



7 - INTRO TO DATA PROCESSING

Management and Analysis of Physics Datasets - Module B
Physics of Data

A.A. 2023/2024

PROCESSING



Processing is going to be referred in this context in a very broad sense as the set of operations to be performed on your dataset to extract higher-level information



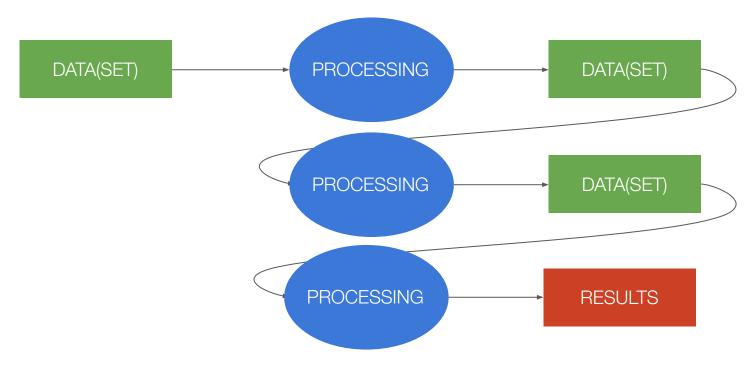
It's a very simplistic view of the topic, as many subtleties and nuances are hidden under the bonnet of the term

- Analytics
- Data Reduction
- Data Cleansing
- Feature enrichment
- -

PROCESSING



Furthermore, within a generic computing model, more often than not multiple processing "steps/stages" have to be performed on the input datasets, including the storage of the intermediate results



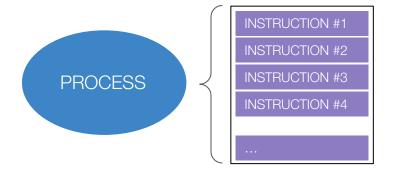
The usual question still stand...

how does the processing scale with the data size / task complexity / ...?

A BRIEF RECAP



At is very core, *processing* can be seen as the sequential execution of a set of instructions by the CPU, where both the data and the instructions are kept in memory



The CPU follows a fetch-decode-execute cycle for each instruction

- Fetch → retrieving instruction from memory
- Decode → interpreting of the instruction
- Execute → execution of the instruction

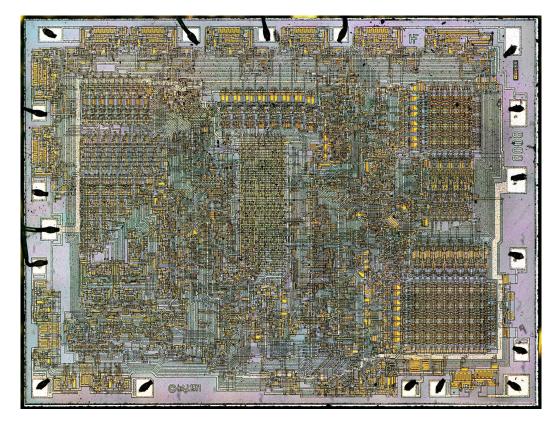
Input Device

Output Device

Von Neumann architecture

A BRIEF RECAP

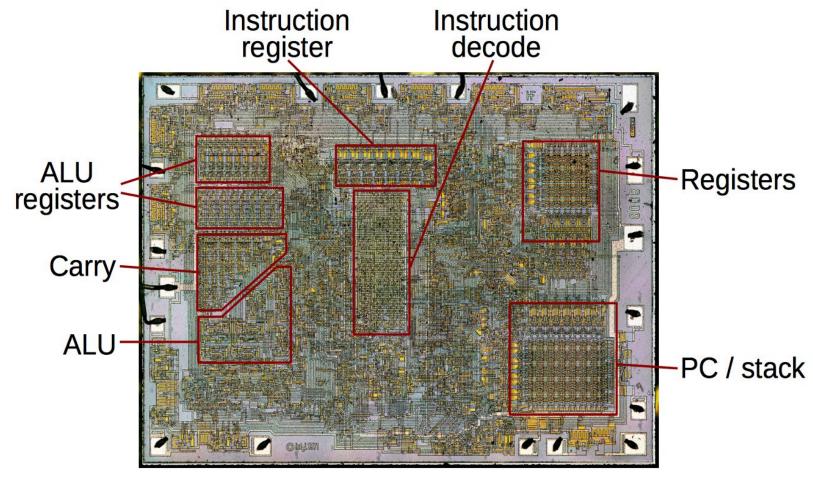




The Intel 8008 microprocessor, from 50 years ago https://www.righto.com/2016/12/die-photos-and-analysis-of 24.html

