PDP Lab 1 Documentation

Alex Ovidiu Popa

936/1

2. Bank Accounts

The solution involves using a mutex for each account (particularly a Java ReentrantLock()), and when a transfer is being made between two accounts, their respective mutexes are locked before their balances are changed and unlocked afterwards. However, the order in which they are locked matters, because if a1 wants to transfer to a2 and a2 to a1 at the same time, this would be a deadlock. To solve this problem, the mutexes are locked in the order of the account ids, like so:

```
if (this.uid<other.uid){
    this.mtx.lock();
    other.mtx.lock();
}
else {
    other.mtx.lock();
    this.mtx.lock();
}
```

The idea is that by using this method instead of having a single lock for the whole bank, the bottleneck is smaller than it would be with one lock, due to the fact that in this case, a1->a2 and a3->a4 could happen at the same time from different threads without any problems, whereas if we were to only have one lock, a3->a4 would only happen after a1->a2.

The consistency check occurs with a 0.1 probability, which one could say it is low, but considering the number of operations divided by the number of threads, it is actually reasonable.

To make sure the consistency check does not interfere with the other transactions, it is done by a separate “checker” thread, and its access to the resources is controlled by a separate mutex (ReentrantLock()) kept in the Bank class.

The number of accounts was always 100 when running the tests, however the operations count and number of threads were changed consistently. After each test, the time elapsed is documented.

1.5 threads, 5000 transactions

```
Time elapsed: 0.064 seconds
```

2.5 threads, 50000 transactions

```
Time elapsed: 0.576 seconds
```

3.10 threads, 50000 transactions

```
Time elapsed: 0.832 seconds
```

4.10 threads, 500000 transactions

```
Time elapsed: 192.896 seconds
```

5. 20 threads, 1000000 transactions

```
Time elapsed: 692.16 seconds
```

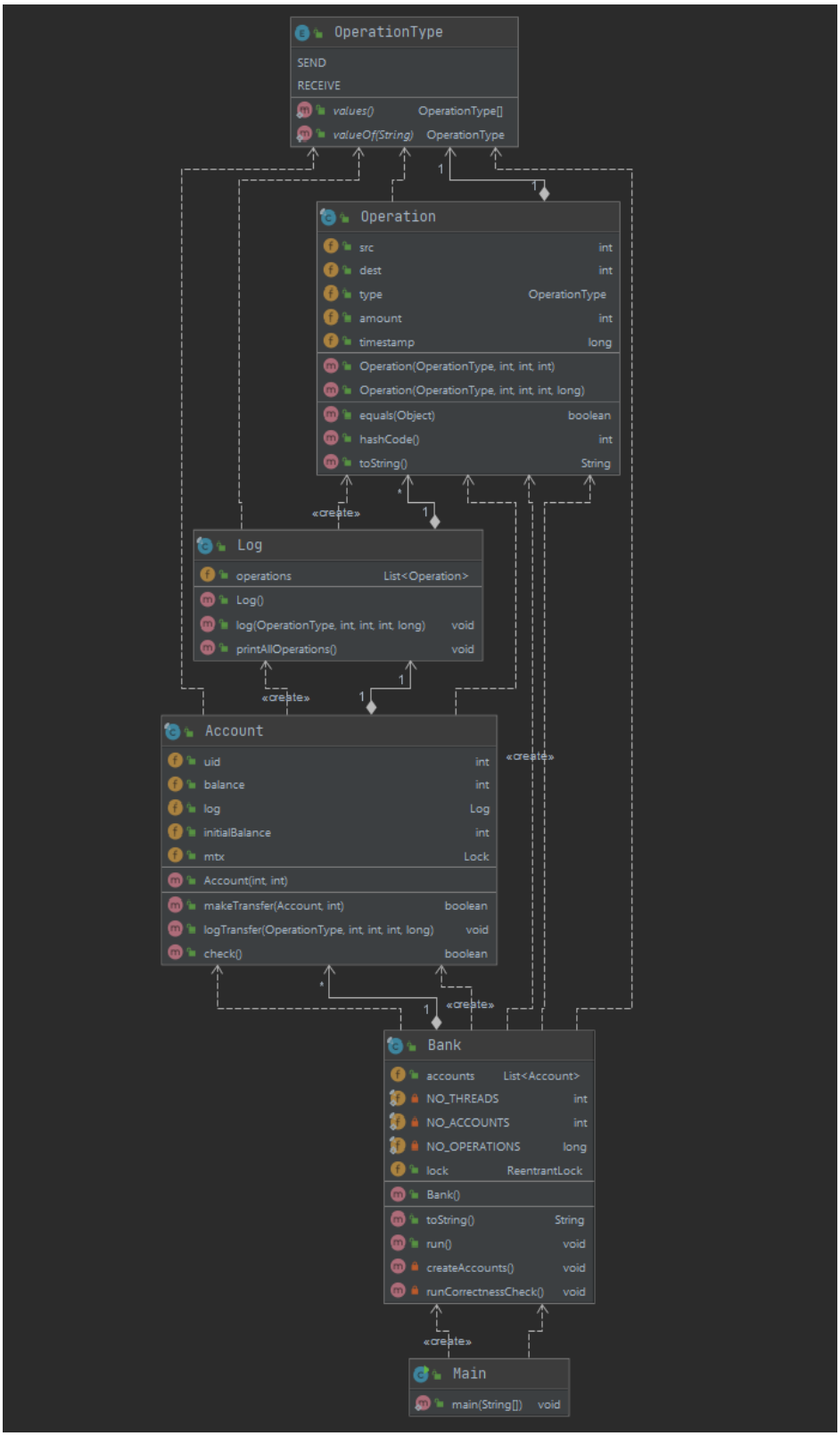
The **main conclusions** one can draw from the tests result are that:

