Dispatcher Worker - Linux Pthreads Dispatcher/Worker Model.

Overview

The **Dispatcher Worker** project implements a dispatcher/worker model in Linux using pthreads. The dispatcher reads commands from an input file and executes some commands serially, while others are queued as jobs for worker threads. The workers execute these jobs in parallel, leveraging multi-threading for efficiency.
This project is written in C and utilizes Linux threading and synchronization mechanisms to efficiently manage task distribution and completion. It includes features like worker thread creation, command queuing, file-based counters, logging, and job execution.

Key Features:

1. **Command Input**: The dispatcher reads commands from an input file and assigns them for serial or parallel execution.
    - Dispatcher commands like dispatcher msleep and dispatcher wait are executed serially.
    - Worker commands are offloaded to worker threads for parallel execution.

2. **Thread Management**:
    - Creates up to 4096 worker threads at the start, which persist throughout the program's execution.
    - Worker threads pick up jobs from a shared work queue and process them in parallel.

3. **Counter Files**:
    - Up to 100 counter files (countxx.txt) are created, initialized to 0.
    - Workers can increment or decrement these counters, and changes are made directly to the disk files in real-time.

4. **Job Execution**:
    - Worker threads execute job commands serially within the thread.
    - Commands include msleep, increment, decrement, and repeat.
    - Jobs are queued by the dispatcher and executed in parallel by the workers.

5. **Logging**:
    - Optional logging tracks all actions performed by the dispatcher and worker threads.
    - Workers create individual log files (threadxx.txt) to record job start and completion times.
    - The dispatcher logs its actions to dispatcher.txt.

6. **Performance Statistics**:
    - After processing the input file, the dispatcher generates a stats.txt file with job turnaround statistics, including total runtime, minimum, maximum, and average job times.

Command Syntax:

./dispatcher <cmdfile.txt> <num_threads> <num_counters> <log_enabled>

cmdfile.txt: Input file containing dispatcher and worker commands.
num_threads: Number of worker threads to create (max 4096).
num_counters: Number of counter files to create (max 100).
log_enabled: Set to 1 to enable logging, or 0 to disable logging.

Command File Syntax

1. **Dispatcher Commands** (executed serially):
   - dispatcher msleep x: Puts the dispatcher to sleep for x milliseconds.
   - dispatcher wait: Waits for all worker jobs to complete before proceeding.

2. **Worker Commands** (executed in parallel by worker threads):
   - msleep x: Puts the worker thread to sleep for x milliseconds.
   - increment x: Increments counter x.
   - decrement x: Decrements counter x.
   - repeat x; command sequence: Repeats the command sequence x times.

Log Files

- Each worker thread logs its activity in a threadxx.txt file, where xx is the worker thread ID.
- Logs include job start and end times in milliseconds since the program started.

Statistics

At the end of the program, a stats.txt file is generated, containing:

- Total running time (in milliseconds).
- Sum, minimum, maximum, and average job turnaround times (in milliseconds).