Parallel languages as extensions of sequential ones

Alexey A. Romanenko
arom@ccfit.nsu.ru

What this section about?

- Computers. History. Trends.
- What is parallel program?
- What is parallel programming for?
- Features of parallel programs.
- Development environment.
- etc.

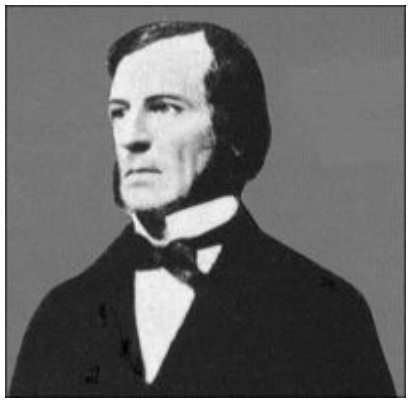
Agenda

1. Sequential program
2. Applications, required computational power.
3. What does parallel programming for?
4. Parallelism inside ordinary PC.
5. Architecture of modern CPUs.
6. What is parallel program?
7. Types of parallelism.

Agenda

8. Types of computational installations.
9. Specificity of parallel programs.
10. Amdahl's law
11. Development environment
12. Approaches to development of parallel programs. Cost of development.
13. Self-test questions

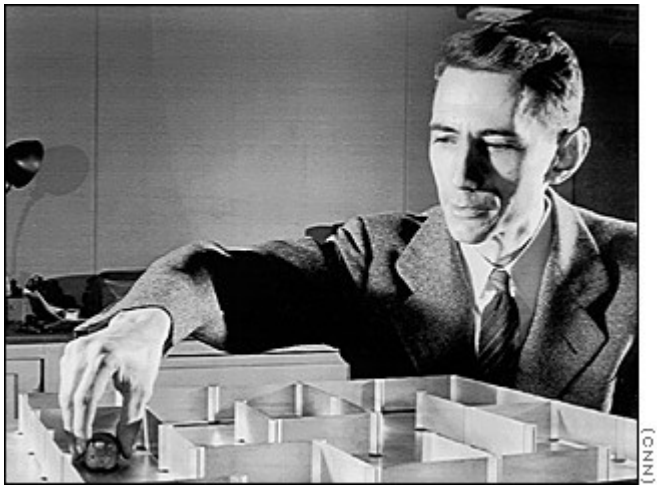
History



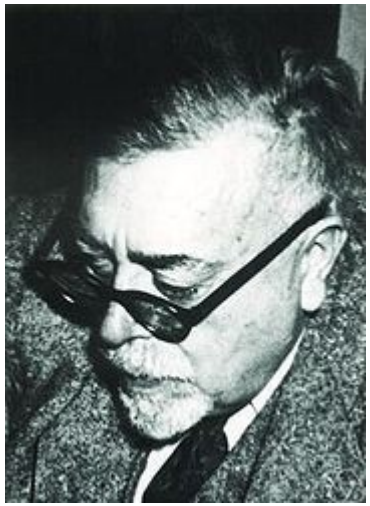
George Boole

Charles Babbage

Alan Turing



Claude Elwood Shannon



Norbert Wiener

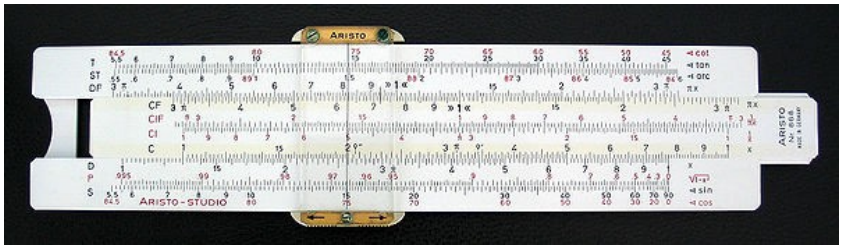
Henry Edward Roberts

John von Neumann

Sciences

- Computer science is the study of the theoretical foundations of information and computation, and of practical techniques for their implementation and application in computer systems.
- Cybernetics is the interdisciplinary study of the structure of regulatory system

Difference machine



Arithmometer

Altair 8800 Computer with
8-inch floppy disk system

Sequential program

A program perform calculation of a function
F = G(X)
for example:

