Sebastian Torrejon Alonzo
Fabian Kirberg

Bakeoff Documentation

Project Overview

The project aims to simulate a shared kitchen environment where multiple bakers, each running as a separate thread, compete for resources to create various recipes. The simulation involves the use of semaphores for synchronization, shared memory for communication, and dynamic color-coding to visually distinguish between different bakers.

Project Components

The project consists of the following components:

1. Baker Threads

- Each baker is represented by a separate thread.
- Bakers compete for resources such as mixers, pantry items, refrigerator items, bowls, spoons, and the oven.
- One baker is designated randomly as "Ramsied," requiring them to restart one random recipe from the beginning.

2. Resources and Semaphores

- Ingredients (Flour, Sugar, Yeast, Baking Soda, Salt, Cinnamon, Eggs, Milk, and Butter) are located in the pantry and refrigerator.
- Semaphores are used to control access to shared resources:
    - Mixer (Counting Semaphore - 2 permits)
    - Pantry (Binary Semaphore - 1 permit)
    - Refrigerator (Counting Semaphore - 2 permits)
    - Bowl (Counting Semaphore - 3 permits)
    - Spoon (Counting Semaphore - 5 permits)
    - Oven (Binary Semaphore - 1 permit)

3. Baking Process

- Bakers acquire the necessary ingredients and utensils (bowl, spoon, mixer) to mix the ingredients.
- Once mixed, the recipe is cooked in the oven.
- Each baker completes each recipe once and announce their completion.

Sebastian Torrejon Alonzo
Fabian Kirberg

4. Color Management

- Dynamic color-coding is implemented to visually distinguish between different bakers.
- Each baker's output is displayed in a unique color for real-time tracking.

5. Ramsied Mechanisn

- One baker is randomly designated as "Ramsied" at the start of the program.
- Ramsied baker releases all acquired semaphores and restarts a recipe randomly.

Implementation Details

1. Baking Threads (`baker_thread` function)

- Each baker thread follows a similar process for creating different baked goods.
- Bakers acquire ingredients and utensils, perform mixing, and use the oven to bake the goods.
- "Ramsied" baker redo a random recipe.

2. Color Management (`getColor` function)

- The `getColor` function generates a random color for each baker, ensuring visual distinction.
- It avoids repeating colors for different bakers.

3. Main Function (`main` function)

- Collects user input to determine whether the terminal supports RGB colors.
- Takes user input for the number of bakers participating in the simulation.
- Creates threads for each baker, initializing semaphores and baker-specific data.
- Waits for all baker threads to finish.
- Cleans up semaphores after the simulation.

Sebastian Torrejon Alonzo
Fabian Kirberg

Compilation and Execution

- The program can be compiled using either System V or POSIX semaphores, depending on user preference.
- Compilation command and execution remain consistent regardless of the chosen semaphore type.

```
# Compilation with System V semaphores
gcc -o bake bakeoff.c -lpthread

# Compilation with POSIX semaphores
gcc -o bake bakeoff.c -lpthread -lrt

# Execution
./bake
```

Conclusion

The project simulates a shared kitchen environment with multiple bakers using threads, semaphores, and shared memory. The implementation ensures proper synchronization and resource management during the baking process. The inclusion of dynamic color-coding enhances the visual representation of the simulation, making it easier to track the real-time activities of each baker. The Ramsied mechanism adds an element of unpredictability to the simulation, requiring bakers to adapt to occasional challenges.