A BRIEF RECAP





The Intel 8008 microprocessor, from 50 years ago https://www.righto.com/2016/12/die-photos-and-analysis-of 24.html

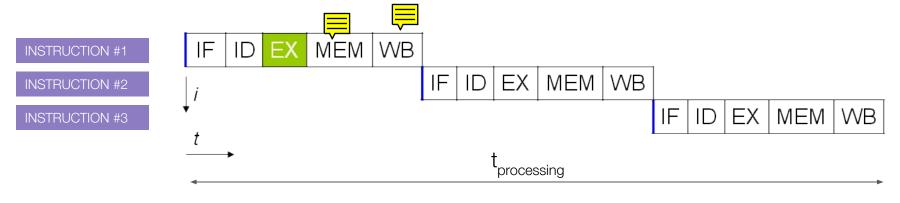
PROCESSING TIME



The **CPU processing time** is defined as the time needed to execute the whole set of instructions

This depends on:

- the complexity of the instruction set
- the CPU clock frequency
- the overheads induced by the communication with registers and memory



How can processing performances be improved?

 \rightarrow i.e. how can we reduce the overall execution time of a given process?

PROCESSING TIME



The CPU processing time is defined as the time needed to execute the whole set of

instr

More on the "processing time" (still, an extremely simplified view of it)

This

Wall-time (or response time, or elapsed time)

Total latency to perform a task, including all IO operations (memory and storage access), communication over network, OS overhead, ...

CPU time

Latency due <u>only</u> to the instruction processing by the CPU, excluding the time waited performing other activities

INSTE

INST

 $t_{CPU} = \Sigma_i (IC_i \cdot CPI_i) / CF$

Instruction count

→ number of instruction per program

CPI: Clock per instruction

→ number of clock cycles per instruction

How CF: Clock frequency

→ 1/time per clock cycle

TIOW CAIT WE FEGUCE THE OVERALL EXECUTION TILLE OF A GIVEN PI

POWER WALL



An easy solution to boost the performances might appear to simply rise from an increase in Clock Frequency.

This (together with other advancements) have been the case for many years.

However, with the 2000s, the chips have started to reach a technological limitation related to the power dissipation

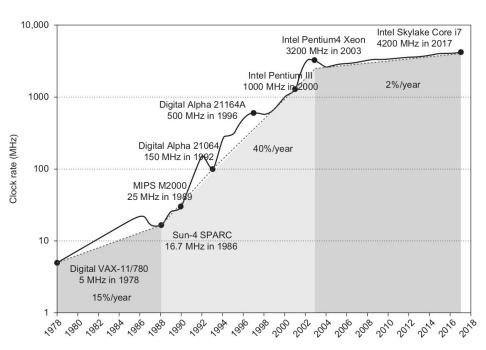
$P \propto n C V^2 f$

n number of transistors

C transistor capacity

V transistor voltage

f clock frequency



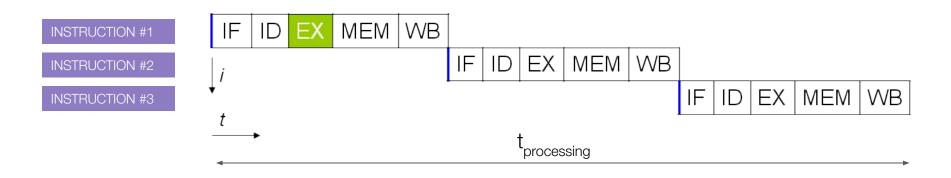
The first 32-bit microprocessors (such as the Intel 80386) consumed about 2 W, whereas a 2021 4.0 GHz Intel Xeon Gold 5318Y consumes more than 150 W!

SHIFT IN COMPUTER ARCHITECTURE



There really is no free lunch... not even in computing architectures!

As performance boosts cannot rely solely on increase in clock frequency, alternative strategies were designed, aimed at increasing the number of instructions a chip can run per second.



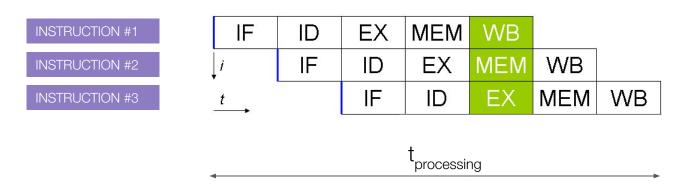
PIPELINING



There really is no free lunch... not even in computing architectures!