$a*x^2+b*x+c=0, \quad a \neq 0.$

$x1=(-b-\sqrt{(b^2-4ac)})/(2a),$
$x2=(-b+\sqrt{(b^2-4ac)})/(2a)$

Turing machine

Plasma modeling

$$N \sim 10^6$$

$$dX_j \sim F_j \, dT^2$$

$$F_j \sim \mathrm{sum}_i(q_i, q_j)$$

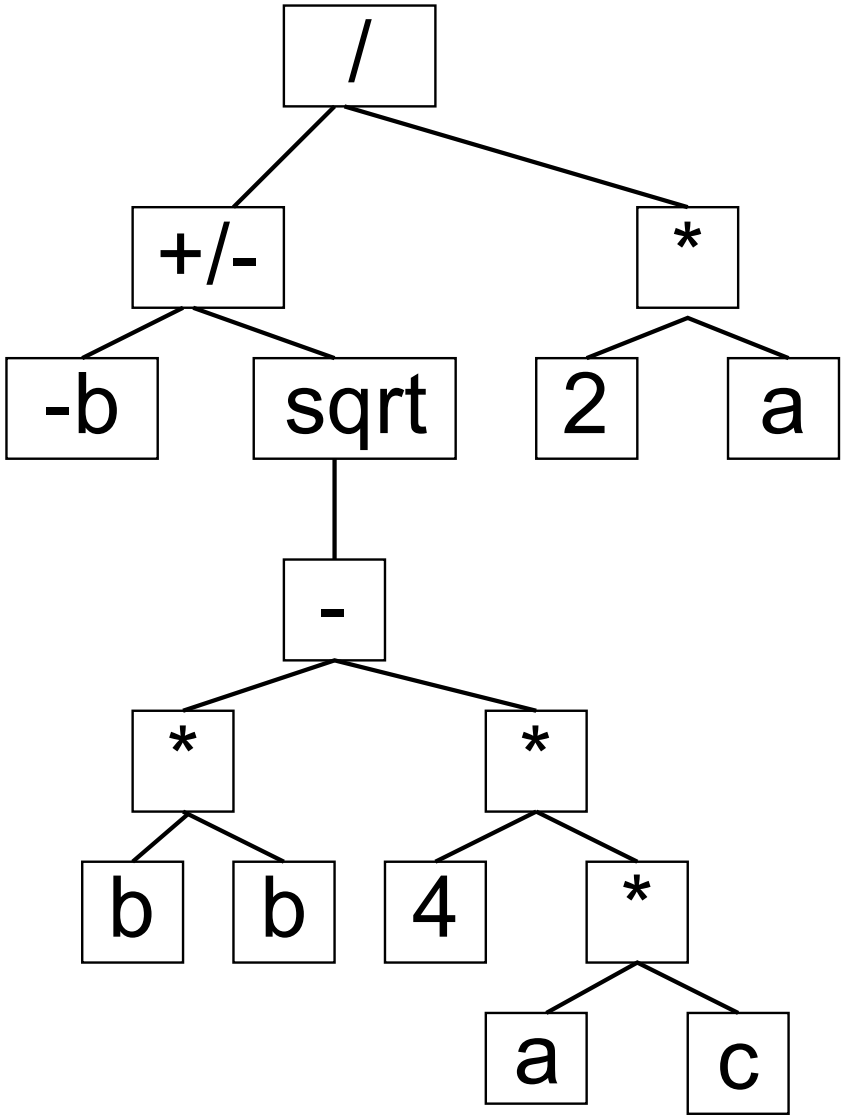
Complexity ~ O(N*N)

more then 10^{12} * 100...1000 operations

Resource consumable calculations

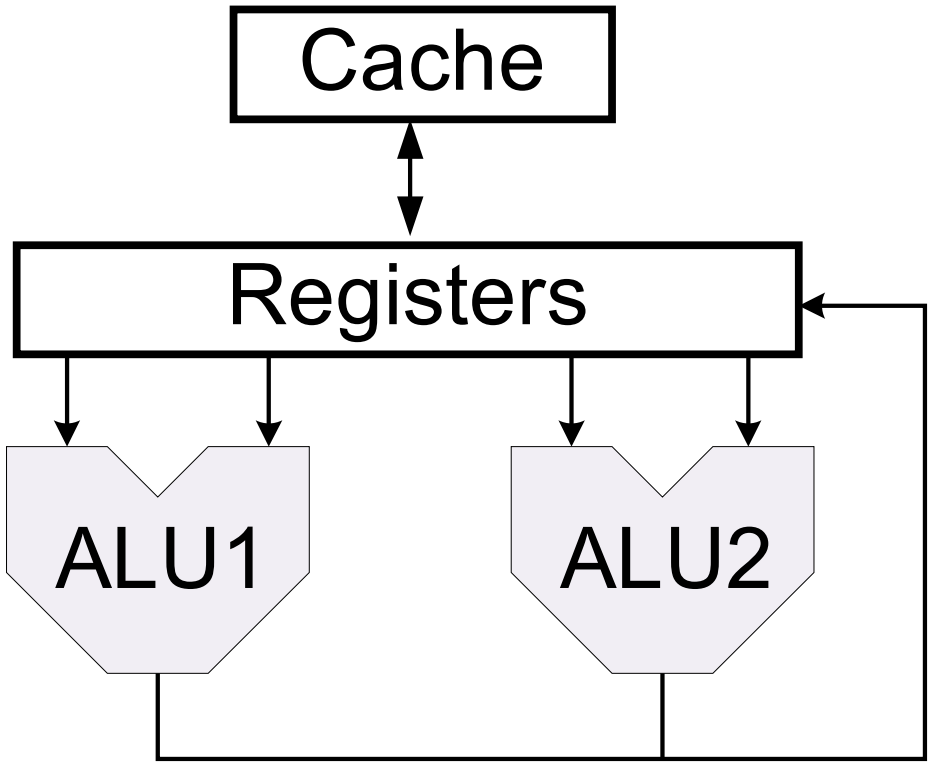
- Nuclear/Gas/Hydrodynamic physics
- Bio-informatics
- Chemical modeling
- Oil&Gas drilling
- Medicine
- Signal processing
- etc.

