01.Introduction

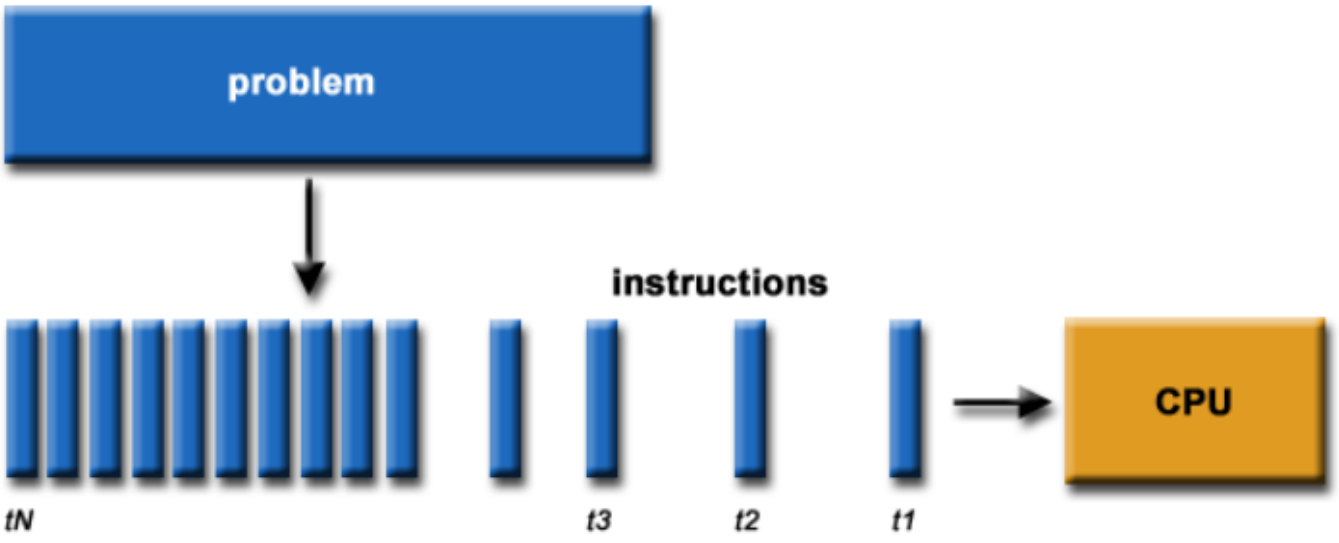
Flynn Taxonomy

- SISD: Single Instruction Single Data
  - Uniprocessor systems
- MISD: Multiple Instruction Single Data
  - No practical configuration and no commercial systems
- SIMD: Single Instruction Multiple Data
  - Simple programming model, low overhead, flexibility, custom integrated circuits
- MIMD: Multiple Instruction Multiple Data
  - Scalable, fault tolerant, off-the-shelf micros

Traditional" Computation

Software is written for serial computation

- It has to be executed on a single computer having a single Central Processing Unit (CPU)
- A problem is broken into a discrete series of instructions
- Instructions are executed one after another
- Only one instruction may execute at any moment in time



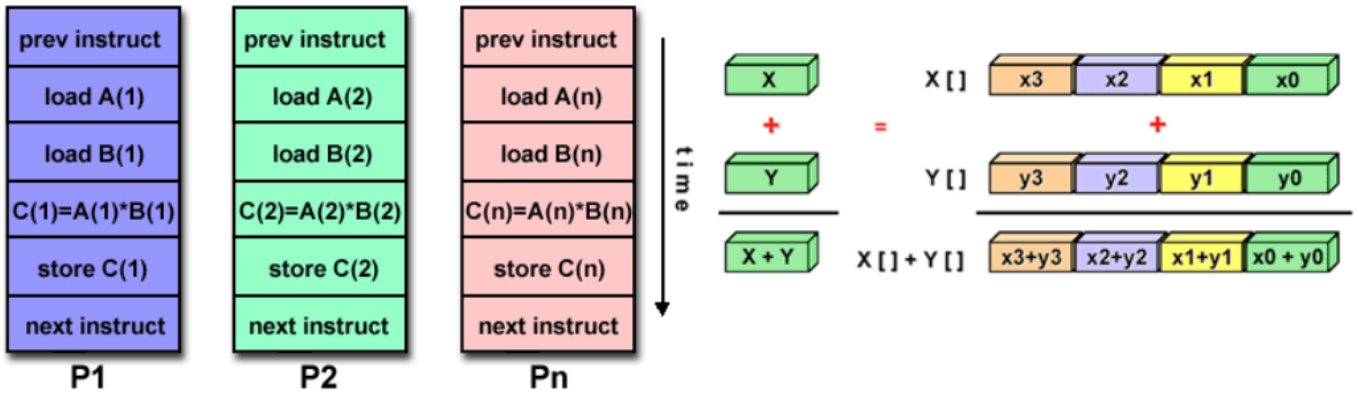
SISD

A serial (non-parallel) computer
Single instruction: only one instruction stream is being acted on by the CPU during any one clock cycle
Single data: only one data stream is being used as input during any one clock cycle

