

Below is a **complete, in-depth explanation of Condor (HTCondor) for Parallel & Distributed Computing**, written clearly for exams, concepts, and real understanding, with architecture and examples.

What is Condor (HTCondor)?

1. Definition of Condor

Condor, now called HTCondor, is a **distributed workload management system** used to schedule, manage, and execute large numbers of jobs on a collection of computers.

One-line definition (exam-ready):

Condor is a distributed job scheduling system that harnesses idle computing resources to execute parallel and distributed jobs efficiently.

2. Why Condor is Needed

In universities, labs, and organizations:

- Many computers stay **idle**
- High-performance tasks need **massive computation**
- Dedicated supercomputers are expensive

Condor solves this by:

- Using **idle machines**
 - Running jobs in the background
 - Migrating jobs if machines become busy
-

3. What Type of System is Condor?

- **Distributed Computing System**
- **High-Throughput Computing (HTC)** (not HPC)
- Focuses on:
 - Long-running jobs
 - Many independent tasks

- ⚠ Condor is **not** a programming language
 - ⚠ Condor is **not** a compiler
 - ✓ It is a **resource manager / scheduler**
-

4. High-Throughput vs High-Performance Computing

Feature	HTC (Condor)	HPC
Goal	Max jobs over time	Fast single job
Jobs	Independent	Tightly coupled
Example	Parameter sweeps	Weather simulation

5. Condor Architecture

Main Components

1 Submit Machine

- Where user submits jobs
- Contains job queue

2 Execute Machines

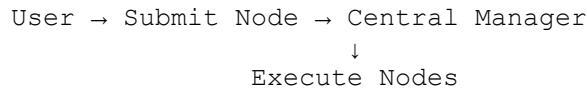
- Run the jobs
- Often idle desktop PCs

3 Central Manager

Controls the entire pool:

- **Collector** → collects resource info
 - **Negotiator** → matches jobs to machines
-

Architecture Diagram (Conceptual)



6. Key Condor Daemons

Daemon	Role
condor_schedd	Manages job queue
condor_startd	Manages execute machine
condor_collector	Collects resource info
condor_negotiator	Matches jobs
condor_master	Controls other daemons

7. How Condor Works (Step-by-Step)

1. User submits job
2. Job waits in queue
3. Negotiator finds matching machine
4. Job is transferred
5. Job executes
6. Results returned

8. Job Submission Example

Condor Submit File

```
Executable = myprogram.sh
Output = output.txt
Error = error.txt
Log = log.txt
Queue
```

Command

```
condor_submit job.sub
```

Condor:

- Finds free machine

- Runs job
 - Saves output
-

9. Parallel Computing with Condor

Condor supports **parallel jobs** using **multiple independent tasks**.

Example: Parameter Sweep

Queue 100

- Same program
 - Different parameters
 - 100 jobs run in parallel on different machines
-

10. Distributed Computing Example

Scientific Simulation

- 1000 simulations
- Each simulation independent
- Runs on 100+ machines

Condor:

- Distributes jobs
 - Handles failures
 - Maximizes throughput
-

11. Job Migration (Key Feature)

If:

- User starts using machine
- Laptop is unplugged

Condor:

- **Suspends or migrates job**

- Resumes elsewhere
-

12. Fault Tolerance

- Automatic retry
 - Checkpointing
 - No data loss
-

13. Advantages of Condor

- Uses idle resources
 - Scalable
 - Fault tolerant
 - Supports heterogeneous systems
 - Open source
-

14. Limitations of Condor

- Not good for tightly coupled parallel jobs
 - Setup complexity
 - Network overhead
-

15. Condor vs Globus vs MPI

Feature	Condor	Globus	MPI
Type	Scheduler	Middleware	Library
Focus	HTC	Grid computing	HPC
Parallelism	Task-level	Resource-level	Process-level
Use Case	Many jobs	Resource sharing	One big job

16. Real-World Use Cases

- Universities
 - Research labs
 - Bioinformatics
 - Monte Carlo simulations
 - Data analysis
-

17. Exam-Oriented Summary

Keywords:

- HTCondor
- High-Throughput Computing
- Scheduler
- Idle resources
- Job migration

One-Paragraph Answer:

Condor (HTCondor) is a distributed workload management system designed for high-throughput computing. It schedules and executes large numbers of independent jobs by utilizing idle computing resources across a network. Condor provides fault tolerance, job migration, and efficient resource utilization, making it ideal for distributed computing environments.

If you want next, I can:

- Solve **numerical problems**
- Compare **Condor vs Hadoop**
- Explain **Condor scheduling algorithm**
- Prepare **short notes & MCQs**

Just tell me.