Parallel program

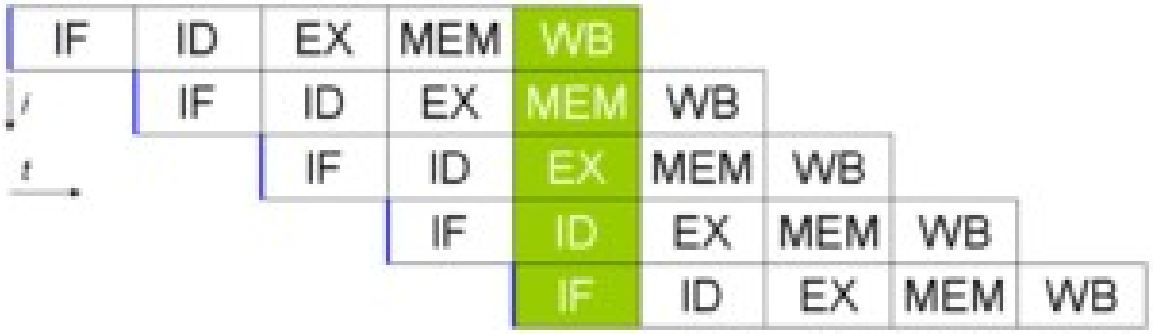


PP is a program which allows computational environment to do some operations in parallel

Instruction parallelism



Instruction parallelism



IF – Fetch instruction
ID – decode instruction
EX – Execute instruction
MEM – Memory access
WB – Write result back (mem/reg)

