Report of Project2

Yiheng Bing

April 15, 2024

1 BurgerBuddies

1.1 Description

The BurgerBuddies problem is a classic problem in the field of operating system. The problem is that there are three types of threads: cooks, cashiers, and customers. The cooks make burgers and put them on the rack. The cashiers take the burgers from the rack and give them to the customers. The customers take the burgers from the cashiers and eat them. The problem is to synchronize the three types of threads so that the customers do not eat burgers that have not been paid for, and the cashiers do not give burgers to customers that have not been cooked.

1.2 usage

```
./BBC <cooks> <cashiers> <Customers> <rack_num>
```

1.3 Design

I use several semaphores to implement the synchronization of the three types of threads.

For the number of burgers, the cooks and the cashiers can both change it so I need a mutex to protect the number of burgers. And I use a semaphore to represent the number of burgers on the rack. The cooks would increase the semaphore when they put a burger on the rack and the cashiers would decrease the semaphore when they take a burger from the rack. And I use another two semaphores to represent the status of the customer and the cashier.

If there are customers waiting for ordering, or the cashier is ready to work for the customer's order, their semaphore will be set. If there are no burgers on the rack, the cashier would be blocked.

2 SantaClaus

2.1 Description

The SantaClaus problem is also a classic problem in the field of operating system. The problem is that there are three types of threads: elves, reindeer, and SantaClaus. Different from the problem one, the number of the three kinds of threads is fixed to 1, 9 and 10. The elves need help from SantaClaus when three of them are in trouble. The reindeer need to be hitched by SantaClaus when nine of them are ready to pull the sleigh. The SantaClaus would help the elves when three of them are in trouble, and would hitch the reindeer when nine of them are ready to pull the sleigh.

2.2 usage

```
./SantaClaus
```

2.3 Design

In this problem, I use several semaphores to implement the synchronization of the three types of threads and two mutexes to protect the number of elves and reindeer. Then as we know the number of the three kinds of threads is fixed, so I need to keep the number of elves and reindeer smaller than the limit in each pthread. SantaClaus will keep sleeping until there are three elves in trouble or nine reindeer ready to pull the sleigh. When SantaClaus is woken up, he will help the elves or hitch the reindeer and then go back to sleep. And the reindeer's requests are prior to the elves' requests. So I need to use a semaphore to protect the requests of the reindeer and keep elves waiting when Santaclaus is giving gifts with the reindeer.

In the process, I need to output the current status of the SantaClaus and the elves and reindeer using id.

Last, the project will print "Christmas is over!" when the SantaClaus has given 30 times gifts.