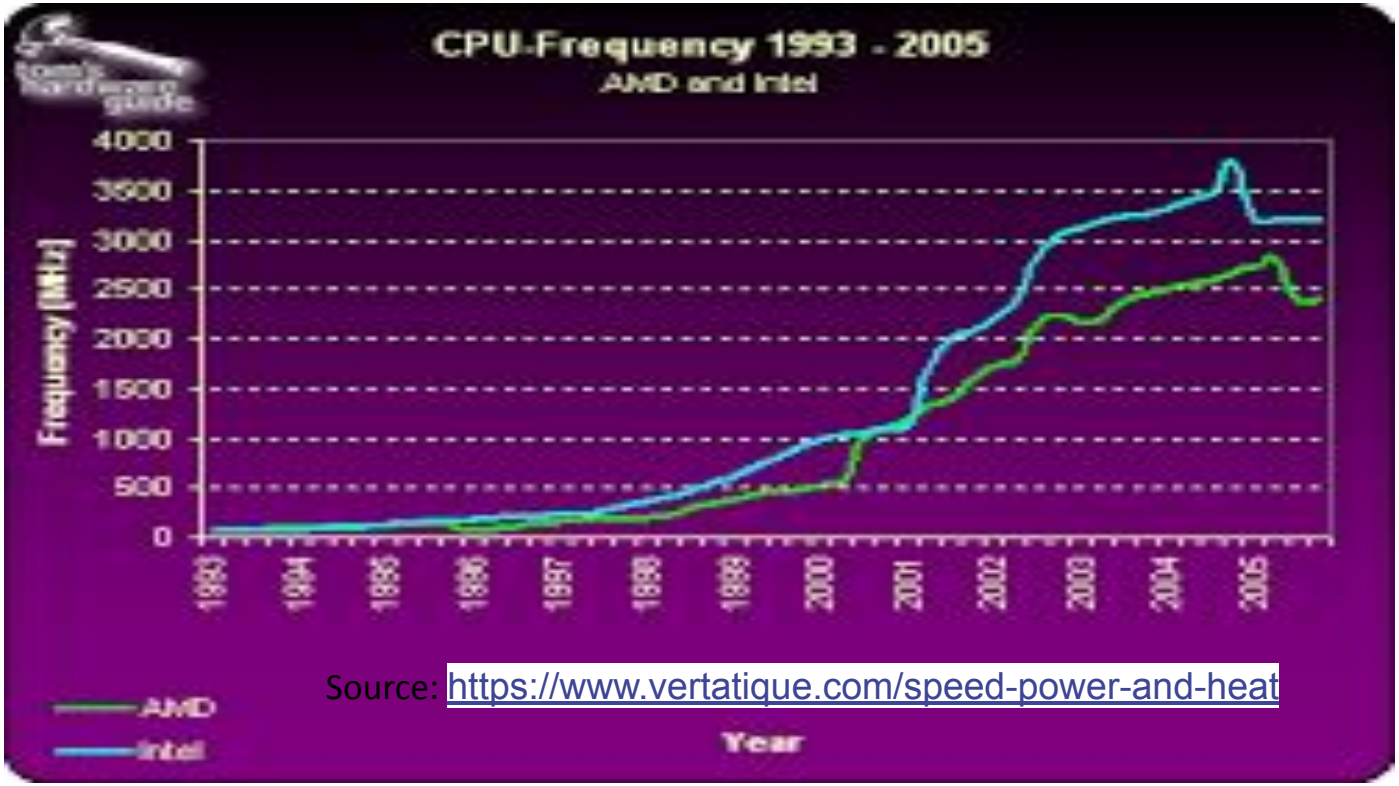
- Teaching Assistant
  - Nilanjana Debnath
  - MS Student
- Course Contents
  - PRAM Model, multi-core, many-core, and distributed programming
  - OpenMP, CUDA, MPI
  - Parallel Algorithms, if time permits parallel data structures also.



