CS516: Parallelization of Programs

Introduction

Vishwesh Jatala

Assistant Professor

Department of CSE

Indian Institute of Technology Bhilai

vishwesh@iitbhilai.ac.in

2023-24 W

What is the output?

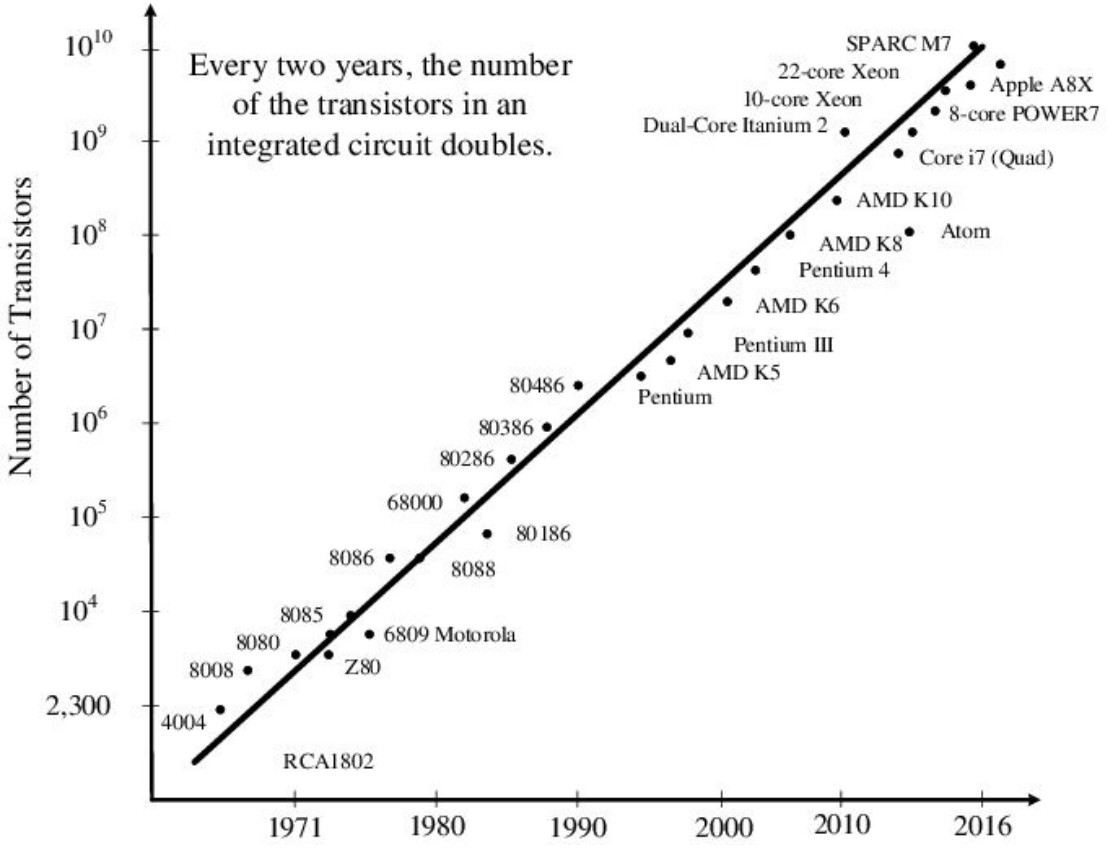
```
1 #include<stdio.h>
2
3 void main()
4 {
5
6         printf("----------------------------------------\n\n\t   ");
7         printf("%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c",
8                         72,65,80,80,89,32,78,69,87,32,89,69,65,82,32,50,48,50,52,33);
9         printf("\n\n----------------------------------------\n");
10 }
```