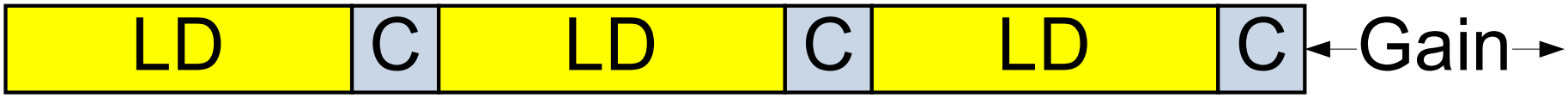
Vector operations

X1	X2	X3	X4
+			
Y1	Y2	Y3	Y4
=			
X1+Y1	X2+Y2	X3+Y3	X4+Y4

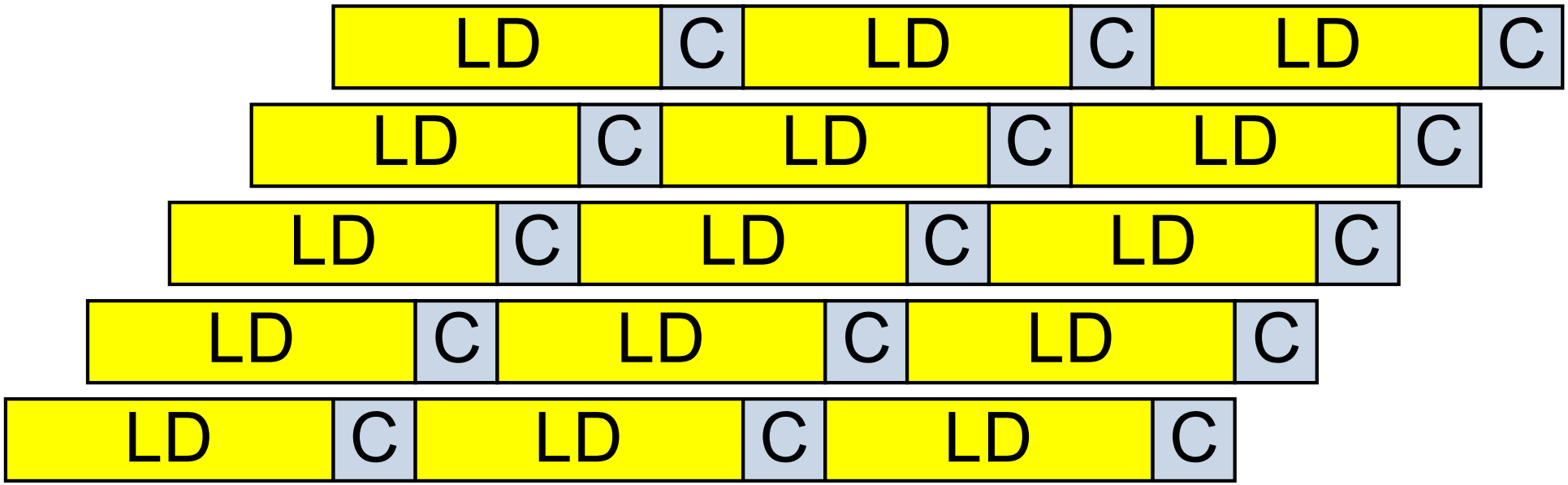
MMX, 3DNow, SSE, SSE2

Data loading

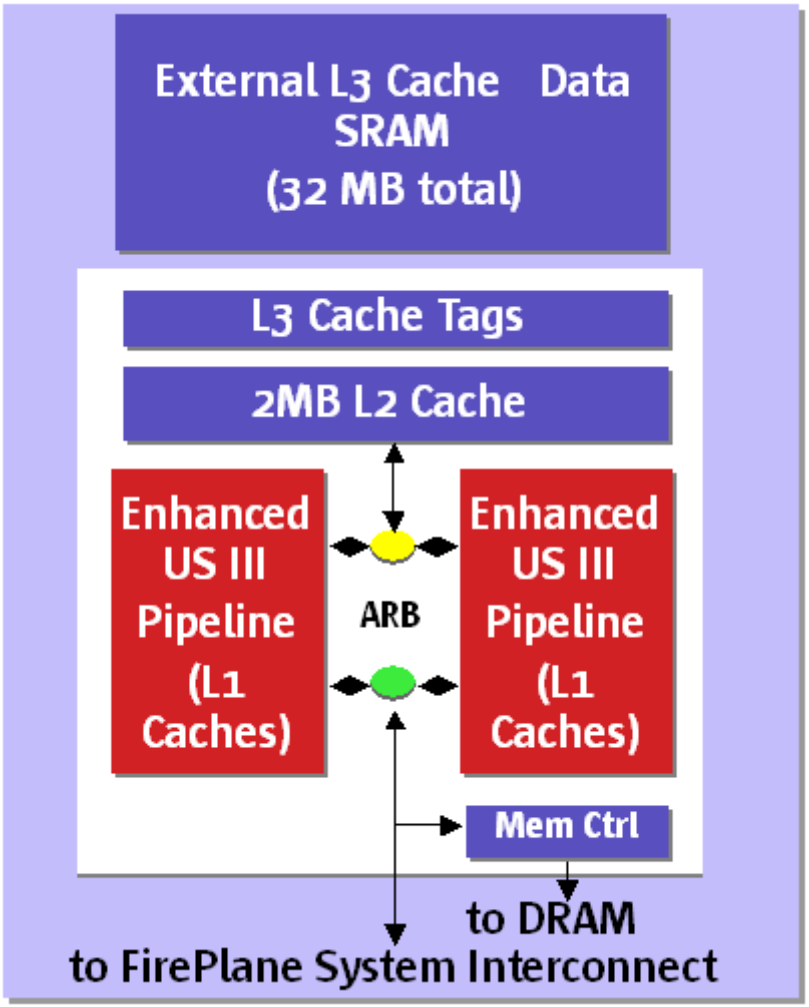
Calc. 2 times faster

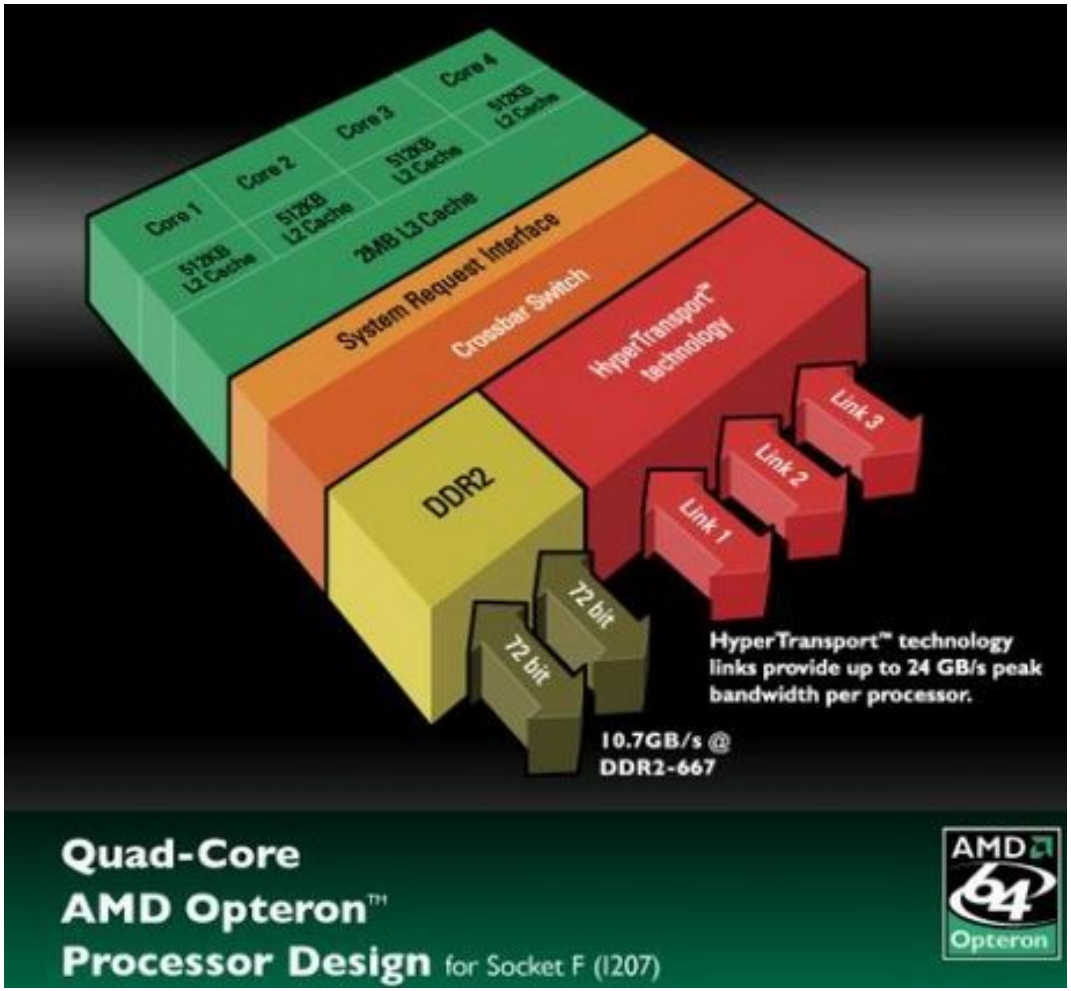


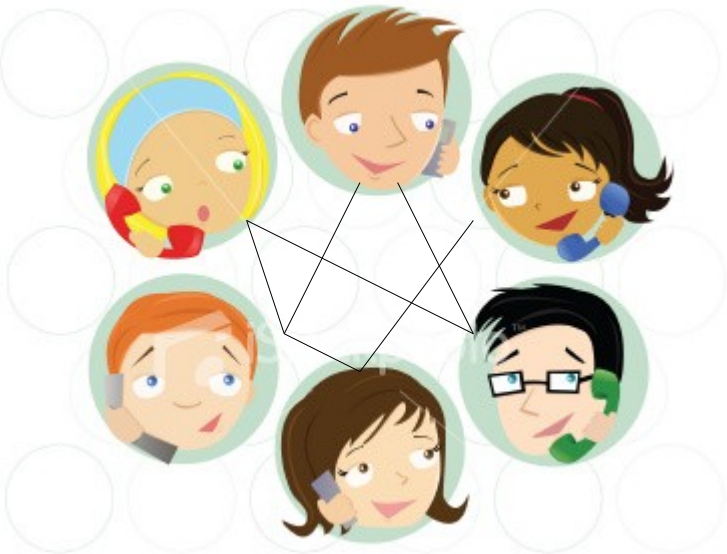
Data loading



Multi-core


External L3 Cache Data SRAM
(32 MB total)
L3 Cache Tags
2MB L2 Cache
Enhanced US III Pipeline (L1 Caches)
ARB
Enhanced US III Pipeline (L1 Caches)
Mem Ctrl
to DRAM
to FirePlane System Interconnect