The Good Old Days for Software: Automatic Performance Improvement



Source: http://www.cs.columbia.edu/~sedwards/classes/2012/3827-spring/advanced-arch-2011.pdf



Source: https://www.vertatique.com/speed-power-and-heat

Power Wall: Frequency cannot be increased beyond a limit.

What you do with Increased Computing Power

- Do more computation.
- Increased Storage Requirements.
- Personal computers in late 90’s had hard disk with storage capacity less than 10 GB
- What about storage capacity in personal computers now?
  - > 1000 GB(one TB).
  - More than 100 fold in two decades
- Transistors which can be occupied in a an area double every year (Moore’s Law based on observation)
- So Frequency cannot increase beyond a limit what is the alternative?

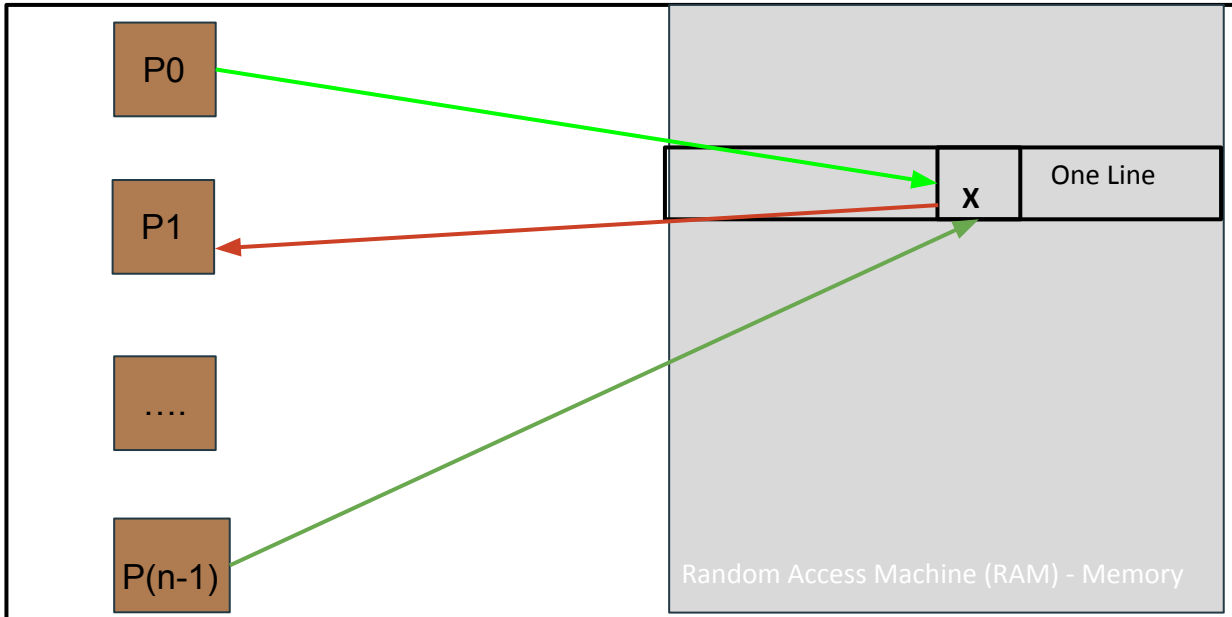
Parallel Computers

- Multi-core architectures: Current CPUs in laptop or Desktop
- Many-core architecture: GPUs.
- Cluster : Distributed Systems with machines
- Other parallel architectures
  - FPGA
  - ASIC
- Homogeneous and heterogeneous clusters possible.

