load balancing

1. Definition

Load Balancing is the process of **distributing tasks or computational work evenly across all processors or nodes** in a parallel or distributed system.

- Goal: **maximize resource utilization**, **minimize execution time (makespan)**, and **avoid idle processors**.
- Critical for **parallel efficiency and scalability**.

---

2. Key Concepts

Concept	Description
Static Load Balancing	Task assignment done **before execution**; no runtime adjustments
Dynamic Load Balancing	Task assignment done **during execution** based on processor availability
Makespan	Total time to complete all tasks; used to measure balance
Overload / Underload	Some processors have more or fewer tasks than others
Task Granularity	Size of a task; smaller tasks allow finer balancing but increase overhead

---

3. Types of Load Balancing

A. Static Load Balancing

- **Definition:** Tasks are **pre-assigned** to processors.
- **Example:** Divide 1000 matrix elements among 4 processors equally (250 elements each).
- **Pros:** Simple, low runtime overhead
- **Cons:** Cannot adapt to irregular workloads or runtime delays

B. Dynamic Load Balancing

- **Definition:** Tasks are assigned **at runtime** based on processor availability.
- **Example:** Work-stealing: idle processor takes tasks from busy processors.
- **Pros:** Adapts to irregular workloads, heterogeneous processors
- **Cons:** Overhead for monitoring and task migration

C. Centralized vs Distributed Scheduling

- **Centralized:** One master assigns tasks to all workers
- **Distributed:** Processors coordinate to balance load among themselves

4. Illustrative Example

Scenario: 4 processors computing 16 tasks

Processor	Static Assignment	Dynamic Assignment
P0	4 tasks	3 tasks initially, steals 1 from P2
P1	4 tasks	4 tasks
P2	4 tasks	2 tasks, gives 1 to P0, 1 to P3
P3	4 tasks	4 tasks total after stealing

- **Observation:** Dynamic load balancing reduces idle time and improves **makespan**.

5. Behavior in Parallel and Distributed Systems

Feature	Parallel Systems (Shared Memory)	Distributed Systems (Cluster/Nodes)
Task Assignment	Threads scheduled by OS/runtime	Master node or distributed scheduler assigns tasks
Overhead	Minimal, shared memory access	Higher due to network communication for task migration
Scalability	Limited by core count	Can scale across nodes, but network delays matter
Synchronization	Needed if tasks share memory	Needed to coordinate task distribution and results
Use Cases	Multithreaded simulations, HPC	Distributed simulations, cloud computing, big data jobs

6. Key Points

- **Load balancing improves parallel efficiency** by reducing idle processor time.
- **Static:** simple but inflexible; best for predictable workloads
- **Dynamic:** adaptable to irregular workloads; adds runtime overhead
- **Critical in distributed systems** where communication and heterogeneity impact performance