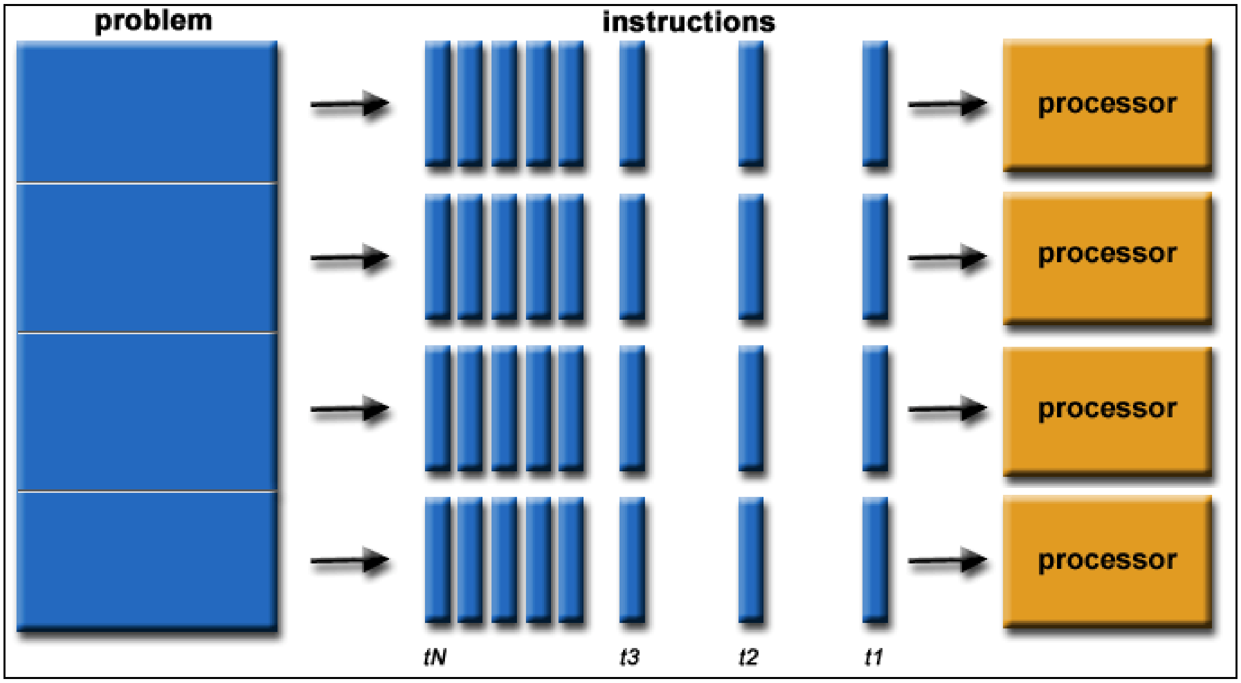
Definitions from: https://www.omnisci.com/technical-glossary/parallel-computing

Parallel Computing Definition

Parallel computing is a type of computing architecture in which several processors simultaneously execute multiple, smaller calculations broken down from an overall larger, complex problem.



What is Parallel Computing?

Parallel computing refers to the process of breaking down larger problems into smaller, independent, often similar parts that can be executed simultaneously by multiple processors communicating via shared memory, the results of which are combined upon completion as part of an overall algorithm. The primary goal of parallel computing is to increase available computation power for faster application processing and problem solving.

Parallel computing infrastructure is typically housed within a single datacenter where several processors are installed in a server rack; computation requests are distributed in small chunks by the application server that are then executed simultaneously on each server.

Definitions from: https://www.omnisci.com/technical-glossary/parallel-computing

There are generally four types of parallel computing, available from both proprietary and open source parallel computing vendors -- bit-level parallelism, instruction-level parallelism, task parallelism, or superword-level parallelism:

- Bit-level parallelism: increases processor word size, which reduces the quantity of instructions the processor must execute in order to perform an operation on variables greater than the length of the word.
- Instruction-level parallelism: the hardware approach works upon dynamic parallelism, in which the processor decides at run-time which instructions to execute in parallel; the software approach works upon static parallelism, in which the compiler decides which instructions to execute in parallel
- Task parallelism: a form of parallelization of computer code across multiple processors that runs several different tasks at the same time on the same data
- Superword-level parallelism: a vectorization technique that can exploit parallelism of inline code

Parallel applications are typically classified as either fine-grained parallelism, in which subtasks will communicate several times per second; coarse-grained parallelism, in which subtasks do not communicate several times per second; or embarrassing parallelism, in which subtasks rarely or never communicate. Mapping in parallel computing is used to solve embarrassingly parallel problems by applying a simple operation to all elements of a sequence without requiring communication between the subtasks.

The popularization and evolution of parallel computing in the 21st century came in response to processor frequency scaling hitting the power wall. Increases in frequency increase the amount of power used in a processor, and scaling the processor frequency is no longer feasible after a certain point; therefore, programmers and manufacturers began designing parallel system software and producing power efficient processors with multiple cores in order to address the issue of power consumption and overheating central processing units.

The importance of parallel computing continues to grow with the increasing usage of multicore processors and GPUs. GPUs work together with CPUs to increase the throughput of data and the number of concurrent calculations within an application. Using the power of parallelism, a GPU can complete more work than a CPU in a given amount of time.