Basics of Computer Architecture

- Flynn's Taxonomy (classification of Computers)
  - Single Instruction Single Data (SISD)
  - Multiple Instruction Multiple Data (MIMD)
  - Single Instruction Multiple Data (SIMD)
  - Multiple Instruction Single Data (MISD)
- Focus Architectures for the course
  - Multi-core CPU (MIMD)
  - Graphics Processing Unit (GPU) - SIMD/SIMT

Parallel RAM Machine - MIMD



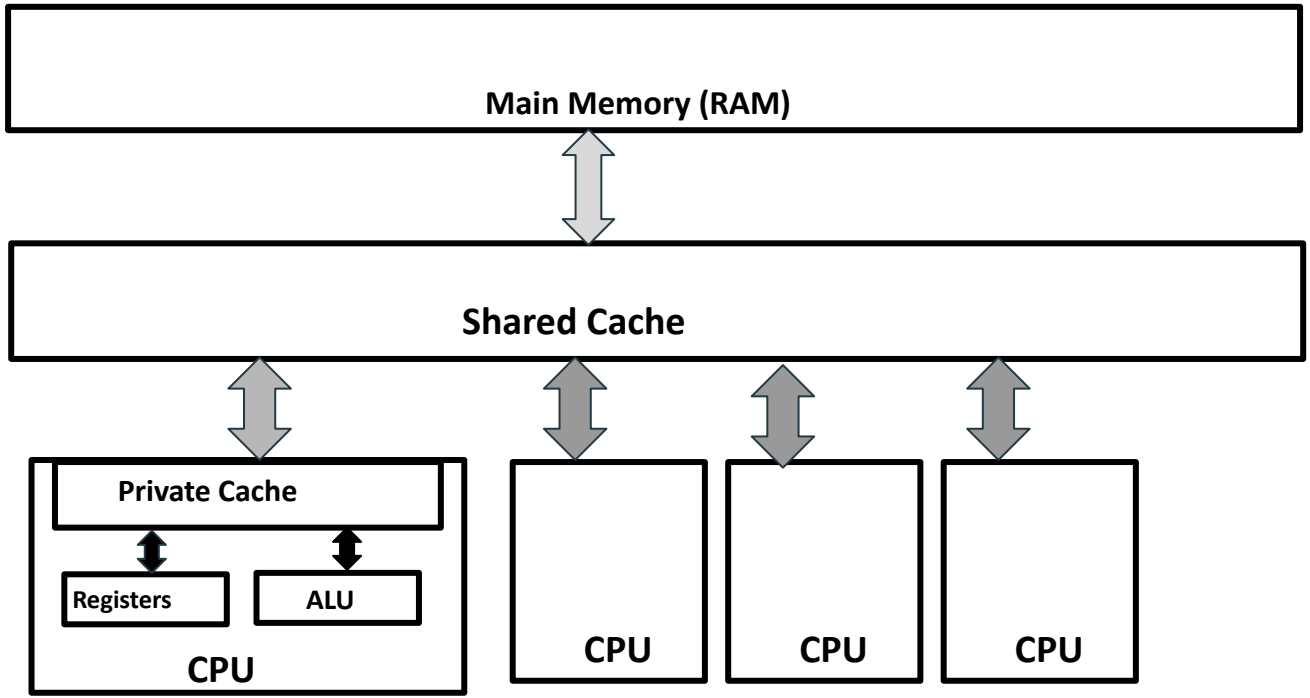
Parallel Writes or
Parallel Read and Write
leads to incorrect results

If all threads only writes, and
write the same value
It is okay.