As performance boosts cannot rely solely on increase in clock frequency, alternative strategies were designed, aimed at increasing the number of instructions a chip can run per second.

A speedup is commonly obtained in CPUs with the *Instruction Pipelining*, enabling multiple instructions to be run *concurrently* on the same ALU

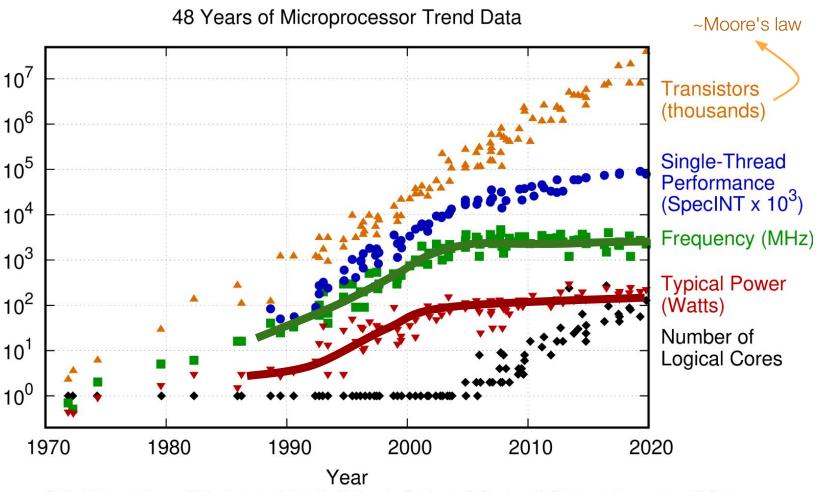


In order to achieve better performance, a change of paradigm is required



PROCESSING GROWTH

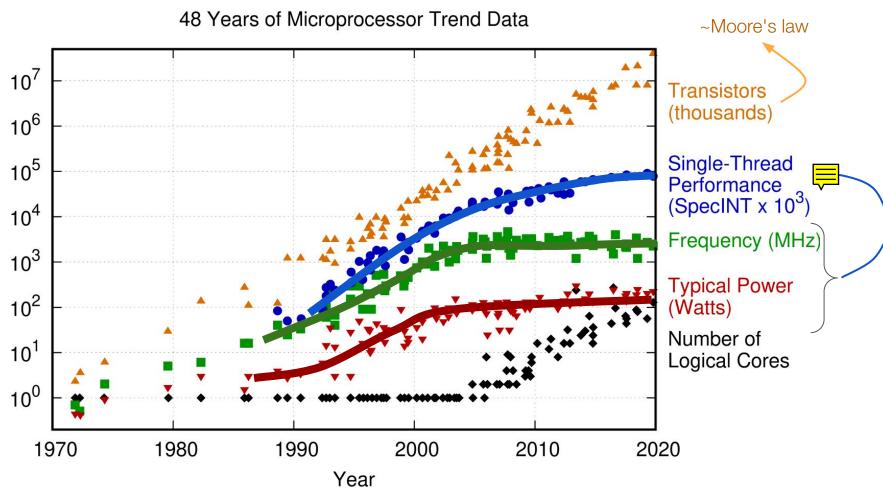




Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten New plot and data collected for 2010-2019 by K. Rupp

