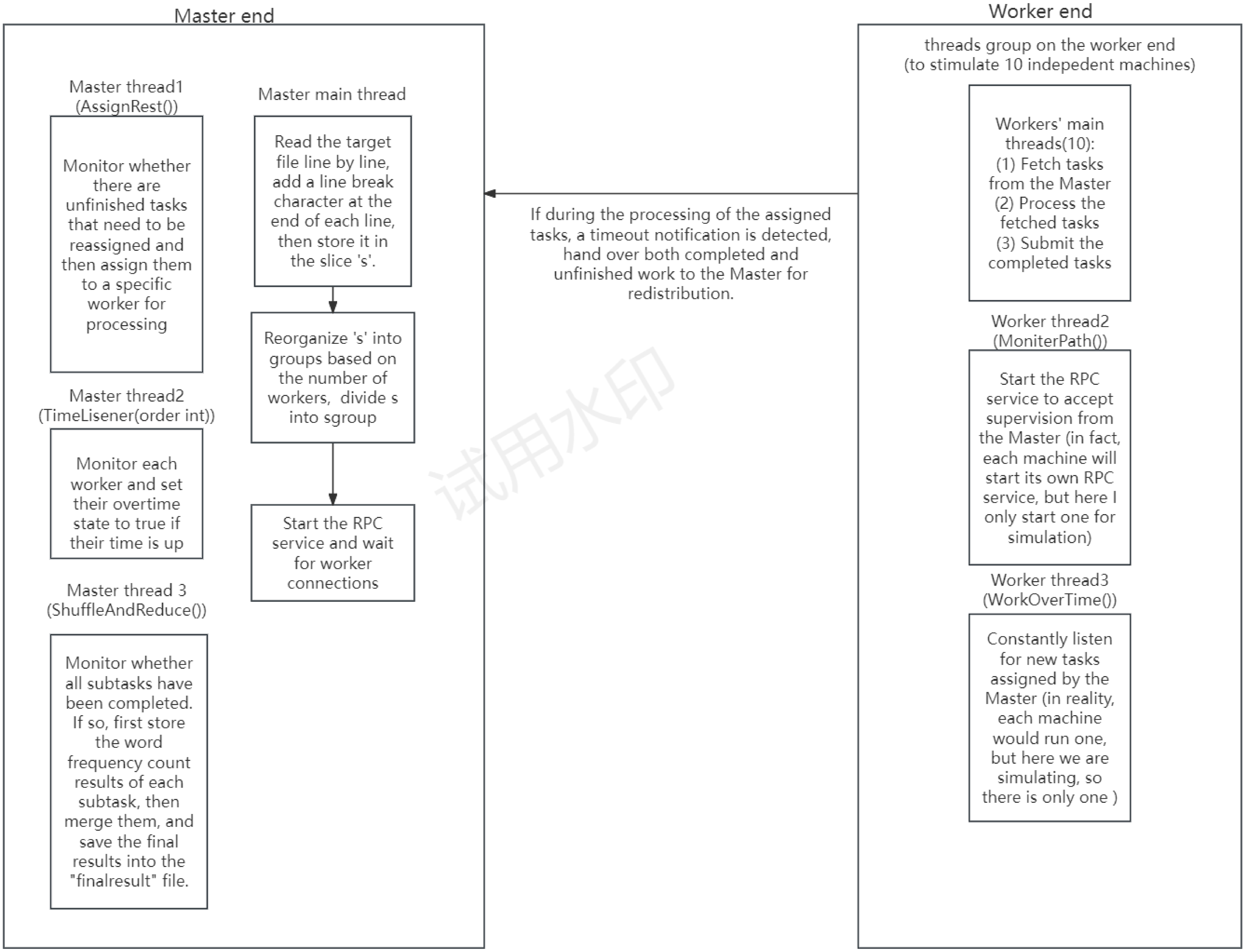
Master end
Master thread1 (AssignRest())
Monitor whether there are unfinished tasks that need to be reassigned and then assign them to a specific worker for processing
Master main thread
Read the target file line by line, add a line break character at the end of each line, then store it in the slice 's'.
Reorganize 's' into groups based on the number of workers, divide s into sgroup
Start the RPC service and wait for worker connections
Master thread2 (TimeLisener(order int))
Monitor each worker and set their overtime state to true if their time is up
Master thread 3 (ShuffleAndReduce())
Monitor whether all subtasks have been completed. If so, first store the word frequency count results of each subtask, then merge them, and save the final results into the "finalresult" file.
If during the processing of the assigned tasks, a timeout notification is detected, hand over both completed and unfinished work to the Master for redistribution.
Worker end
threads group on the worker end (to stimulate 10 indepedent machines)
Workers' main threads(10): (1) Fetch tasks from the Master (2) Process the fetched tasks (3) Submit the completed tasks
Worker thread2 (MoniterPath())
Start the RPC service to accept supervision from the Master (in fact, each machine will start its own RPC service, but here I only start one for simulation)
Worker thread3 (WorkOverTime())
Constantly listen for new tasks assigned by the Master (in reality, each machine would run one, but here we are simulating, so there is only one)