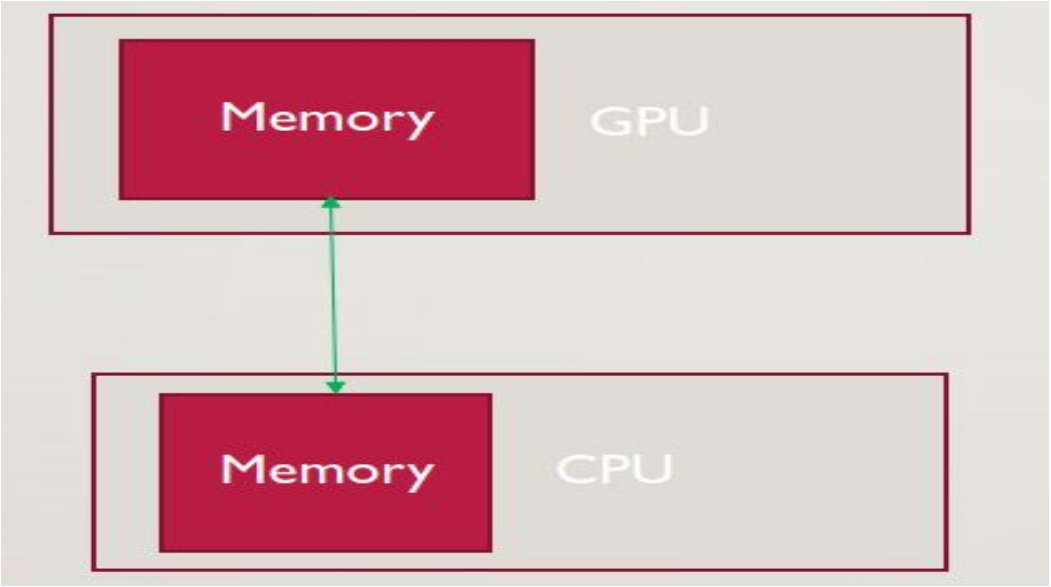
Examples for Different Hardwares

Device	Architecture	Total cores	RAM	Programmming
Multi-core CPU	MIMD	8 to 64	high	C++/C & OpenMP Library
Nvidia-GPU	SIMT	> 1000	low	CUDA
Intel i7-8550U		8 cores	high	C++/C & OpenMP
Nvidia K10	SIMT	2496	low	CUDA
Distributed systems				C/C++ & OpenMP + MPI

Multi-core CPU Architecture



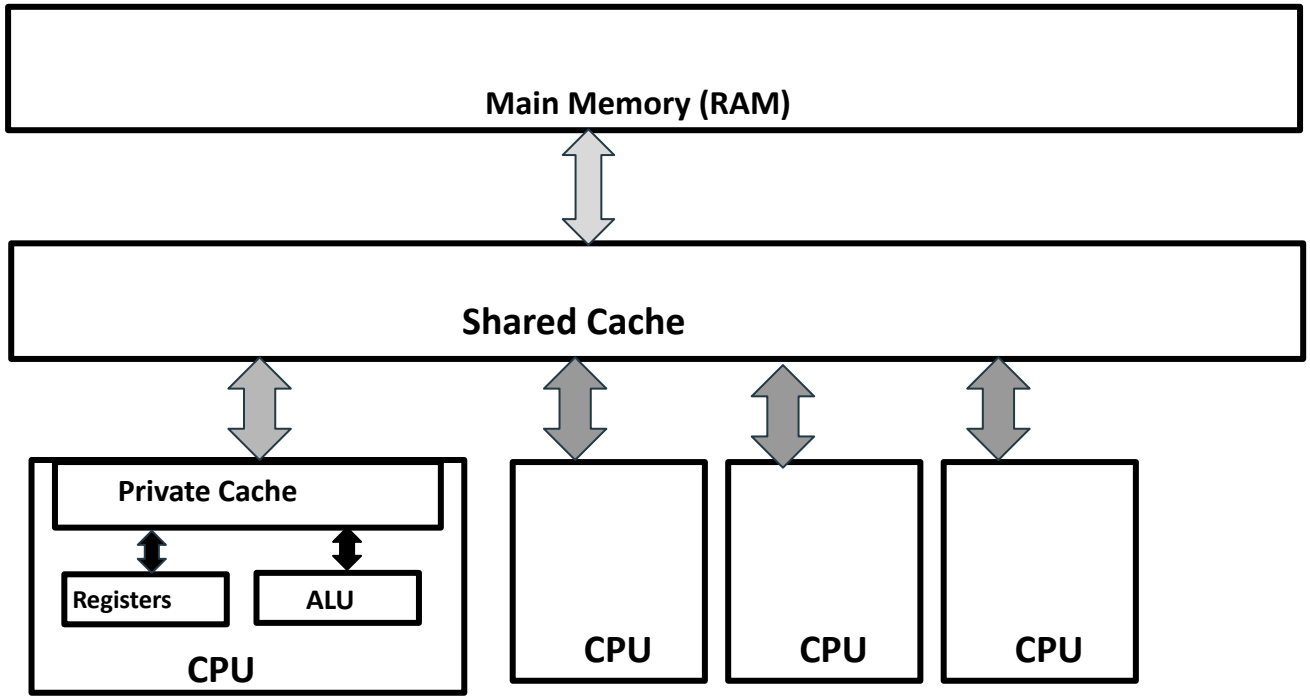
A CPU (host) with GPU (device) attached to it