Core 1
Core 2
Core 3
Core 4
512KB L2 Cache
2MB L3 Cache
System Request Interface
Crossbar Switch
HyperTransport™ technology
DDR2
72 bit
72 bit
Link 1
Link 2
Link 3
HyperTransport™ technology links provide up to 24 GB/s peak bandwidth per processor.
10.7GB/s @ DDR2-667
Quad-Core AMD Opteron™ Processor Design for Socket F (1207)
AMD 64 Opteron



Parallel program is a system of communicated processes

Types of parallelism

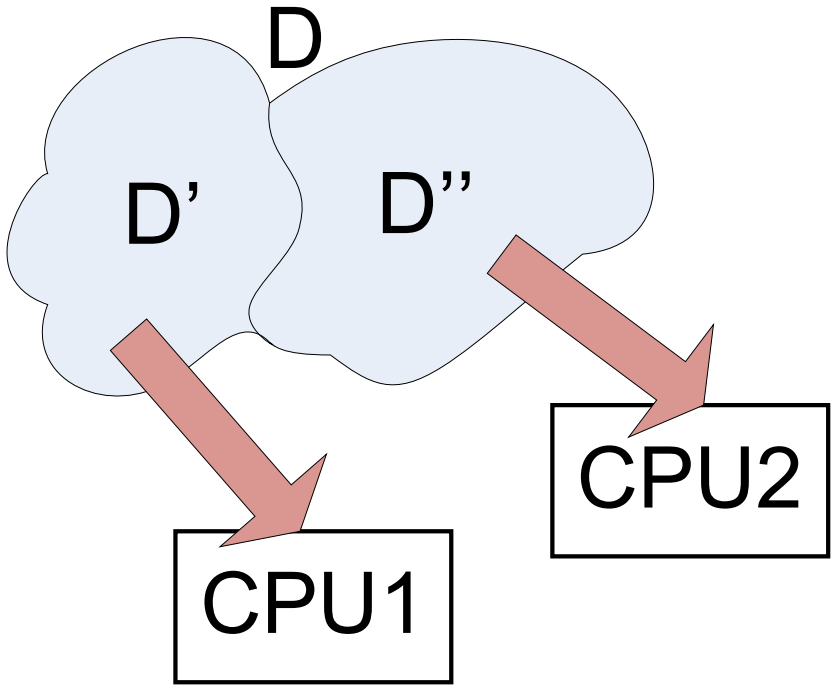
- Bit-level parallelism
- Instruction level parallelism
- Data parallelism
- Task parallelism