PROCESSING GROWTH

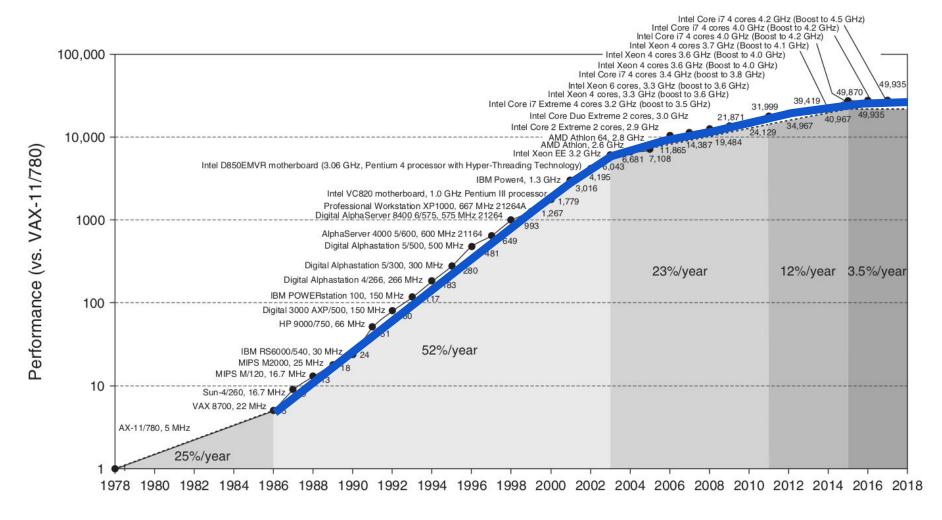




Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten New plot and data collected for 2010-2019 by K. Rupp

PROCESSING GROWTH

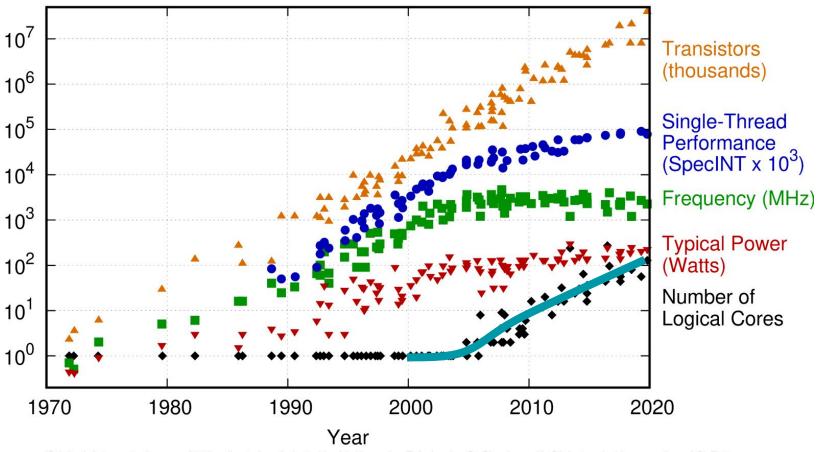




MULTI-PROCESSING ARCHITECTURES



48 Years of Microprocessor Trend Data



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten New plot and data collected for 2010-2019 by K. Rupp

MULTI-PROCESSING ARCHITECTURES



Multiple PUs per CPU ("multi-core CPUs") and/or multiple CPUs per machine

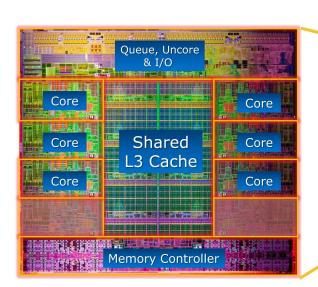
Can be realized in a variety of ways:

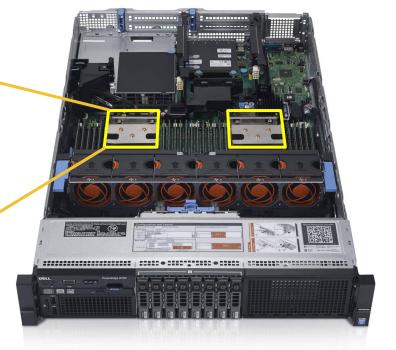
Symmetric MultiProcessingAsymmetric MultiProcessing