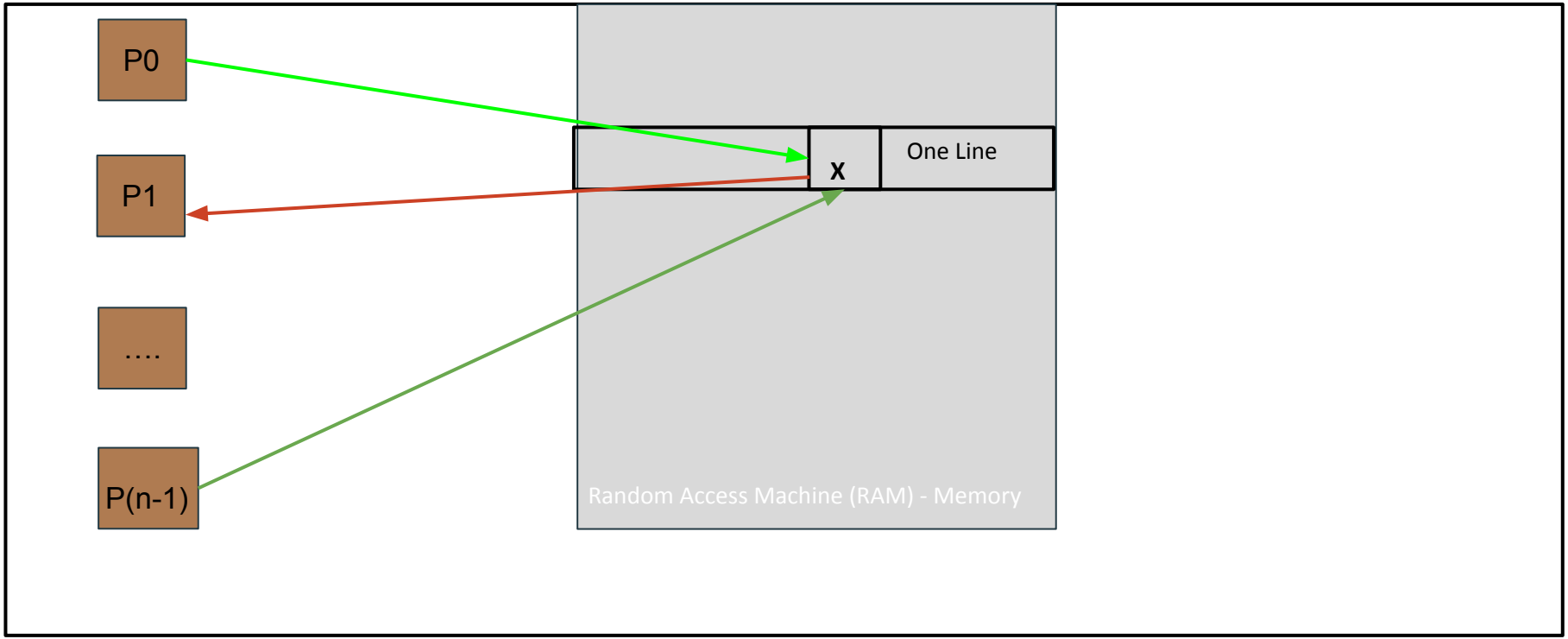
Efficient Algorithms on CPU and GPU

- The algorithms needs to be parallel
- CPU number of threads will be less than hundred.
- GPU number threads greater than thousand.
- Enough work and parallelism should be there in the algorithm.
- Matrix addition with matrices of size 10x10: Where it can be run?
- What if size of the matrices 1024x1024?
- Graph Algorithms
    - Irregular Access Patterns
    - Two threads may try to update the same memory locations
    - How to make the algorithm parallel
    - Atomic operations- CAS, ADD etc.

Multi-core CPU Architecture