Deterministic execution
This is the oldest and even today, the most common type of computer

SIMD

A type of parallel computer
Single instruction: all processing units execute the same instruction at any given clock cycle
Multiple data: each processing unit can operate on a different data element
Best suited for specialized problems characterized by a high degree of regularity, such as graphics/image processing

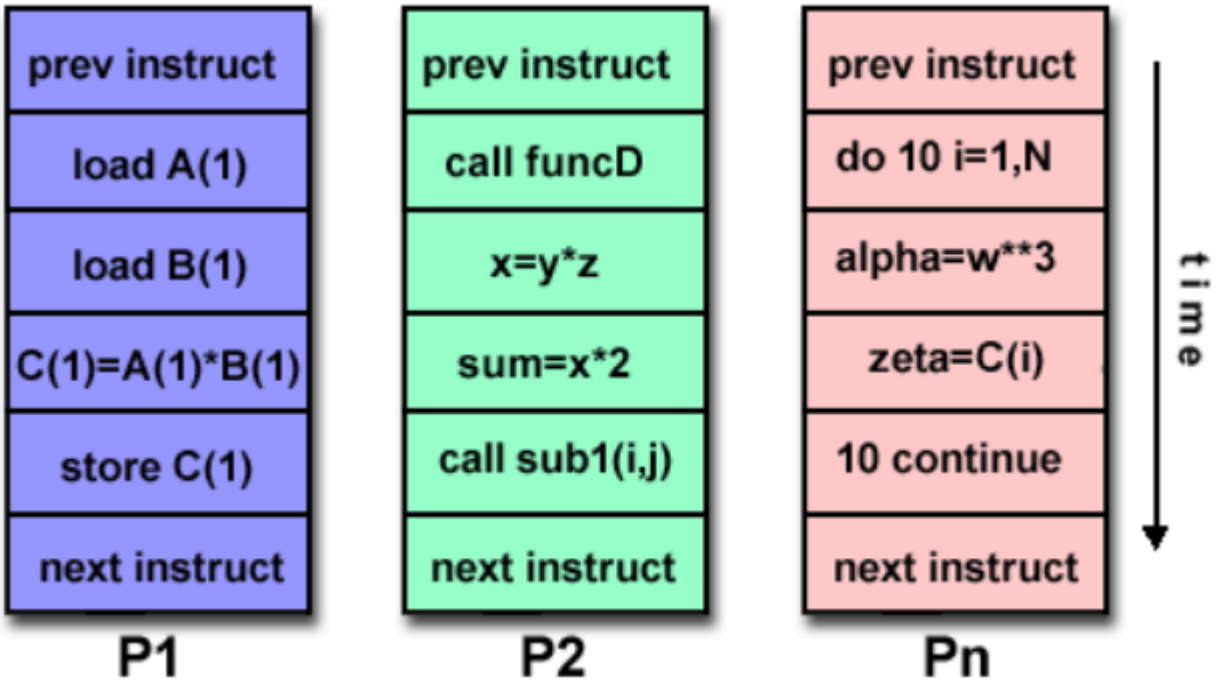


Hardware parallelism

- Instruction-Level Parallelism: Exploits data-level parallelism at modest level through compiler techniques such as pipelining and at medium levels using speculation
- Vector architectures and Graphic Processor Units: Exploit data-level parallelism by applying a single instruction to a collection of data in parallel
- Thread-level parallelism: Exploits either data-level parallelism or task-level parallelism in a tightly coupled hw model that allows interaction among threads
- Request-level parallelism: Exploits parallelism among largely decoupled tasks specified by the programmer or the OS

MIMD

Nowadays, the most common type of parallel computer
Multiple Instruction: every processor may be executing a different instruction stream
Multiple Data: every processor may be working with a different data stream
Execution can be synchronous or asynchronous, deterministic or non-deterministic


prev instruct
load A(1)
load B(1)
C(1)=A(1)*B(1)
store C(1)
next instruct
P1
prev instruct
call funcD
x=y*z
sum=x*2
call sub1(i,j)
next instruct
P2
prev instruct
do 10 i=1,N
alpha=w**3
zeta=C(i)
10 continue
next instruct
Pn
time