Bit-level parallelism

Increasing the word size reduces the number of instructions the processor must execute in order to perform an operation on variables whose sizes are greater than the length of the word.

Historically, 4-bit microprocessors were replaced with 8-bit, then 16-bit, then 32-bit microprocessors. This trend generally came to an end with the introduction of 32-bit processors, which has been a standard in general purpose computing for two decades. Only recently, with the advent of x86-64 architectures, have 64-bit processors become commonplace.

Data parallelism

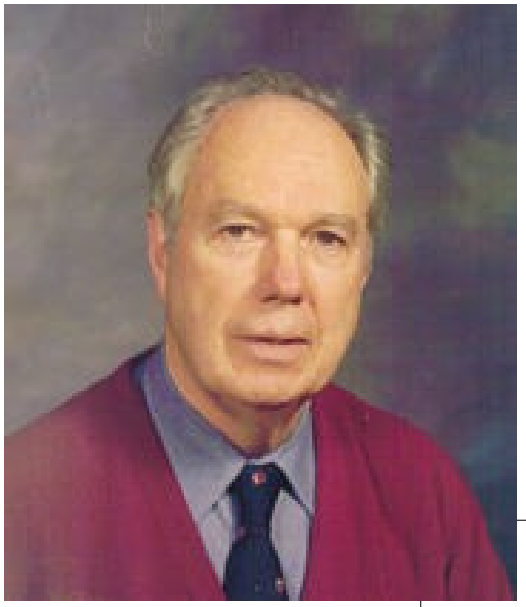


There is a set of data **D**.
For each X_i from **D** do $F(X_i)$

D could be distributed over computational nodes
D = **D**' U **D**", **D**' ∩ **D**" = 0
Process **D**' on CPU1, process **D**" on CPU2 in parallel

Task parallelism

Task parallelism is the characteristic of a parallel program that "*entirely different calculations can be performed on either the same or different sets of data*". This contrasts with data parallelism, where the same calculation is performed on the same or different sets of data. Task parallelism does not usually scale with the size of a problem.



Flinn's taxonomy

	Single instruction	Multiple instruction
Single data	SISD	MISD
Multiple data	SIMD	MIMD

Flinn's taxonomy

SISD - sequential PC, which performs operations one by one

SIMD – vector computers, vector operations like SSE, MMX.

MISD – strange type. It's hardly possible to find any PC with this type of CPUs. One can suggest to look at pipeline as MISD systems.

MIMD – PC (set of PCs) which can execute several different programs at the same time.

Classes of parallel computers

- Multicore computing
- Symmetric multiprocessing
- Distributed computing
- Cluster computing
- Massive parallel processing
- Grid computing