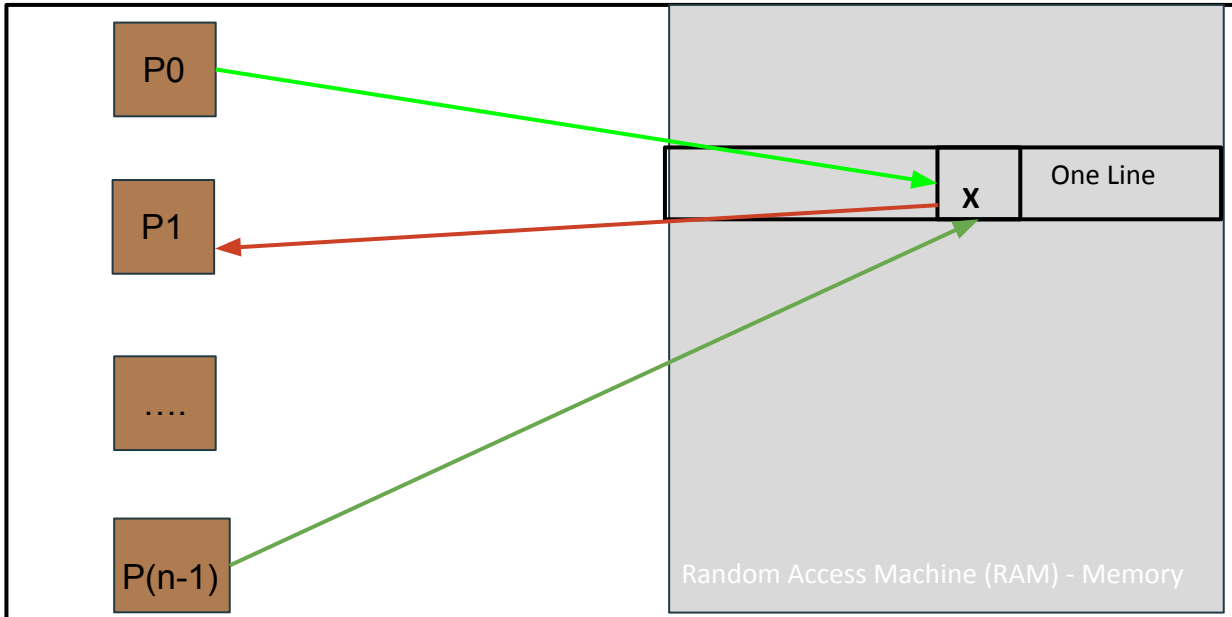
Parallel RAM Machine - MIMD



OpenMP History

Version	Fortran Release Time	C/C++ Release Time
1.0	October, 1997	October, 1998
2.0	2000	2002
2.5	combined C/C++/Fortran (2005)	
3.0	2008	
4.0 (GPU support)	2013	
5.2	Current Version	

Parallel RAM Machine - MIMD



Parallel Writes or
Parallel Read and Write
leads to incorrect results

If all threads only writes, and
write the same value
It is okay.

OpenMP: HelloWorld

```
#include<stdio.h>
#include<omp.h>
int  main() {
int sum=20;
#pragma omp parallel
{
 printf(“Hello World”);
sum=sum+20;
}
printf(“SUM %d\n”,sum);
}//end main
```