Master/Slave



- → Heterogeneous CPUs architecture
- → Master CPUs in charge of assigning tasks to the others









```
pazzini@pazzini-x1$ lscpu
Architecture:
                                  x86 64
                                  32-bit, 64-bit
CPU op-mode(s):
Byte Order:
                                  Little Endian
Address sizes:
                                  39 bits physical, 48 bits virtual
CPU(s):
                                   8
                                  0 - 7
On-line CPU(s) list:
Thread(s) per core:
Core(s) per socket:
Socket(s):
NUMA node(s):
Vendor ID:
                                  GenuineIntel
CPU family:
Model:
                                  142
Model name:
                                  Intel(R) Core(TM) i7-10510U CPU @ 1.80GHz
                                  12
Stepping:
                                   2300.000
CPU MHz:
                                  4900,0000
CPU max MHz:
CPU min MHz:
                                   400,0000
                                  4599.93
BogoMIPS:
Virtualization:
                                  VT-X
Lld cache:
                                  128 KiB
Lli cache:
                                  128 KiB
L2 cache:
                                  1 MiB
L3 cache:
                                   8 MiB
```



```
pazzini@pazzini-x1$ lscpu
Architecture:
                                  x86_64 — CPU Architecture (instruction set)
CPU op-mode(s):
                                  32-bit, 64-bit
Byte Order:
                                  Little Endian
Address sizes:
                                  39 bits physical, 48 bits virtual
CPU(s):
                                  8
                                  0 - 7
On-line CPU(s) list:
Thread(s) per core:
Core(s) per socket:
Socket(s):
NUMA node(s):
Vendor ID:
                                  GenuineIntel
CPU family:
Model:
                                  142
Model name:
                                  Intel(R) Core(TM) i7-10510U CPU @ 1.80GHz
                                  12
Stepping:
                                  2300.000
CPU MHz:
                                  4900,0000
CPU max MHz:
CPU min MHz:
                                  400,0000
                                  4599.93
BogoMIPS:
Virtualization:
                                  VT-x
Lld cache:
                                  128 KiB
Lli cache:
                                  128 KiB
L2 cache:
                                  1 MiB
L3 cache:
                                  8 MiB
```



```
pazzini@pazzini-x1$ lscpu
Architecture:
                                   x86 64
CPU op-mode(s):
                                   32-bit, 64-bit
Byte Order:
                                   Little Endian
Address sizes:
                                   39 bits physical, 48 bits virtual
CPU(s):
                                   8
                                   0 - 7
On-line CPU(s) list:
Thread(s) per core:
Core(s) per socket:
Socket(s):
                                                            1 CPU socket on the motherboard
NUMA node(s):
Vendor ID:
                                   GenuineIntel
CPU family:
Model:
                                   142
Model name:
                                   Intel(R) Core(TM) i7-10510U CPU @ 1.80GHz
                                   12
Stepping:
                                   2300.000
CPU MHz:
                                   4900,0000
CPU max MHz:
CPU min MHz:
                                   400,0000
                                   4599.93
BogoMIPS:
Virtualization:
                                  VT-X
L1d cache:
                                   128 KiB
Lli cache:
                                  128 KiB
L2 cache:
                                   1 MiB
L3 cache:
                                   8 MiB
```



```
pazzini@pazzini-x1$ lscpu
Architecture:
                                   x86 64
CPU op-mode(s):
                                   32-bit, 64-bit
Byte Order:
                                   Little Endian
Address sizes:
                                   39 bits physical, 48 bits virtual
CPU(s):
                                   8
                                   0 - 7
On-line CPU(s) list:
Thread(s) per core:
                                                             4 Physical PUs (cores) in the/each CPU
Core(s) per socket:
                                                             1 CPU socket on the motherboard
Socket(s):
NUMA node(s):
Vendor ID:
                                   GenuineIntel
CPU family:
Model:
                                   142
Model name:
                                   Intel(R) Core(TM) i7-10510U CPU @ 1.80GHz
                                   12
Stepping:
                                   2300.000
CPU MHz:
                                   4900,0000
CPU max MHz:
CPU min MHz:
                                   400,0000
                                   4599.93
BogoMIPS:
Virtualization:
                                   VT-X
L1d cache:
                                   128 KiB
Lli cache:
                                   128 KiB
L2 cache:
                                   1 MiB
L3 cache:
                                   8 MiB
```



```
pazzini@pazzini-x1$ lscpu
                                    x86 64
Architecture:
CPU op-mode(s):
                                    32-bit, 64-bit
Byte Order:
                                    Little Endian
Address sizes:
                                    39 bits physical, 48 bits virtual
CPU(s):
                                    8
On-line CPU(s) list:
                                    0 - 7
                                                               Number of Logical PUs per core
Thread(s) per core:
                                                              4 Physical PUs (cores) in the/each CPU
Core(s) per socket:
                                                               1 CPU socket on the motherboard
Socket(s):
NUMA node(s):
Vendor ID:
                                    GenuineIntel
CPU family:
Model:
                                    142
Model name:
                                    Intel(R) Core(TM) i7-10510U CPU @ 1.80GHz
                                    12
Stepping:
                                                               (Intel) Hyper-threading technology
                                    2300.000
CPU MHz:
                                    4900,0000
CPU max MHz:
                                                              2 independent "feeding lanes" per CPU.
CPU min MHz:
                                    400,0000
                                    4599.93
                                                               Fetch and decode 2 instructions, but can
BogoMIPS:
Virtualization:
                                    VT-x
                                                              execute only 1 at a time (1 ALU)
Lld cache:
                                    128 KiB
Lli cache:
                                    128 KiB
                                                                        Fetch/
                                                                                     Fetch/
L2 cache:
                                    1 MiB
                                                                        Decode
                                                                                    Decode
L3 cache:
                                    8 MiB
                                                                               ALU
```