Outline of Today's Lecture

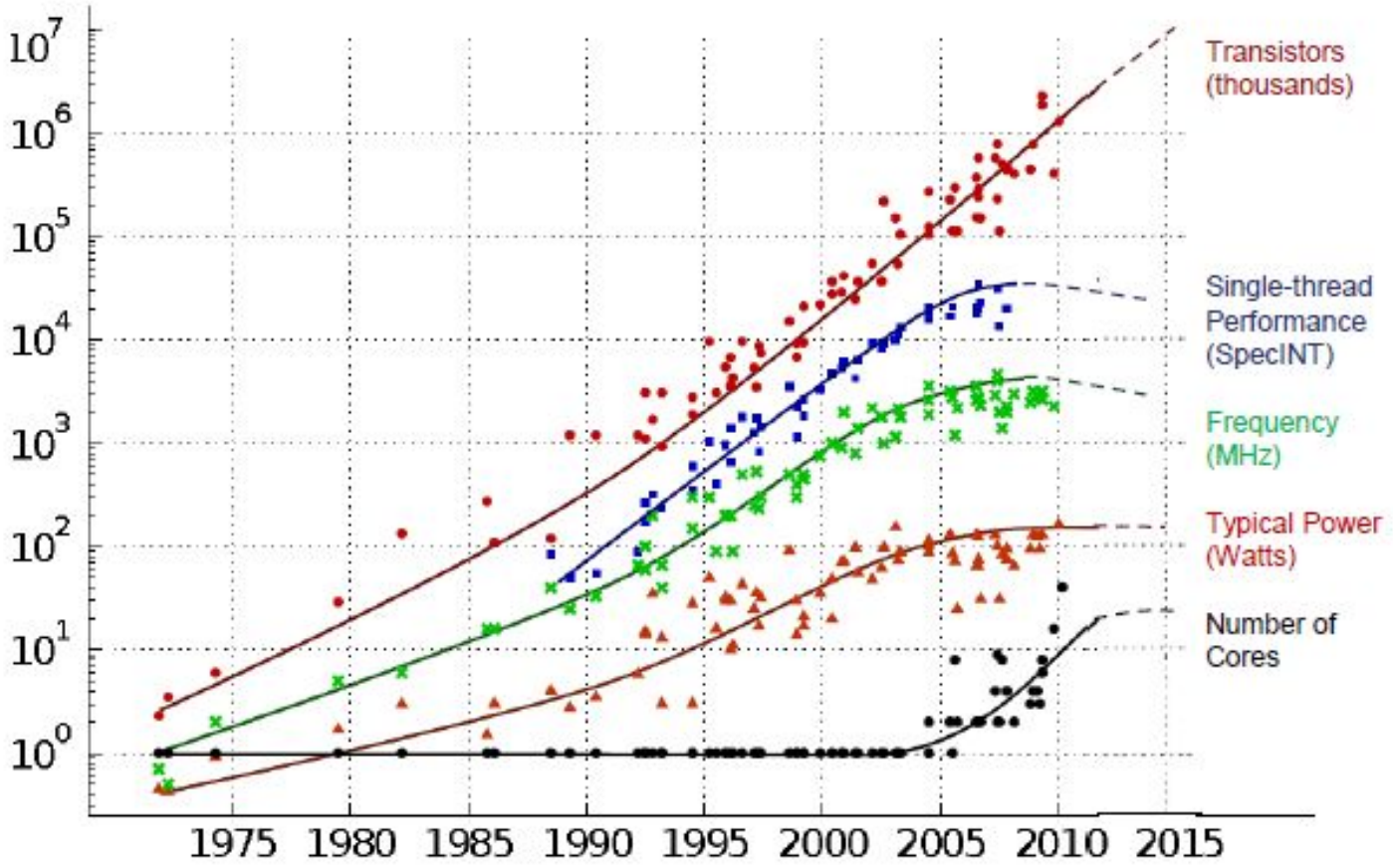
- Why?
- What?
- How?

Moore's Law

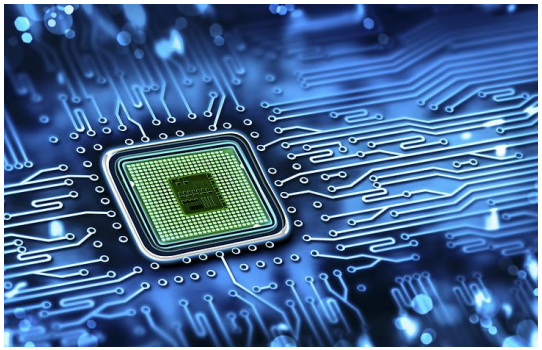
- *The number of transistors on a IC doubles about every two years*

