Q: PARALLEL AND DISTRIBUTED COMPUTING types with brief descriptions and example?

Parallel computing and distributed computing are two approaches to performing high-performance computing tasks that involve breaking down a large problem into smaller sub-tasks that can be executed simultaneously by multiple processors or computers.

Parallel computing is a technique that utilizes multiple processors to work on a single problem concurrently, thereby reducing the total execution time. In parallel computing, a single program is executed by multiple processors simultaneously, with each processor working on a different portion of the task. Parallel computing can be implemented in two ways: shared memory and distributed memory. In shared memory systems, multiple processors share access to a common memory pool, whereas in distributed memory systems, each processor has its own memory, and communication between processors occurs over a network.

Distributed computing, on the other hand, is a technique where a single problem is broken down into smaller sub-tasks that can be executed on multiple computers connected by a network. In distributed computing, each computer works on its own sub-task, and the results are combined to produce the final output. Distributed computing is best suited for tasks that are too large for a single computer to handle or require significant amounts of data transfer.

Both parallel computing and distributed computing are widely used in many different fields, including scientific computing, data analysis, and artificial intelligence. These techniques can significantly increase performance and enable the processing of large datasets that would be impossible to process using traditional computing techniques.

Types of Parallel Computing:

Bit-level parallelism: It is the form of parallel computing which is based on the increasing processor's size. It reduces the number of instructions that the system must execute in order to perform a task on large-sized data. **Example:** Consider a scenario where an 8-bit processor must compute the sum of two 16-bit integers. It must first sum up the 8 lower-order bits, then add the 8 higher-order bits, thus requiring two instructions to perform the operation. A 16-bit processor can perform the operation with just one instruction.

Instruction-level parallelism: A processor can only address less than one instruction for each clock cycle phase. These instructions can be re-ordered and grouped which are later on executed concurrently without affecting the result of the program. This is called instruction-level parallelism.

Task Parallelism: Task parallelism employs the decomposition of a task into subtasks and then allocating each of the subtasks for execution. The processors perform execution of sub tasks concurrently.

Data-level Parallelism: Instructions from a single stream operate concurrently on several data – Limited by non-regular data manipulation patterns and by memory bandwidth.

Applications of Parallel Computing:

- **Databases and Data mining.**
- **Real-time simulation of systems.**
- **Science and Engineering.**
- **Advanced graphics, augmented reality, and virtual reality.**

Parallel Computing Example:

This computing method is ideal for anything involving complex simulations or modeling. Common applications for it include seismic surveying, computational astrophysics, climate modeling, financial risk management, agricultural estimates, video color correction, medical imaging, drug discovery, and computational fluid dynamics.

Types of Distributed Computing:

Client/Server Systems: In client server systems, the client requests a resource and the server provides that resource. A server may serve multiple clients at the same time while a client is in contact with only one server. Both the client and server usually communicate via a computer network and so they are a part of distributed systems.

Peer to Peer Systems: The peer to peer systems contains nodes that are equal participants in data sharing. All the tasks are equally divided between all the nodes. The nodes interact with each other as required as share resources. This is done with the help of a network.

Three-Tier Model: The three-tier model introduces an additional tier between client and server — the agent tier. This middle tier holds the client data, releasing the client from the burden of managing its own information. The client can access its data through a web application, typically. Through this, the client application's and the user's work is reduced and automated easily. For example, a cloud storage space with the ability to store your files and a document editor. Such a storage solution can make your file available anywhere for you through the internet, saving you from managing data on your local machine.

Multi-Tier Model: Enterprises need business logic to interact with various backend data tiers and frontend presentation tiers. This logic sends requests to multiple enterprise network services easily. That's why large organizations prefer the n-tier or multi-tier distributed computing model. For example, an enterprise network with n-tiers that collaborate when a user publishes a social media post to multiple platforms.

Distributed Computing Example:

Distributed computing is best for building and deploying powerful applications running across many different users and geographies. Anyone performing a Google search is already using distributed computing. Distributed system architectures have shaped much of what we would call "modern business," including cloud-based computing, edge computing, and software as a service (SaaS).