(Execution Unit)



```
pazzini@pazzini-x1$ lscpu
Architecture:
                                      x86 64
CPU op-mode(s):
                                      32-bit, 64-bit
Byte Order:
                                      Little Endian
Address sizes:
                                      39 bits physical, 48 bits virtual
                                                                  n. Sockets
CPU(s):
                                                                                         X
On-line CPU(s) list:
                                      0 - 7
                                                                  n. Cores per Socket
                                                                                         Χ
Thread(s) per core:
                                                                  n. Threads per core
Core(s) per socket:
Socket(s):
                                                                  Total number of PUs as "seen"
NUMA node(s):
                                                                  by the operating system
Vendor ID:
                                      GenuineIntel
CPU family:
Model:
                                      142
Model name:
                                      Intel(R) Core(TM) i7-10510U CPU @ 1.80GHz
                                      12
Stepping:
                                      2300.000
CPU MHz:
                                      4900,0000
CPU max MHz:
                                                                                                           2.1%]
                                                                           12.6%
CPU min MHz:
                                      400,0000
                                                                                                           2.6%]
                                      4599.93
BogoMIPS:
                                                                                                           3.3%
                                                                                                           4.2%]
Virtualization:
                                      VT-x
                                                                                 Tasks: 278, 2150 thr; 1 running
                                                                       13.0G/15.3G1
Lld cache:
                                      128 KiB
                                                                       5.26G/16.0G
                                                                                 Load average: 1.08 1.34 1.43
                                                                                 Uptime: 22 days, 07:00:38
Lli cache:
                                      128 KiB
L2 cache:
                                      1 MiB
L3 cache:
                                      8 MiB
```

VS A SERVER



```
pazzini@PowerEdge-R750:~$ lscpu
Architecture:
                                      x86 64
CPU op-mode(s):
                                      32-bit, 64-bit
Byte Order:
                                      Little Endian
Address sizes:
                                      46 bits physical, 57 bits virtual
CPU(s):
                                      96
On-line CPU(s) list:
                                      0 - 95
Thread(s) per core:
Core(s) per socket:
                                      2.4
Socket(s):
NUMA node(s):
Vendor ID:
                                      GenuineIntel
CPU family:
Model:
                                      106
Model name:
                                      Intel(R) Xeon(R) Gold 5318Y CPU @ 2.10GHz
Stepping:
                                      6
                                      2100.000
CPU MHz:
BogoMIPS:
                                      4200.00
Virtualization:
                                      VT-x
L1d cache:
                                      2,3 MiB
Lli cache:
                                      1,5 MiB
                                      60 MiB
L2 cache:
L3 cache:
                                      72 MiB
```

Multi-processing machines are currently quite common... how can we leverage on them?

PARALLELISM



Parallelism refers to the idea of exploiting multi-processing architectures in order to solve complex problems.

There are mainly two kinds of parallelism:

1. Data-level parallelism (DLP)

Many data items (records/elements) that can be operated on at the same time.

 \rightarrow i.e. split the data and run a task on all elements in parallel

2. Task-level parallelism (TLP)

Multiple sub-tasks (or *threads*) of a process that can operate independently and *largely* in parallel.

 \rightarrow i.e. split the task into sub-tasks and run them in parallel

The two are not mutually exclusive!

Very often the most efficient way to solve a problem is to exploit both DLP and TLP

PARALLEL PROCESSING



Simultaneous execution of a single job by subdividing it in **n** sub-tasks, executed across **m** CPUs/cores

→ The *main* goal is to reduce the computing time (*speedup*) of the job

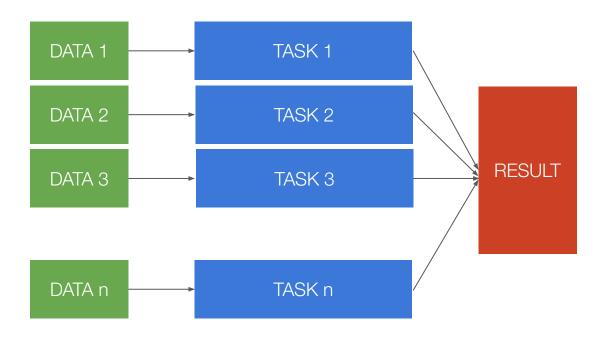


PARALLEL PROCESSING



Simultaneous execution of a single job by subdividing it in **n** sub-tasks, executed across **m** CPUs/cores