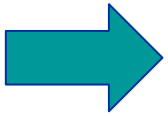


Moore's Law Effect

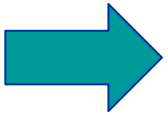


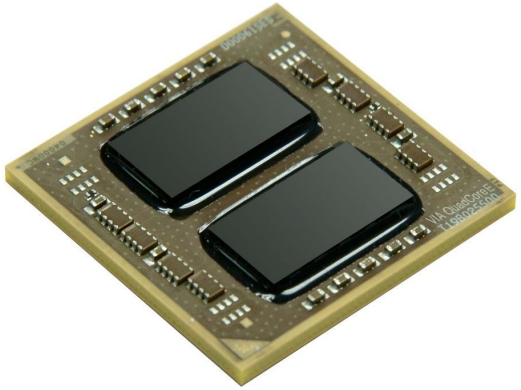
Moore's Law Effect





Single Core Performance

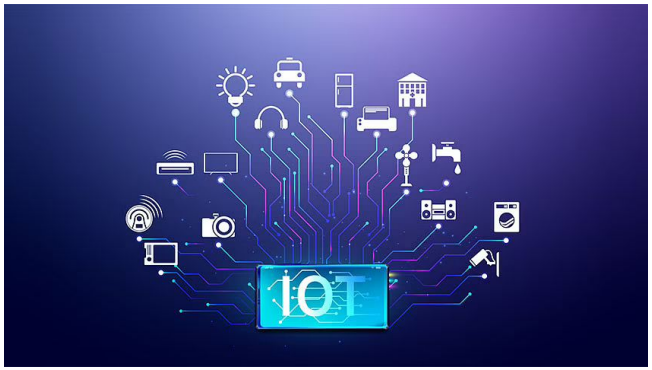




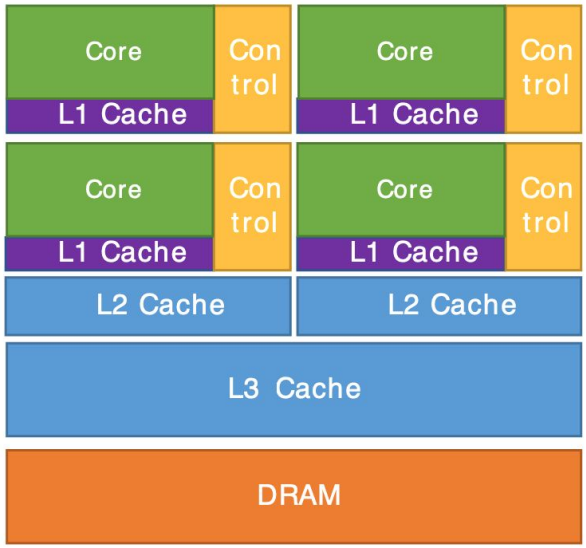
Power and Heat

Multicore Processors

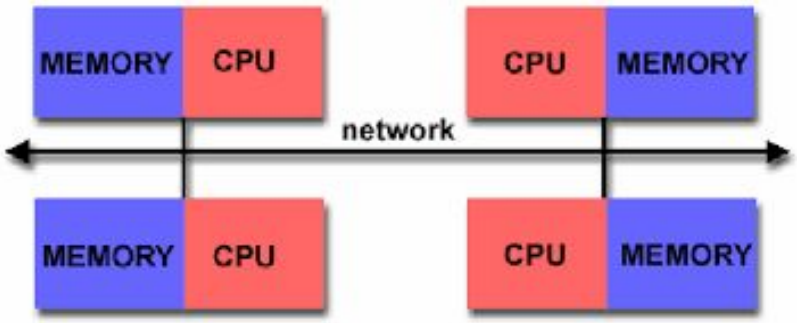
Parallel Architectures are Everywhere!



