Construction and Verification of Software

Verifast project assignment 2019/2020

A Verified Blockchain Implementation

Change log

08-05-2020: Initial version

Introduction

This is the project assignment for the verifast project. Its goal is the verification of a simple blockchain, its corresponding data structure and associated construction algorithms.

A Blockchain

In a very simplified way, a blockchain is a linked list of data blocks that can contain information about money transactions between users. Each block also contains the hash information of the previous block in the blockchain, which guarantees the integrity of the blockchain. A block also contains a random number that is used to achieve the acceptance of a block in the blockchain. A block can be either a simple block, containing a list with a fixed number of transactions, or a summary block, that contains a summary of the balance for all users.

You mission is to implement a blockchain and a design a pool of concurrent workers that iteratively draw transactions from a queue of transactions, and tries to build a valid block for the blockchain. A valid block is composed of a fixed number of valid transactions and the corresponding hash obeys an acceptance criterion. Each work will try to compute a new block separately, and then tries to append it to the blockchain provided that the block is still eligible to be appended. Otherwise, it starts over again with the same transaction set, but using the (new) last block in the blockchain as previous. Every ten blocks, a new kind of block needs to be inserted into the blockchain with a summary of the balances of all user accounts. The insertion of a summary block is done by a special worker that only does this.

Block Acceptance Criterion

- In simple blocks all transactions must be valid in the sequence they are dequeued (i.e. the balance for all users after the transaction must be positive), and the hash of a block must end with two zeros.
- Summary blocks store the correct balances resulting from the transactions in the blockchain, and the hash of a block must end with two zeros. These balances are computed from the latest summary block in the blockchain.

Architecture

The project comprises several components:

- Transactions: A triple of integers indicating the id of the sender, recipient, and amount of the transaction.
- Blockchain blocks: A block that contains a reference to the previous block, its hash and a random number. A blockchain block can be a simple block or a summary block containing more, specialised, information. A blockchain block is valid if its hash meets an acceptance criteria (see previous section).
- Simple Blocks: a block chain block that contains a list of transactions. To be inserted in the blockchain all transactions must be valid, i.e. leave the balance for a given user in a positive state.
- Summary Blocks: a block with the balance for all users. Use an array whose indexes work as the user identifiers. Indicate the max number of users as a global constant value.
- Blockchain: A linked list of blockchain blocks, where each block contains the hash value of the previous block as a guarantee of integrity of the blockchain.
- Transaction Queue: An ADT that stores a sequence of transactions yet to be stored in the blockchain.
- Blockchain Workers: Objects that load a set of transactions from the queue of transactions and builds a new valid block by finding a new value for the random value such that the block hash satisfy the criterion.
- Pool of workers: A set of threads running the algorithm of blockchain workers.
- Main module/client code: a class that adds new transactions to the queue to be processed.

Project acceptance criteria

You must complete the given Java code to implement the requirements. Your code must compile and verify using verifast. The annotations provided should represent the weakest precondition and the strongest precondition possible. Not all guaranties enunciated in this assignment are easy to be statically verified by verifast, be conservative and ensure the absnce of runtime errors first.

Evaluation

You should work in teams of two students and evaluation will include an online discussion with individual questions to both students about the project

development and other topics related to the course.

Code of conduct

- It is expected from you to be the author of all code that you submit as your own.
- It is expected that the work submitted as a team is the result of true collaborative work.
- It is expected that you don't share any Java code, verifast specifications, or comments that lead to concrete solutions with anyone else than your team mate.
- Any detected miscondict can be punished with failing the course.

Starter code

Consider the following files as starter code for your project

- `Blockchain.jarsrc`

  This file is the "project" file, that gathers all the files involved in the verification project. Its contents is the following:

  ```
  Block.java
  SimpleBlock.java
  SummaryBlock.java
  Transaction.java
  ```

- `Block.java`
- `SimpleBlock.java`
- `SummaryBlock.java`
- `Transaction.java`

Important dates

The project must be submitted in two phases:

- 19th of May - Intermediate submission: the sequential blockchain ADT, and a client that adds new blocks to the blockchain.
- 29th of May - Final submission: The concurrent pool of workers that build valid blocks and a queue of transactions that is accessed concurrently to form new blocks in the blockchain.

Submission details

The project can be submitted via a google form whose link will be made available in due time.