Specialized parallel computers

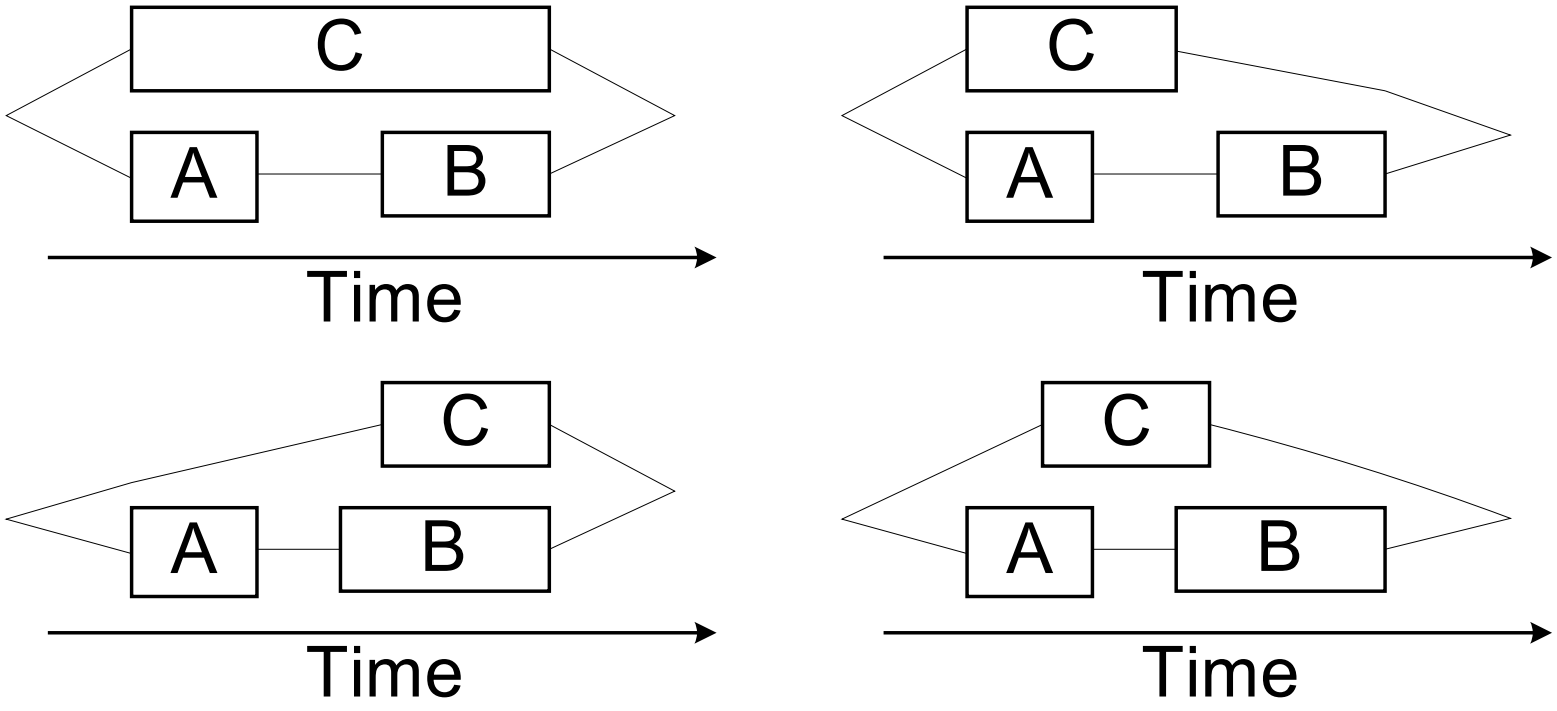
- Reconfigurable computing with field-programmable gate arrays
- GPGPU with graphics processing units
- Application-specific integrated circuits
- Vector processors

Specificity of parallel programs

- Nondeterminism
- Errors
  - Deadlocks
  - Race conditions
- Scalability

Nondetermenism

Nondeterminism is a specificity of parallel program which tells that sometimes it is impossible to say which function/process start or finish its execution first.



Errors. Race condition

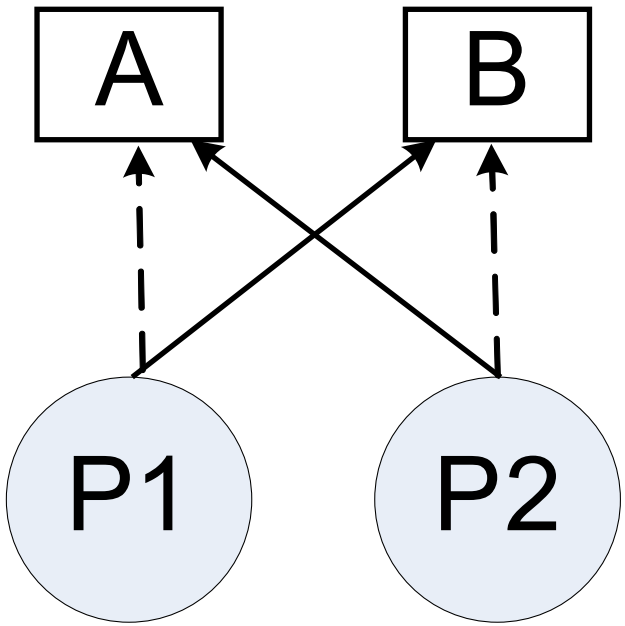
Program is executed on shared memory system.
sum is shared variable

```
// thread 0
int k;
for(k=0, i=0; i< 100; i++){
    k = (k + arr[i])%0xFF;
}
sum = (sum + k)%0xFF;
```

```
// thread 1
int k;
for(k=0, i=100; i< 200; i++){
    k = (k + arr[i])%0xFF;
}
sum = (sum + k)%0xFF;
```

Synchronization required

Errors. Deadlock



Process P1 locks resource B, at the same time process P2 locks resource A. If P1 will lock resource A and P2 will try to lock resource B, they will blocked forever.

Deadlock could turn into livelock, which is more unpleasant than its counterpart.