Parallel Hardwares



Multicores



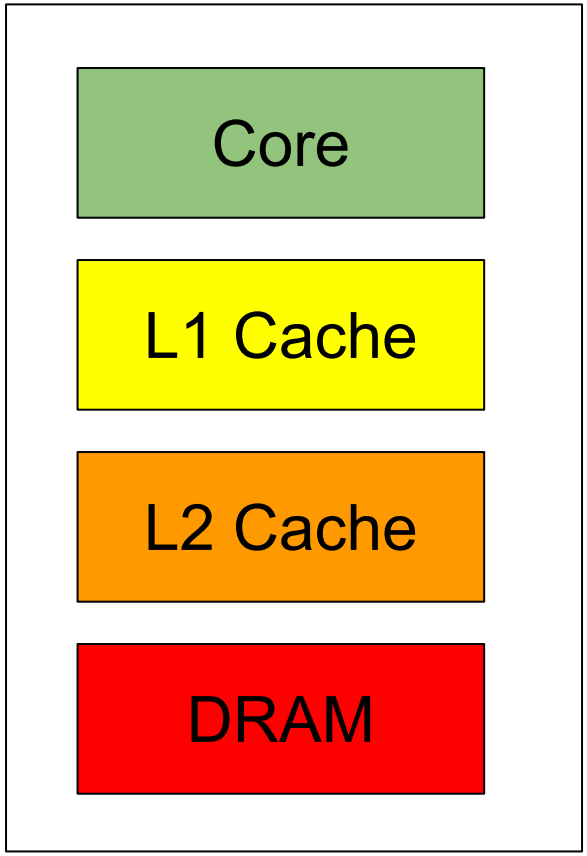
Distributed CPUs

GPUs

Hardware and Software

```
1  package com.beginnersbook;
2  public class JavaExample {
3      public static void main(String []args) {
4          String str[] = { "Ajeet", "Steve", "Rick", "
5          String temp;
6          System.out.println("Strings in sorted order:
7          for (int j = 0; j < str.length; j++) {
8              for (int i = j + 1; i < str.length; i++)
9                  // comparing adjacent strings
10                 if (str[i].compareTo(str[j]) < 0) {
11                     temp = str[j];
12                     str[j] = str[i];
13                     str[i] = temp;
14                 }
15             }
16             System.out.println(str[j]);
17         }
18     }
19 }
```

Sequential

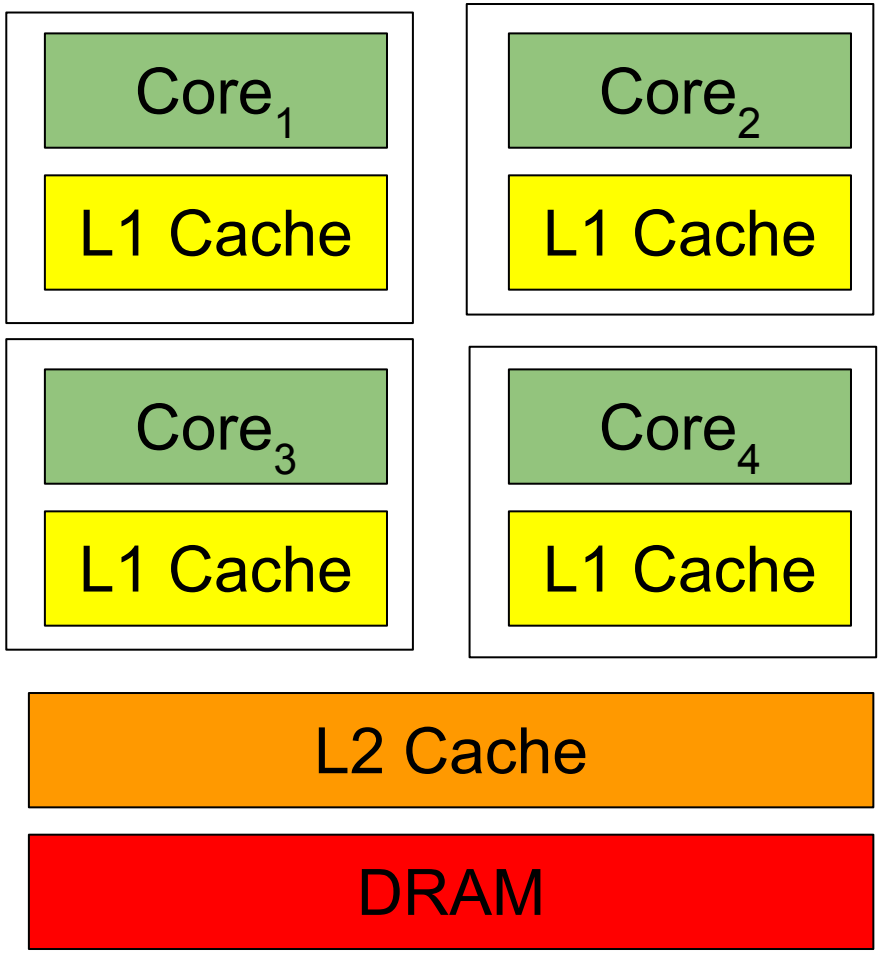


Single-core CPU

Output

Hardware and Software

```
1  package com.beginnersbook;
2  public class JavaExample {
3      public static void main(String []args) {
4          String str[] = { "Ajeet", "Steve", "Rick", "
5          String temp;
6          System.out.println("Strings in sorted order:
7          for (int j = 0; j < str.length; j++) {
8              for (int i = j + 1; i < str.length; i++)
9                  // comparing adjacent strings
10                 if (str[i].compareTo(str[j]) < 0) {
11                     temp = str[j];
12                     str[j] = str[i];
13                     str[i] = temp;
14                 }
15             }
16             System.out.println(str[j]);
17         }
18     }
19 }
```

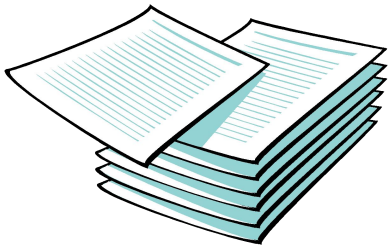


Same sequential?

