Parallel and distributed programming Documentation lab1 - Mos Daniele 935

GitHub repository: https://github.com/Daniele1209/University-year-3-projects/tree/main/Parallel and distributed programming/Lab1_multi-threading

Classes:

- Entities:
  - Operation
  - Account
- Repository

Program execution:

- Create accounts equal to the ACCOUNT_NUMBER constant, which are added to the repository with a random balance between 1000 and 5000
- After the account creation we start a timer for the transactions
- We generate as many threads in a for loop as the THREAD_NUMBER constant
- Every thread accesses the threadTransaction function which takes as parameters the operation which contains the amount of balance that needs to be transferred

```
t = threading.Thread(target = threadTransaction(transferOperation, accountX, accountY), args=(threadIndex, ))
```

- In the threadTransaction function we generate a unique ID for the log and perform the subtraction operation for the accounts that performs the transfer (in which we also check if that account has enough balance to execute the transfer, if he does not, we throw an Exception) and we log the transaction
- After the subtraction operation we perform an addition operation on the receiver account and we log this transaction as well
- Both of the transactions make use of mutexes in the Account class (in the functions addBalance and subtractBalance), we acquire the mutex before we change the balance of the account and release after the addition or subtraction operation

```
    self.balance_mutex.acquire()
    self.accountBalance += amount
    self.balance_mutex.release()
```

- We use another mutex for the log process, after the addition and subtraction operation we acquire and save the log string in the object's log dictionary (saving it at the given generated ID from earlier) and the release

```
    self.log_mutex.acquire()
    self.log[id] = logString
    self.log_mutex.release()
```

- Each log generated by the transaction process is also saved in the repository and written in a log.txt file, which contains the type of operation, the amount transferred and the account names
- For every 'n' threads generated in the loop we have a consistency check which gets the initial balance of the accounts from the repository and get the current balance from each account from the repository, then we get all the logs as we iterate through the accounts and we add or subtract the amount found in the logs on the initial balance and at the end check if the amount that we got corresponds to the current amount of the given account

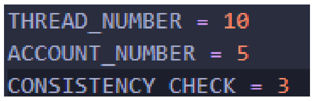
```
for log in accountLogs.values():
    log_split = log.split(',')
    if log_split == 'send':
        initialAccountBalance -= int(log_split[1].split(':')[1])
    else:
        initialAccountBalance += int(log_split[1].split(':')[1])
    if initialAccountBalance == currentAccountBalance:
        consistencyList[account.getName()] = True
```

- At the end we join the threads and print the elapsed time
- Also have another version of the threading function which contains a while loop that generate and execute transactions (so that each thread does more that just one transfer)

Statistics:

a. With 1 transaction/thread

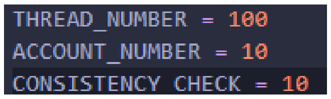
- 10 threads



```
THREAD_NUMBER = 10
ACCOUNT_NUMBER = 5
CONSISTENCY_CHECK = 3
```

Elapsed time: 0.0476

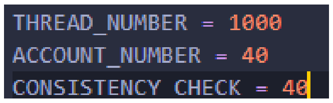
- 100 threads



```
THREAD_NUMBER = 100
ACCOUNT_NUMBER = 10
CONSISTENCY_CHECK = 10
```

Elapsed time: 0.4445

- 1000 threads

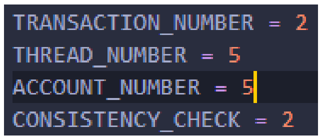


```
THREAD_NUMBER = 1000
ACCOUNT_NUMBER = 40
CONSISTENCY_CHECK = 40
```

Elapsed time: 8.2011

b. With n transactions/thread

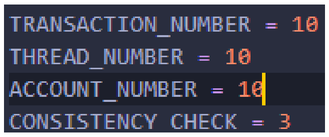
- 5 threads and 2 transactions/thread



```
TRANSACTION_NUMBER = 2
THREAD_NUMBER = 5
ACCOUNT_NUMBER = 5
CONSISTENCY_CHECK = 2
```

Elapsed time: 0.02200

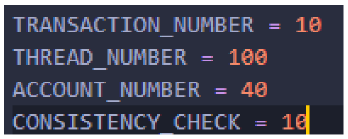
- 10 threads and 10 transactions/thread



```
TRANSACTION_NUMBER = 10
THREAD_NUMBER = 10
ACCOUNT_NUMBER = 10
CONSISTENCY_CHECK = 3
```

Elapsed time: 0.4710

- 100 threads and 10 transactions/thread



```
TRANSACTION_NUMBER = 10
THREAD_NUMBER = 100
ACCOUNT_NUMBER = 40
CONSISTENCY_CHECK = 10
```

Elapsed time: 9.6572