→ The *main* goal is to reduce the computing time (*speedup*) of the job



Subtasks could be either identical or very different, depending on the type of parallelism and computing architecture



Architecture classification developed in the mid-60s based on the **number of**

- simultaneously-executed instructions
- parallel data streams processed

SISD (SINGLE INSTRUCTION SINGLE DATA)

MISD
(MULTIPLE INSTRUCTION SINGLE DATA)

SIMD (SINGLE INSTRUCTION MULTIPLE DATA)

MIMD (MULTIPLE INSTRUCTION MULTIPLE DATA)



SISD (SINGLE INSTRUCTION SINGLE DATA)

SIMD (SINGLE INSTRUCTION MULTIPLE DATA)

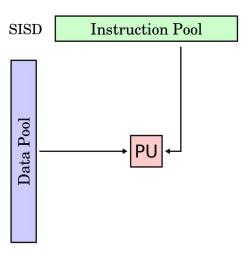
MISD (MULTIPLE INSTRUCTION SINGLE DATA)

MIMD (MULTIPLE INSTRUCTION MULTIPLE DATA)

One instruction on one data stream at a time

No parallelism at all

e.g.: 1 single-threaded process in execution on a single-core PUs





SISD (SINGLE INSTRUCTION SINGLE DATA)

SIMD (SINGLE INSTRUCTION MULTIPLE DATA)

MISD (MULTIPLE INSTRUCTION SINGLE DATA)

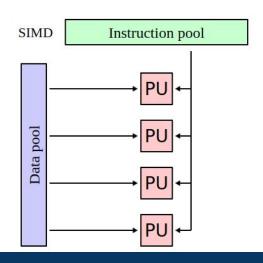
MIMD (MULTIPLE INSTRUCTION MULTIPLE DATA)

One *instruction* performed on multiple data streams at the same time

Enables data-parallelism (DLP)

Instructions can be further pipelined

e.g.: specific instructions (avx/mmx/sse) of modern CPUs, most GPUs, specific "vector" SuperComputer





SISD (SINGLE INSTRUCTION SINGLE DATA)

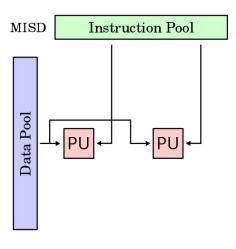
SIMD (SINGLE INSTRUCTION MULTIPLE DATA)

MISD (MULTIPLE INSTRUCTION SINGLE DATA)

MIMD (MULTIPLE INSTRUCTION MULTIPLE DATA)

Multiple *instructions* performed on a single data stream

Very unusual in common data processing, but dedicated to special applications





SISD (SINGLE INSTRUCTION SINGLE DATA)

SIMD (SINGLE INSTRUCTION MULTIPLE DATA)

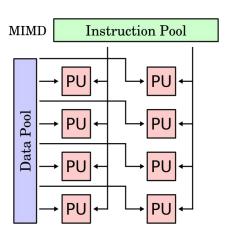
MISD (MULTIPLE INSTRUCTION SINGLE DATA)

MIMD (MULTIPLE INSTRUCTION MULTIPLE DATA)

Multiple instructions performed on multiple data streams at the same time

Flexible parallelism (both DLP and TLP)

Each PU runs independently and (possibly) "asynchronously" from the others





SISD (SINGLE INSTRUCTION SINGLE DATA)

SIMD (SINGLE INSTRUCTION MULTIPLE DATA)

MISD (MULTIPLE INSTRUCTION SINGLE DATA)

MIMD (MULTIPLE INSTRUCTION MULTIPLE DATA)

Multiple instructions performed on multiple data streams at the same time

Flexible parallelism (both DLP and TLP)

Each PU runs independently and (possibly) "asynchronously" from the others

A sub-category of MIMD is **SPMD**→ **SINGLE PROGRAM MULTIPLE DATA**

Widely used to run the same entire program (not just single instructions or threads) on multiple "chunks" of data using several PUs

⇒ Embarrassingly parallel problems



Is it possible to speedup the computation time indefinitely by adding more PUs?