Multi-core CPU

Professor P

15 questions
300 Answer sheets



Professor P's Teaching Assistants

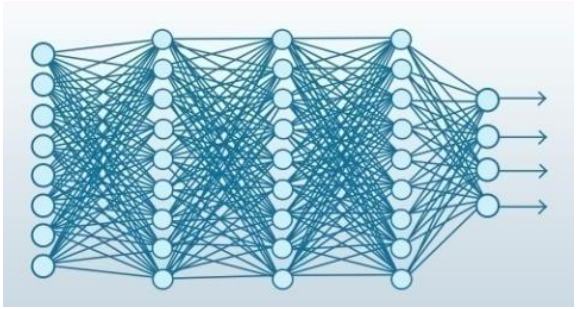
Benefits of Parallel Programming

Fast: Less execution time

Save Money

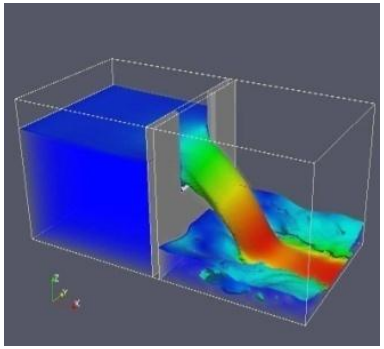
Solves Larger Problem

Parallel Programming Applications

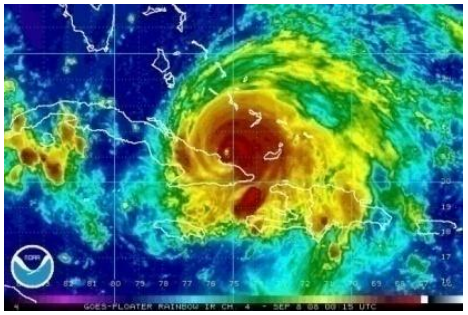


Deep Learning and
Machine Learning

Medical Imaging



CFD



Climate Modeling

Parallel Programming Applications

OpenAI used a super computer
with more than 285,000 CPU cores,
10,000 GPUs and
400 gigabits per second of network
connectivity for each GPU server.

Challenges!

Example-1

```
for (int i=0; i<5; i++)
    A[i]= i
```

Sequential Execution

Core-0

A[0]=0

A[1]=1

A[2]=2

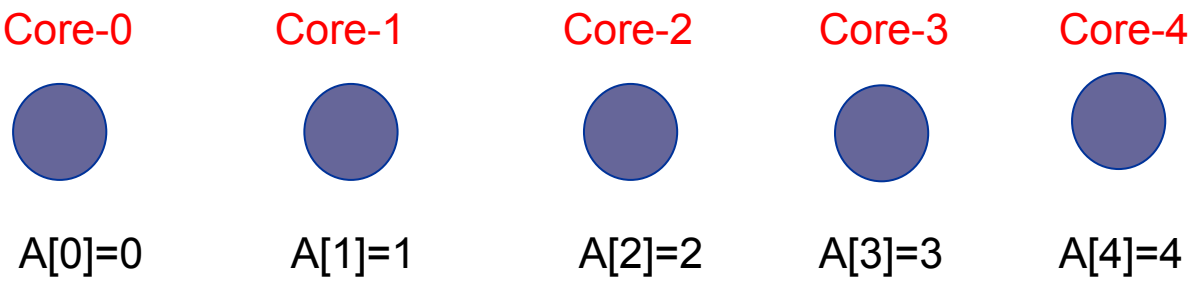
A[3]=3

A[4]=4

Example-1

```
for (int i=0; i<5; i++)
    A[i]= i
```

Parallel Execution



Example-2

```
int count = 0;
for (int i=0; i<5; i++)
    A[i]= count++;
```

Sequential Execution

Core-0

A[0]=0

A[1]=1

A[2]=2

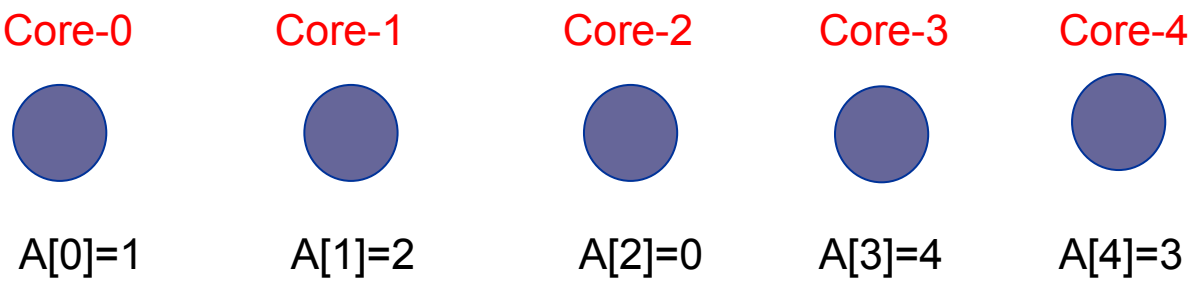
A[3]=3

A[4]=4

Example-2

```
int count = 0;
for (int i=0; i<5; i++)
    A[i]= count++;
```

Parallel Execution



Challenges:

