Construction Site Management Project

Operating Systems

Course Instructor

Bilal Hassan

Course Instructor

Obaid Ullah

Submitted By

Talal habib 21I-1111

Umar Farooq 21I-1143

SE-P

Department of Software Engineering

FAST – National University of Computer & Emerging Sciences

Report: Construction Site Task Management System Stimulator

Introduction

Our program implements a simulation of a construction site task management system. The system assigns tasks to workers based on their skill sets, prioritizes tasks, and manages available resources such as bricks, cement, and tools. The simulation runs for a specified number of days, and each day involves task assignment, resource replenishment, and worker shift changes among other things that will be discussed further in this document.

Assumptions

We have assumed that

- Tasks should be adjusted based on the weather conditions at the start of the day.
- Tasks assignments are based on the skillset of the workers, if a worker does not have that skillset, then they cannot perform that task.
- One worker can have multiple skillsets
- Threads would only be created of the Tasks and not the workers.
- Even if high priority tasks are available but if no worker to perform them then the workers that are free can perform other priority tasks provided that resource is available
- Resources are not fixed and can grow past the initial starting amount.

Program Components

1. Enums and Data Structures

Enums: The program defines enums for weather conditions, skill sets, and work types.

Structs: Several structs are defined for tasks, workers, memory management, day logs, resource utilization and state stack.

2. Initialization and Configuration

Skills and Task Types: The program initializes vectors for skills and predefined task types with associated properties such as priority, skill set required, and necessary resources.

3. Task and Worker Management

Task Struct: Describes a construction task with attributes such as name, priority, skill set required, and assigned worker.

Worker Struct: Represents a worker with an ID, skill levels, on-break status, fatigue counter, and shift information.

4. Memory Management

Memory Struct: Manages queues for high, medium, and low-priority tasks, in-progress tasks, available and occupied workers, and resource information. It also keeps track of the current day's weather.

5. Day Logging

Day_Log Struct: Records completed tasks, day number, weather, and workers on shift for each day.

Log Struct: Manages logs for multiple days and provides a function to print the log to a file.

6. Resource and Resource Utilization

Resources Struct: Represents the availability of bricks, cement, and tools.

Resource_Utilization Struct: Tracks resource usage and history.

7. State Stack

StateStack Struct: Maintains the state of interrupted tasks using a stack.

8. Thread Synchronization

Mutexes: Three mutexes (dataMutex, queueMutex, resourceMutex) are used for thread synchronization.

9. Main Simulation Loop

Dynamic Task Adjustment:

The DynamicTaskAdjustment function incorporates different task assignment strategies based on daily weather conditions.

Sunny Day: Tasks are assigned based on priority.

Rainy Day: Tasks are assigned based on priority, but outdoor tasks are put on hold.

Disaster Day: Tasks are put on Hold except those tasks which are performed offsite.

These adjustments enhance the realism of the simulation by considering weather conditions in task scheduling.

Worker Management:

Check Fatigue:

The CheckFatigue function checks if a worker has been fatigued more than 3 times. If so, that person is put on break, and they are replaced with a person with a similar skill set from the NotOnShift vector.

Simulate Break:

The StimulateBreak function randomly decides that a worker is now on break and replaces the worker with a person with a similar skill set from the NotOnShift vector.

Dynamic Break Assignment:

On a disaster day, all workers are put on break except for the engineers. This is to ensure that essential personnel are available in case of emergencies.

The main function runs a simulation loop for a specified number of days. Each day involves:

- Weather check
- Resource replenishment
- Initialization of tasks
- Shift changes for workers
- Dynamic task adjustment
- Task assignment to workers

- Task execution in separate threads
- Logging completed tasks, workers on shift, and weather conditions

Simulation Flow

1. **Initialization**: The program initializes workers and resources.

2. **Daily Simulation Loop:**

   Weather Check: Determines the weather for the day.

3. **Resource Replenishment:** Adds resources based on the previous day's utilization.
4. **Task Initialization:** Creates a random number of tasks for the day.
5. **Shift Changes:** Adjusts worker availability based on shift changes.
6. **Dynamic Task Adjustment:** Adjusts tasks based on weather conditions.
7. **Task Assignment:** Assigns tasks to available workers based on priority and skill.
8. **Task Execution:** Threads are spawned to execute assigned tasks concurrently.
9. **Check Fatigue and Stimulate Break:** Additional features to manage worker fatigue and breaks.
10. **Logging:** Records completed tasks and workers on shift for the day.

Task Execution (Thread Function)

- The program uses multi-threading to execute tasks concurrently.
- Each task is assigned a thread (TaskThread2).
- Threads check resource availability, execute tasks, and update resource utilization.
- Resources Like Tools are utilized and released as soon as Task is completed.
- Threads communicate resource availability or errors to the main program through pipes.

Conclusion

The construction site task management system efficiently allocates tasks to workers based on priority, skill sets, and resource availability. The simulation provides insights into daily operations, task completion, and resource utilization on a construction site.

Work Division:

Report:

- 80% Done by Umar Farooq
- 20% Done by Talal Habib

Code:

- 40% Done by Umar Farooq (All the Structs and Complete Log Code)
- 60% Done by Talal Habib (Complete TaskThread2 Code)
- In the code itself All functions codes are a mix match of Umar and Talal codes

What we Learn:

Aside from learning how to collaborate on a project, we learned how we could manage shared resources using synchronization and we learned how to effectively manage shared resources which

are being used by tasks that we are creating. We also learned priority scheduling to stimulate how like real life we can have unpredictability. Also learned how not to use Forks and pipes ;(

Things to Improve:

Due to a time crunch we were not able to complete what we had initially planned. We could have improved the dynamic task adjustment by adding a feature that lowers a task priority based on how much it has been assigned previously and a priority can be raised higher if a task is sitting free for quite a while.

Currently we are swapping the on break worker with any worker that is not on shift but we could implement an LRU algorithm to insure we would have assigned a worker that had been sitting free for quite some time

We could also stimulate Task termination by a worker midway through performing that Task for any reason, but we were not able to do that as well

One thing we could not figure out was to implement interrupt behavior for our threads and stimulate the weather change such that weather could change midday and all tasks being performed would have been adjusted accordingly.