**Activity 1: Case Study Analysis (Cluster Applications)**

* **Task:** Divide students into groups. Each group selects one *Grand Challenge Application* (e.g., climate modeling, protein dynamics, quantum chemistry).
* **Activity:** Identify the **parallel programming model/paradigm** best suited for that application and justify the choice.
* **Output:** A short group presentation (3–5 minutes) explaining mapping of problem → model/paradigm → expected challenges.

**Activity 2: Parallel Paradigm Matching Game**

* **Task:** Provide students with real-world problems (e.g., image rendering, web search indexing, genetic algorithms, transaction rollback).
* **Activity:** Students must **match each problem** with the correct **parallel programming paradigm** (Task Farming, SPMD, Data Pipelining, Divide & Conquer, Speculative Parallelism).
* **Output:** Written justification for at least 2 matches (why this paradigm is most efficient for that problem).

**Activity 3: Code Granularity Sorting Challenge**

* **Task:** Provide 6–8 code snippets (pseudo-code only, e.g., loop-level, function call, pipeline stages).
* **Activity:** Students classify each snippet into **Fine Grain, Medium Grain, or Large Grain** parallelism.
* **Output:** Students must explain how the **grain size affects efficiency** and what type of model (shared/distributed) would be ideal.

**Activity 4: Parallel Programming Debate**

* **Task:** Divide class into two teams:
  + Team A → **Implicit Parallelism Advocates**
  + Team B → **Explicit Parallelism Advocates**
* **Activity:** Each team presents **3 arguments in favor** and **2 counterarguments** against the other approach.
* **Output:** A moderated debate where students critically evaluate efficiency, ease of programming, portability, and scalability.

👉 These activities balance **concept understanding + application + critical thinking** and fit your moderate-level difficulty requirement.

Would you like me to also **prepare ready-to-use handouts/worksheets** for these activities (with instructions + examples) so you can directly distribute them in class?

Sources

**You said:**

add difficulty

**ChatGPT said:**

Got it 👍 — let’s **add difficulty (8/10 level)** to make these activities more challenging for final-year B.Tech students. I’ll extend each activity so that they require **more analysis, synthesis, and application** instead of just recall.

**Activity 1: Advanced Case Study Analysis (Cluster Applications)**

* **Added Difficulty:**  
  Instead of just mapping an application to a paradigm, ask students to:
  1. **Identify possible bottlenecks** (e.g., synchronization, load balancing, communication overhead).
  2. **Propose optimization strategies** (e.g., hybrid model, middleware selection, or granularity adjustment).
* **Deliverable:** 5–6 minute presentation + a one-page technical note.

**Activity 2: Complex Paradigm Matching Game**

* **Added Difficulty:**  
  After matching problems to paradigms, students must:
  1. **Design a pseudo-architecture diagram** (showing processors, communication, and data flow).
  2. **Estimate scalability** (how performance changes if number of processors doubles).
* **Deliverable:** A short report with mappings, diagrams, and scalability reasoning.

**Activity 3: Code Granularity Challenge with Trade-offs**

* **Added Difficulty:**  
  Along with sorting snippets into fine/medium/large grain, students must:
  1. **Rewrite one sequential snippet into parallel pseudo-code**.
  2. **Compare trade-offs**: execution speedup vs communication overhead.
* **Deliverable:** Annotated pseudo-code + short write-up on trade-offs.

**Activity 4: Parallel Programming Debate (Extended)**

* **Added Difficulty:**  
  Add a **third dimension**: each team must also argue from the perspective of **hardware constraints** (shared memory multiprocessor vs distributed cluster).
  + Team A defends *Implicit Parallelism* on both architectures.
  + Team B defends *Explicit Parallelism* on both architectures.
* **Deliverable:** Debate + a **joint comparative table** summarizing advantages/disadvantages across architectures.

✨ With these modifications, the activities will now test **higher-order thinking (analysis, evaluation, design)** — suitable for final-year B.Tech.

Do you want me to **design proper worksheets (with problem statements, space for answers, and evaluation rubrics)** for each activity so you can directly hand them out in class?