A) The execution time increases proportionally to the number of transactions

B) Threads certainly help speed up the computations, however at some point switching between them increases the execution time, rather than decrease it (see test 2 vs test 3)

Pictured below are both the class structure/dependencies and the hardware used for testing.

Item	Value
OS Name	Microsoft Windows 10 Education
Version	10.0.17134 Build 17134
Other OS Description	Not Available
OS Manufacturer	Microsoft Corporation
System Name	ALEX-PC
System Manufacturer	To Be Filled By O.E.M.
System Model	To Be Filled By O.E.M.
System Type	x64-based PC
System SKU	To Be Filled By O.E.M.
Processor	Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz, 3601 Mhz, 4 Core(s), 8 Logical Pro...
BIOS Version/Date	American Megatrends Inc. P1.90, 22-Dec-15
SMBIOS Version	2.7
Embedded Controller Version	255.255
BIOS Mode	Legacy
BaseBoard Manufacturer	ASRock
BaseBoard Model	Not Available
BaseBoard Name	Base Board
Platform Role	Desktop
Secure Boot State	Unsupported
PCR7 Configuration	Binding Not Possible
Windows Directory	C:\WINDOWS
System Directory	C:\WINDOWS\system32
Boot Device	\Device\HarddiskVolume1
Locale	United States
Hardware Abstraction Layer	Version = "10.0.17134.1098"
User Name	ALEX-PC\Alex
Time Zone	GTB Daylight Time
Installed Physical Memory (RAM)	16.0 GB
Total Physical Memory	15.9 GB
Available Physical Memory	4.63 GB
Total Virtual Memory	20.7 GB
Available Virtual Memory	2.85 GB
Page File Space	4.80 GB
Page File	C:\pagefile.sys
Kernel DMA Protection	Off
Virtualization-based security	Not enabled
Device Encryption Support	Reasons for failed automatic device encryption: TPM is not usable, PCR7 bindi...
Hyper-V - VM Monitor Mode E...	Yes
Hyper-V - Second Level Addres...	Yes
Hyper-V - Virtualization Enable...	Yes
Hyper-V - Data Execution Prote...	Yes


OperationType
SEND
RECEIVE
values() OperationType[]
valueOf(String) OperationType
Operation
src int
dest int
type OperationType
amount int
timestamp long
Operation(OperationType, int, int, int)
Operation(OperationType, int, int, int, long)
equals(Object) boolean
hashCode() int
toString() String
Log
operations List<Operation>
Log()
log(OperationType, int, int, int, long) void
printAllOperations() void
Account
uid int
balance int
log Log
initialBalance int
mtx Lock
Account(int, int)
makeTransfer(Account, int) boolean
logTransfer(OperationType, int, int, int, long) void
check() boolean
Bank
accounts List<Account>
NO_THREADS int
NO_ACCOUNTS int
NO_OPERATIONS long
lock ReentrantLock
Bank()
toString() String
run() void
createAccounts() void
runCorrectnessCheck() void
Main
main(String[]) void
«create»
1
*