Detecting Parallelism is Hard!

Example-3: Sequential Version

```
int sum = 0;
for (i = 0; i < n; i++) {
    x = f(i);
    sum = sum+x;
}
```

A Sequential Program for Sum

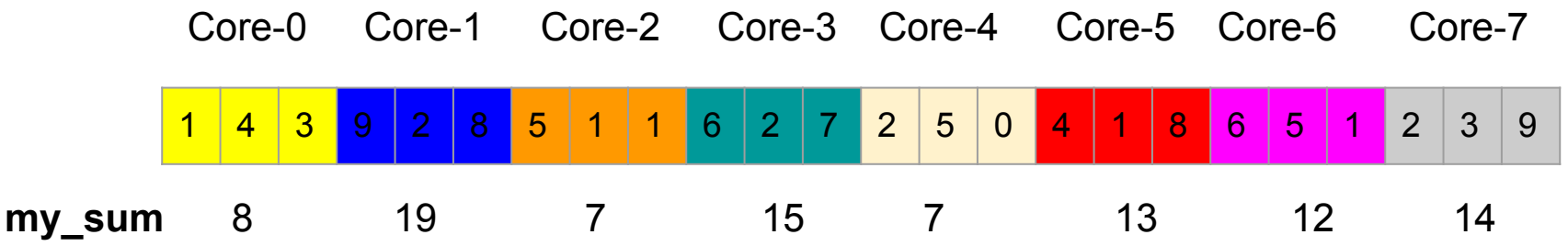
Core-0

1	4	3	9	2	8	5	1	1	6	2	7	2	5	0	4	1	8	6	5	1	2	3	9

sum = 95

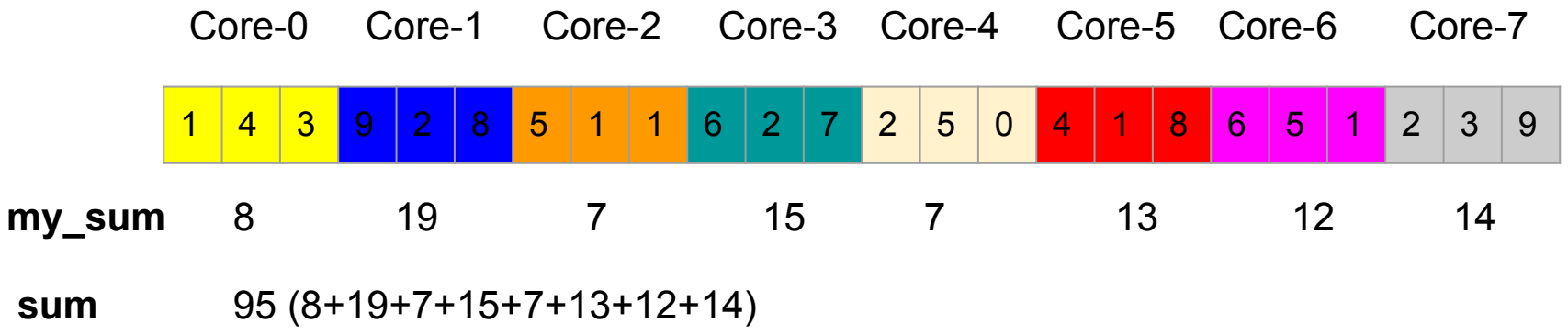
Example-3: Parallel Version-1

```
my_sum = 0;
my_first_i = . . . ;
my_last_i = . . . ;
for (my_i = my_first_i; my_i < p_last_i; my_i++) {
    my_x = f(i);
    my_sum += my_x;
}
```