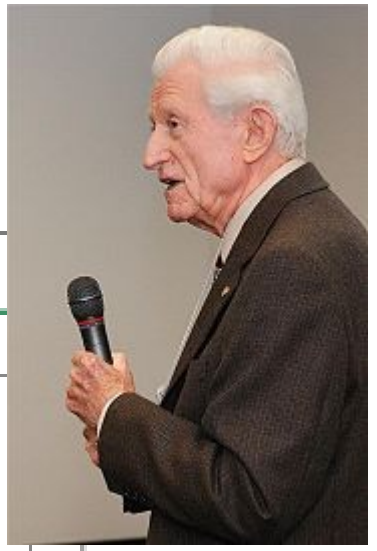
Scalability

As the number of computational units increase, program should run faster. We add more computational power therefore we can expect performance to growth.

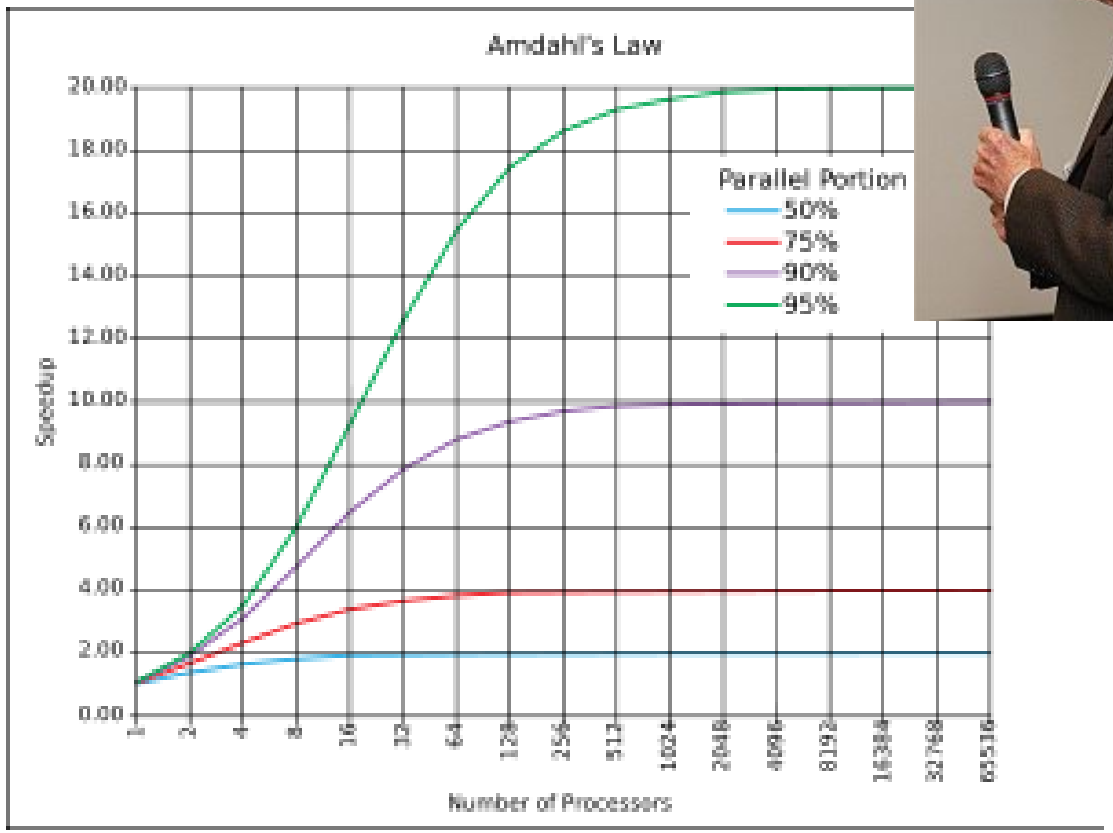
Ability of the program to follow this rule is program's scalability.

Level of scalability is the number of CPUs (computational nodes) at which addition of extra CPUs gain no reasonable performance growth

Gene Amdahl's law



$$\frac{1}{(1-P)+\frac{P}{N}}$$



N – number of CPUs

P – part of the program, that could be paralleled

Advantages/disadvantages of shared and distributed memory systems

	+	-
Shared memory systems	Easy to program	High cost Low scalability
Distributed memory systems	High scalability Low cost	Difficult to program

Development environment

- Compiler directives
- HPF
- OpenMP
- MPI
- POSIX Threads
- etc.

Approaches to PP development

Questions to be answered:

1. Worth this program to be parallelized?
2. Why do I want to parallel this program? Because of time/memory limitation.
3. Which part of the program could be parallelized?
4. Which type of computational environment is suitable for this task?
5. What type of computer do we have (can use) now/in future?
6. How many working hours do I wont to spend parallelizing the code?

Higher degree of parallelism and optimization is higher cost of the final program.

Self-test questions

- What is parallel program?
- Name several computer systems and order it according to Flinn's taxonomy
- What is the difference between data and task parallelism?
- Is it possible to develop parallel program for calculating Fibonacci's numbers?