Fundamentals of Parallel Computer Architecture

Parallel computer architecture exists in a wide variety of parallel computers, classified according to the level at which the hardware supports parallelism. Parallel computer architecture and programming techniques work together to effectively utilize these machines. The classes of parallel computer architectures include:

- Multi-core computing: A multi-core processor is a computer processor integrated circuit with two or more separate processing cores, each of which executes program instructions in parallel. Cores are integrated onto multiple dies in a single chip package or onto a single integrated circuit die, and may implement architectures such as multithreading, superscalar, vector, or VLIW. Multi-core architectures are categorized as either homogeneous, which includes only identical cores, or heterogeneous, which includes cores that are not identical.
- Symmetric multiprocessing: multiprocessor computer hardware and software architecture in which two or more independent, homogeneous processors are controlled by a single operating system instance that treats all processors equally, and is connected to a single, shared main memory with full access to all common resources and devices. Each processor has a private cache memory, may be connected using on-chip mesh networks, and can work on any task no matter where the data for that task is located in memory.
- Distributed computing: Distributed system components are located on different networked computers that coordinate their actions by communicating via pure HTTP, RPC-like connectors, and message queues. Significant characteristics of distributed systems include independent failure of components and concurrency of components. Distributed programming is typically categorized as client–server, three-tier, n-tier, or peer-to-peer architectures. There is much overlap in distributed and parallel computing and the terms are sometimes used interchangeably.
- Massively parallel computing: refers to the use of numerous computers or computer processors to simultaneously execute a set of computations in parallel. One approach involves the grouping of several processors in a tightly structured, centralized computer cluster. Another approach is grid computing, in which many widely distributed computers work together and communicate via the Internet to solve a particular problem.

Other parallel computer architectures include specialized parallel computers, cluster computing, grid computing, vector processors, application-specific integrated circuits, general-purpose computing on graphics processing units (GPGPU), and reconfigurable computing with field-programmable gate arrays. Main memory in any parallel computer structure is either distributed memory or shared memory.

Parallel Computing Software Solutions and Techniques

Definitions from: https://www.omnisci.com/technical-glossary/parallel-computing

Concurrent programming languages, APIs, libraries, and parallel programming models have been developed to facilitate parallel computing on parallel hardware. Some parallel computing software solutions and techniques include:

- Application checkpointing: a technique that provides fault tolerance for computing systems by recording all of the application's current variable states, enabling the application to restore and restart from that point in the instance of failure. Checkpointing is a crucial technique for highly parallel computing systems in which high performance computing is run across a large number of processors.
- Automatic parallelization: refers to the conversion of sequential code into multi-threaded code in order to use multiple processors simultaneously in a shared-memory multiprocessor (SMP) machine. Automatic parallelization techniques include Parse, Analyze, Schedule, and Code Generation. Typical examples of common parallelizing compilers and tools are Paradigm compiler, Polaris compiler, Rice Fortran D compiler, SUIF compiler, and Vienna Fortran compiler.
- Parallel programming languages: Parallel programming languages are typically classified as either distributed memory or shared memory. While distributed memory programming languages use message passing to communicate, shared memory programming languages communicate by manipulating shared memory variables.

Difference Between Parallel Computing and Cloud Computing

Cloud computing is a general term that refers to the delivery of scalable services, such as databases, data storage, networking, servers, and software, over the Internet on an as-needed, pay-as-you-go basis.

Cloud computing services can be public or private, are fully managed by the provider, and facilitate remote access to data, work, and applications from any device in any place capable of establishing an Internet connection. The three most common service categories are Infrastructure as as Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).

Cloud computing is a relatively new paradigm in software development that facilitates broader access to parallel computing via vast, virtual computer clusters, allowing the average user and smaller organizations to leverage parallel processing power and storage options typically reserved for large enterprises.

Definitions from: https://www.omnisci.com/technical-glossary/parallel-computing

Difference Between Parallel Processing and Parallel Computing

Parallel processing is a method in computing in which separate parts of an overall complex task are broken up and run simultaneously on multiple CPUs, thereby reducing the amount of time for processing.

Dividing and assigning each task to a different processor is typically executed by computer scientists with the aid of parallel processing software tools, which will also work to reassemble and read the data once each processor has solved its particular equation. This process is accomplished either via a computer network or via a computer with two or more processors.

Parallel processing and parallel computing occur in tandem, therefore the terms are often used interchangeably; however, where parallel processing concerns the number of cores and CPUs running in parallel in the computer, parallel computing concerns the manner in which software behaves to optimize for that condition.

Difference Between Sequential and Parallel Computing

Sequential computing, also known as serial computation, refers to the use of a single processor to execute a program that is broken down into a sequence of discrete instructions, each executed one after the other with no overlap at any given time. Software has traditionally been programmed sequentially, which provides a simpler approach, but is significantly limited by the speed of the processor and its ability to execute each series of instructions. Where uni-processor machines use sequential data structures, data structures for parallel computing environments are concurrent.

Measuring performance in sequential programming is far less complex and important than benchmarks in parallel computing as it typically only involves identifying bottlenecks in the system. Benchmarks in parallel computing can be achieved with benchmarking and performance regression testing frameworks, which employ a variety of measurement methodologies, such as statistical treatment and multiple repetitions. The ability to avoid this bottleneck by moving data through the memory hierarchy is especially evident in parallel computing for data science, machine learning parallel computing, and parallel computing artificial intelligence use cases.

Definitions from: https://www.omnisci.com/technical-glossary/parallel-computing

Sequential computing is effectively the opposite of parallel computing. While parallel computing may be more complex and come at a greater cost up front, the advantage of being able to solve a problem faster often outweighs the cost of acquiring parallel computing hardware.