

HO CHI MINH UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING



OPERATING SYSTEM LAB (CO2018)

LAB 6 : SYNCHRONIZATION

Teacher: Tran Truong Tuan Phat
Student: Thai Phuc Hiep - 1812227

HO CHI MINH CITY, May 2020



Contents

1	PROBLEM 1 :	2
2	PROBLEM 2 :	2



1 PROBLEM 1 :

We assume that :

- `balance = 20.000.000 VND`
- The husband calls function : `withdraw(5.000.000)`
- The wife calls function : `deposit(10.000.000)`

All possible outcomes we could get :

- `withdraw()` is executed **before** `deposit()` :
 - `withdraw(5.000.000) ⇒ balance = 20.000.000 - 5.000.000 = 15.000.000 (VND)`
 - `deposit(10.000.000) ⇒ balance = 15.000.000 + 10.000.000 = 25.000.000 (VND)`
- `withdraw()` is executed **after** `deposit()` :
 - `deposit(10.000.000) ⇒ balance = 20.000.000 + 10.000.000 = 30.000.000 (VND)`
 - `withdraw(5.000.000) ⇒ balance = 30.000.000 - 5.000.000 = 25.000.000 (VND)`
- `withdraw()` and `deposit()` are executed **concurrently** :
 - `withdraw(5.000.000) ⇒ balance = 20.000.000 - 5.000.000 = 15.000.000 (VND)`
 - Concurrently, `deposit(10.000.000) ⇒ balance = 20.000.000 + 10.000.000 = 30.000.000 (VND)`

As we can see, the third case show the unexpected result, since `withdraw()` function and `deposit()` function change the balance value at the same time.

Solution : Use mutex or semaphore.

2 PROBLEM 2 :

Comparison Condition :

- Number of threads = 1000
- Number of points = 1000000

Here is the result :

- `pi_mutex` : 0.051486 s
- `pi_multi-thread` : 0.050707 s



```
Ubuntu
tphiepbk DESKTOP-J40UA6S mnt > c > ... > thanh > Desktop > OS_LAB6 $ ls -a
total 16
-rwxr-xr-x 1 tphiepbk example1 example1.c pi_multi-thread pi_multi-thread.c pi_mutex pi_mutex.c
tphiepbk DESKTOP-J40UA6S mnt > c > ... > thanh > Desktop > OS_LAB6 $ ./pi_multi-thread 1000000
Estimated PI = 3.180000
Collapsed Time = 0.050707
tphiepbk DESKTOP-J40UA6S mnt > c > ... > thanh > Desktop > OS_LAB6 $ ./pi_mutex 1000000
Estimated PI = 3.152000
Collapsed Time = 0.051486
tphiepbk DESKTOP-J40UA6S mnt > c > ... > thanh > Desktop > OS_LAB6 $ |
```

So the mutex lock approach is slower than the multi-thread approach.