Solving with Condor

Working on the same data in Condor is different from the start. With Condor, you have to manually partition large data while this is handled using Hadoop on your behalf. Condor also lacks a sorting algorithm and you would have to implement this yourself. The sorting algorithm is native to Hadoop. The comparison below is specific to the work task we've tackled in the assignment.

Condor vs. Hadoop

- **Condor Pros**:
    - Flexible job types (not just MapReduce).
    - Lightweight, no heavy framework overhead.
    - Good for parallel tasks (e.g., initial parsing).
- **Condor Cons**:
    - Manual data partitioning and result merging.
    - No built-in shuffle/sort (custom code needed).
    - Scales poorly for iterative tasks or large intermediate data.
- **Hadoop Pros**:
    - Built-in data partitioning (HDFS splits petabyte data).
    - Automatic shuffle/sort for grouping (e.g., by genre).
    - Fault-tolerant with data replication.
    - Optimized for large-scale batch processing.
- **Hadoop Cons**:
    - Heavier framework (HDFS, YARN overhead).
    - Rigid MapReduce paradigm.
    - Disk I/O intensive (slower for some workloads).

Which Wins For Our Use Case?

- **Hadoop Wins**:
    - Petabyte-scale data fits HDFS naturally.
    - Your task (grouping by genre, aggregating ratings) aligns with MapReduce strengths.
    - Condor requires custom orchestration; Hadoop automates it.
    - Fault tolerance and scalability favor Hadoop for this size.