PROCESSING TIME	
SERIAL	PARALLELIZABLE

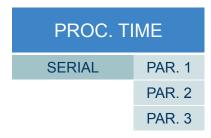
1 PU ⇒ no parallelization



Is it possible to speedup the computation time indefinitely by adding more PUs?

PROCESSING TIME	
SERIAL	PARALLELIZABLE

1 PU ⇒ no parallelization



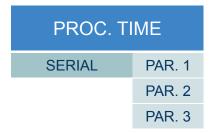
$$n PUs \Rightarrow speedup = \frac{time_{1 PU}}{time_{n PUs}}$$



Is it possible to speedup the computation time indefinitely by adding more PUs?

PROCESSING TIME SERIAL PARALLELIZABLE

1 PU ⇒ no parallelization



n PUs
$$\Rightarrow$$
 speedup = $\frac{1}{(1-p)+\frac{p}{n}}$

Amdahl's law

p : fraction of parallelizable processing

(1-p): fraction of serial processingn : number of processing units



Is it possible to speedup the computation time indefinitely by adding more PUs?

PROCESSING TIME

SERIAL

PARALLELIZABLE

PROC. TIME

SERIAL PAR. 1

PAR. 2

PAR. 3

PROC. TIME

SERIAL

. . .

1 PU ⇒ no parallelization

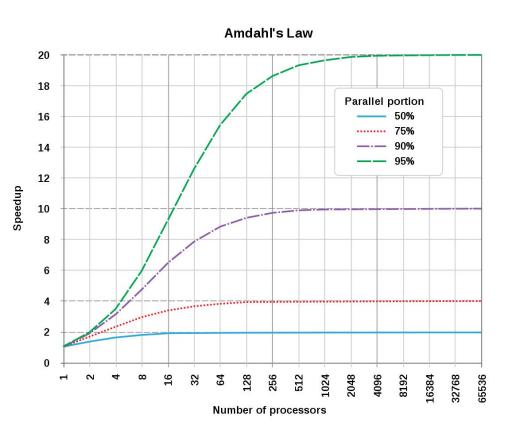
n PUs
$$\Rightarrow$$
 speedup = $\frac{1}{(1-p) + \frac{p}{n}}$

$$n \rightarrow \infty \text{ PUs} \Rightarrow speedup \xrightarrow{n \rightarrow \infty} \frac{1}{1-p}$$

AMDAHL'S LAW







No matter how many PUs can be used to parallelize the task execution, the main limitation is still going to be the serial fraction of the process

⇒ We need to re-think and optimize the process to maximize its parallel fraction

Amdahl's law is often considered quite "pessimistic"

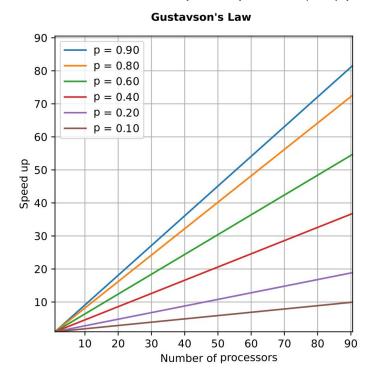
It is based on the assumption that as a problem increase in size, its serial fraction increases proportionally.

GUSTAFSON'S LAW





Gustavson's scaled speedup = 1 + (n-1) p



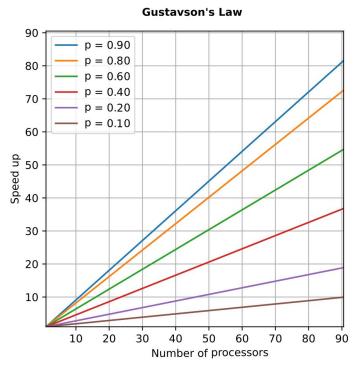
Gustafson's law addresses the advantages of parallel programming from an alternative point of view.

Often, the "problem size" scales with the number of available processors, thus the parallelizable part of a computation grows more than the serial component.

GUSTAFSON'S LAW



Gustavson's scaled speedup = 1 + (n-1) p



Gustafson's law addresses the advantages of parallel programming from an alternative point of view.

Often, the "problem size" scales with the number of available processors, thus the parallelizable part of a computation grows more than the serial component.

Either ways...

Heterogeneous computing (integrating CPUs + GPUs + ...) aims to get the best of both worlds by speeding up both the serial and parallel components.

