Homework 2 Report

Student Name and ID: B12902033 ShangJhe Li

1 Busy waiting

1.1 What is Busy Waiting

The computer is busy waiting when it is doing nothing but checking whether it should wait or there's something to do. That is, in this case, the computer is doing nothing but checking whether there is connection or data to read.

1.2 How to avoid

I use select() and poll() to avoid busy waiting. They are system calls that block for a time until there is connection or data to read or timeout.

1.3 Possibility having Busy Waiting with select()/poll()

It is possible to have busy waiting with select() and poll() if the timeout is set to 0, or less than the CPU time required for checking all the incomming data are ready or not. However, in this case, 10ms is much more than enough for the CPU to check all incomming data are ready or not (`MAX_CONNECTION = 1024`). Therefore, busy waiting is avoided.

2 Starvation

2.1 What is Starvation

Starvation is a condition where the process want a write lock (or other shared resources), but other processes are holding the read lock (or the specific resource this process needs). The process will be blocked until the read lock is released (or the resource is available). However, if more and more read locks are acquired (or other process doesn't release the resource), the process will be blocked forever.

2.2 Possibility having Starvation

Yes, it is possible to have starvation in the assignment. For example, if the read server is holding the read lock, and the write server is asking for the write lock, the write server will be blocked until the read server releases the read lock. However, if the read server keeps holding the read lock, the write server will be blocked forever.

3 File Consistency on One Process

I write the train info into an additional memory (i.e. array `trains.stat`) when the seat is paid or reserved. Furthermore, I write to the file after the seats is paid. Therefore, the file is consistent on one process.

4 File Consistency Across Multiple Process

I add lock on the seats that is accessed. Before the client chose a seat, I check if the seat is locked by other process as well as the `trains.stat`. Therefore, the file is consistent across multiple processes.