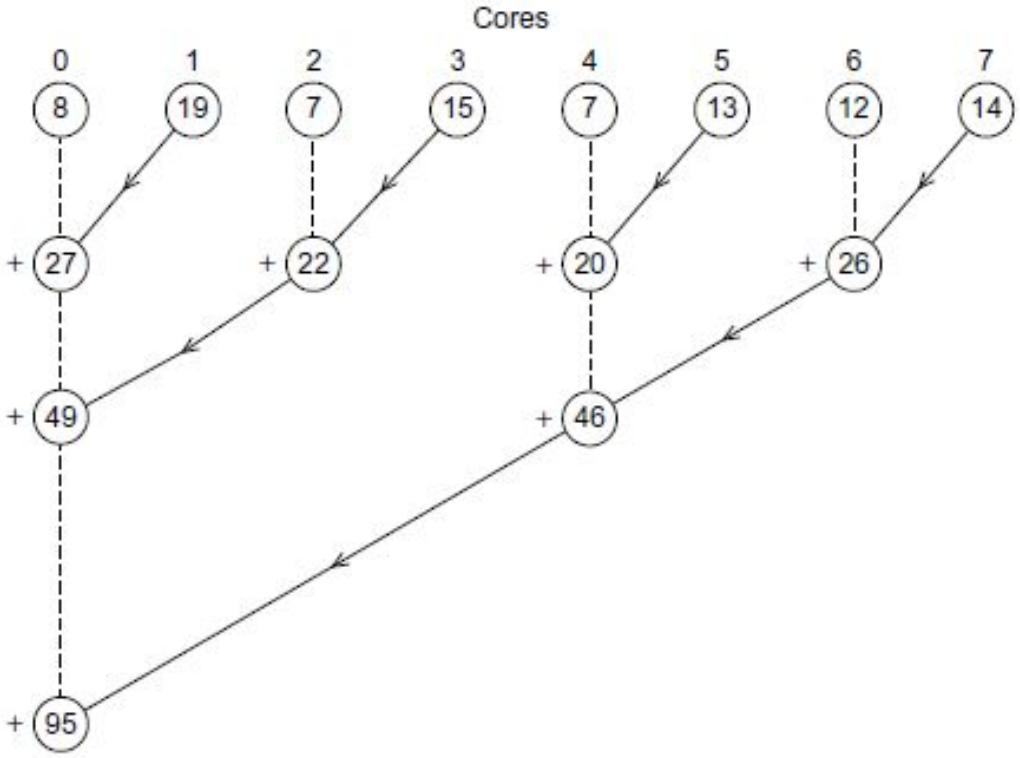


Example-3: Parallel Version-1

```
if (I'm the master core) {
      sum = my sum;
      for (each core other than myself) {
            receive value from core;
            sum += value;
      }
}
else
    { send my_sum to the master; }
```



Example-3: Parallel Version-2



Parallel Version-1 or Version-2?

- Both have same number of operations
- Version-1 sum is sequential
- Version-2 exposes parallelism

Challenges:

- Detecting Parallelism is Hard!
- Communication
- Synchronization

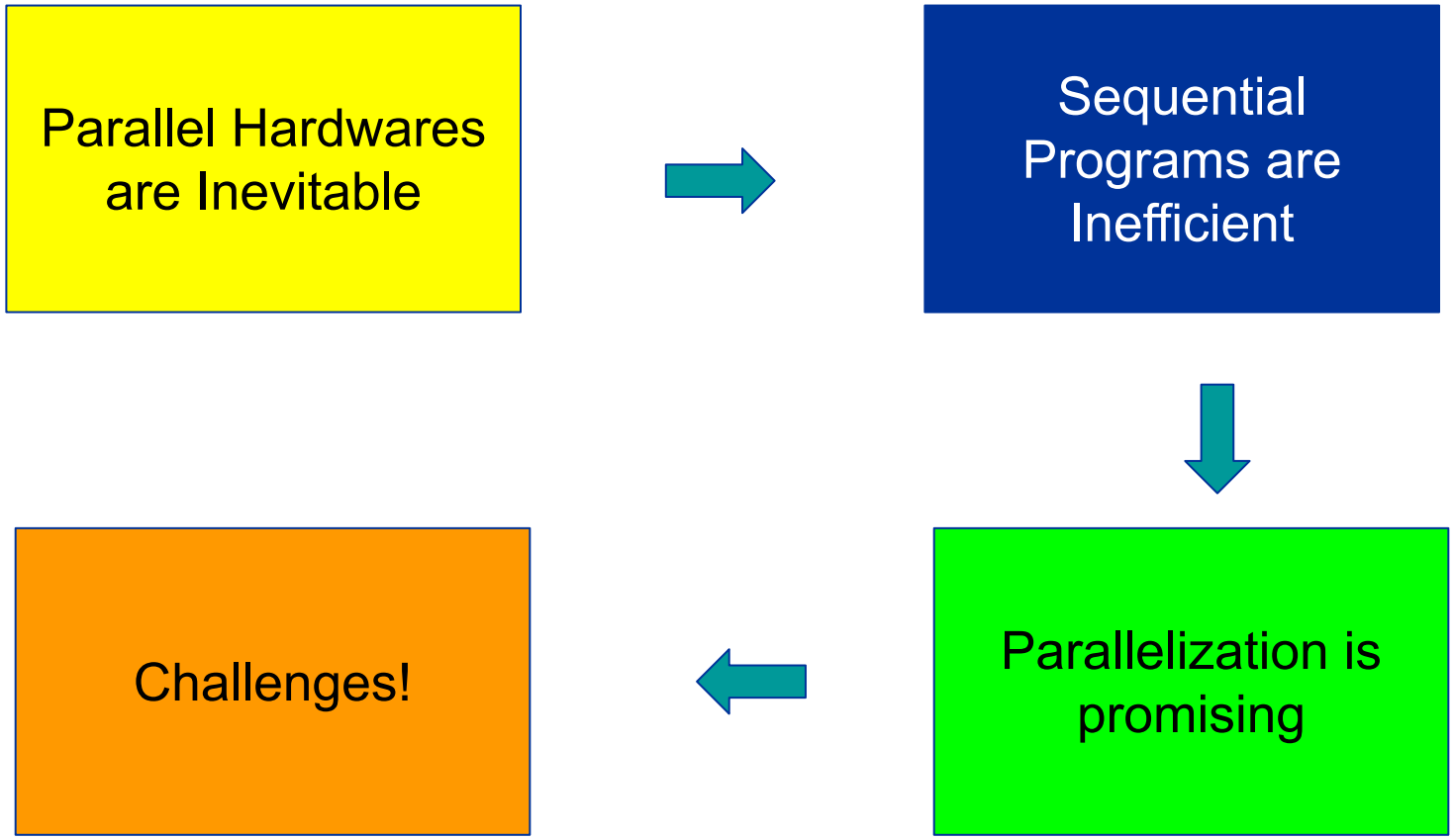
Example-4:

```
int sum = 0;
for (i = 0; i < n; i++) {
    x = f(i);
    sum = sum+x;
}
```

Challenges:

- Detecting Parallelism is Hard!
- Communication
- Synchronization
- Load Balancing

What did we learn so far?



Let’s discuss details from next class!

Course Outline (Part-1)

- Introduction (this lecture)
- Overview of Parallel Architectures and Programming models
- Amdahl's law and Performance
- Parallel programming
  - GPUs and CUDA programming
  - Optimizations
- Case studies
- Extracting Parallelism from Sequential Programs Automatically

Course Logistics

- Lecture Hours:
  - Mon, Tue, Thursday 10:30 am - 11:25 am

- Course Website: Canvas platform
  - Lecture notes
  - Assignments
  - Project
  - Discussions
  - Marks

Course Logistics: Evaluation Scheme

- Evaluation scheme (can be changed slightly):
  - Exams: ~40%
  - Project: ~35%
  - Attendance: ~10%
  - Assignments (Paper presentation?): 15%

- Attendance
  - 0% - 50%: 0 Marks
  - >50%: Marks will be awarded out of 10 accordingly.
  - Example:
    - Total sessions: 16
    - #sessions attended = 7 (<50%), marks = 0
    - #sessions attended = 10 (62.5%), marks = 2.5 (2*10/8)

Course Logistics: References

- Lecture notes will be available on Canvas
- Reference material will be provided on Canvas
- Text book for extracting parallelism:
  - Randy Allen, Ken Kennedy,Optimizing Compilers for Modern Architectures: A Dependence-based Approach, Morgan Kaufmann, 2001

Course Logistics: Tools

- Platforms:
  - Prefer Google Colab for GPUs and CUDA Programming.
  - A demo session

Course Logistics

- Evaluation Policy:
  - Acknowledge all the sources
  - **Do not cheat**

Outcome of the Course?

- State-of-the-art techniques in parallel computing
- Develop parallel programming skills
- Transferable skills -
  - Parallel programming is used in multiple disciplines
  - Industries
  - Education and research
- Handle projects
- High-performance computing

Course Logistics

Office: B-010

vishwesh@iitbhilai.ac.in

CANVAS

References

- https://www.cse.iitd.ac.in/~soham/COL380/page.html
- https://s3.wp.wsu.edu/uploads/sites/1122/2017/05/6-9-2017-slides-vFinal.pptx
- Miscellaneous resources on internet

Thank you!