UNIVERSIDAD NACIONAL DE LUJAN

SISTEMAS DISTRIBUIDOS Y PROGRAMACION PARALELA

BLOCKCHAIN PARALELA

DOCENTES

Rodriguez Caillava, Juan Martin
Petrocelli, David

INTEGRANTES

Sciarrotta, Nicolas
Prados, Nehuén

2020

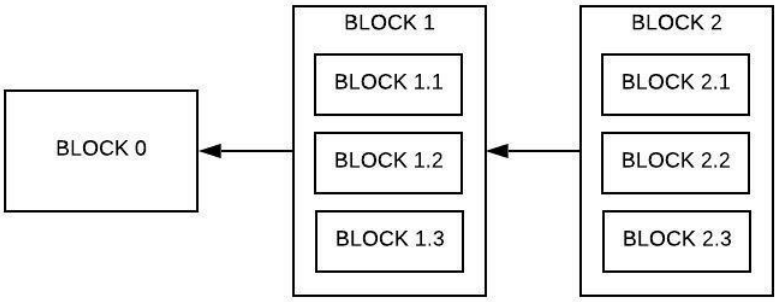
La propuesta del trabajo final, consiste en construir un prototipo de blockchain [1] paralelizable.

El concepto básico de blockchain es bastante simple: una base de datos que mantiene una lista en continuo crecimiento de registros ordenados:

Algo muy similar a un log de transacciones de una base de datos.

Como se puede observar en la imagen, existe un orden y una secuencialidad en las operaciones que se registran en una blockchain, haciendo que, si bien el contenido de cada bloque se puede generar de forma distribuida, su procesamiento deba ser centralizado.

El objetivo de este proyecto, es presentar un prototipo de arquitectura que permita paralelizar la generación de bloques, gráficamente sería algo así:

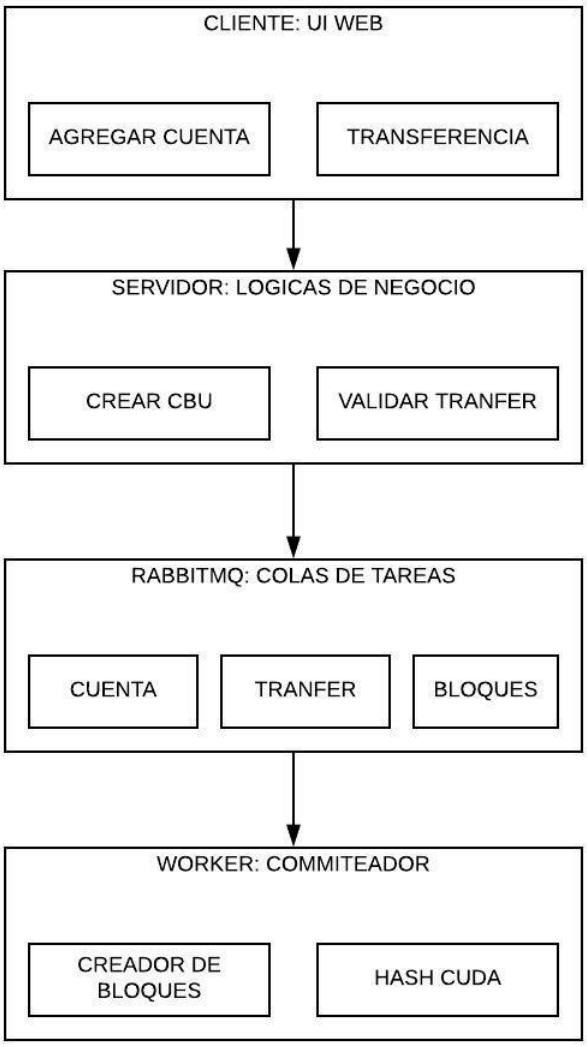


La principal ventaja de esta arquitectura frente a las actuales, es que, si dos operaciones no son mutuamente excluyentes o secuenciales, pueden ser realizadas en paralelo, a diferencia de las blockchain actuales que fuerzan a que sea secuencial.

Para lograr esto, se propone utilizar herramientas vistas en la materia como RabbitMQ para el manejo de colas de los bloques, CUDA para el cálculo criptográfico intensivo de hashes y resolución de desafíos y una arquitectura cliente servidor para la comunicación con los usuarios.

Los goles principales serán:

1. Mecanismo de sincronización y bloqueos empleando RabbitMQ
2. Mecanismo de calculo de hashes empleando CUDA
3. Lógicas / Validaciones de negocio parametrizables
4. UI Web para el usuario final



Arquitectura

A grandes rasgos, la arquitectura general pude verse como en la imagen de la izquierda, en la UI web el usuario contara con la posibilidad de solicitar aperturas de cuenta y transferencias de fondos además de una pantalla de solo lectura donde podra ver los saldos consolidados de todas las cuentas y historial de los bloques procesados.

En la capa del servidor, se encontrará la lógica del negocio, aquellas validaciones sobre las entradas de los usuarios, si esta todo bien, la solicitud se encolara en el server rabbitmq.

El server de rabbitmq contiene las colas de operaciones pendientes y una cola para cada cuenta, sirviendo esto como herramienta de sincronización distribuida.

Los workers tendrán como función commitear los bloques, para esto hay dos funciones principales, la de armar los bloques y la de procesarlos, el creador lee de rabbitmq las operaciones pendientes y las encola en bloques, el procesador lee de la cola de bloques pendientes y las procesa utilizando CUDA para el cálculo de los challenge.

Estructura de bloque

El primer paso lógico es decidir la estructura del bloque. Para mantener las cosas lo más simples posible, incluimos solo las necesarias: índice, marca de tiempo, datos, hash y hash previo.

El hash del bloque anterior debe encontrarse en el bloque para preservar la integridad de la cadena

Hash de bloque

El bloque necesita ser hasheado para mantener la integridad de los datos. Un SHA-256 se toma sobre el contenido del bloque (incluyendo marca de tiempo, datos, índice y hash previo).

Para generar un bloque, debemos conocer el hash del bloque anterior y crear el resto del contenido requerido (índice, hash, datos y marca de tiempo). Los datos de bloque

son algo proporcionado por el usuario final sobre los cuales se aplican validaciones de negocio.

El hash de cada bloque a agregar será realizado mediante una prueba de trabajo (minería) donde se utilizará el cómputo paralelo para que los mineros compitan por los siguientes bloques a incorporar.

Además de las tecnologías antes mencionadas, se implementará una base de datos Rethinkdb [2] para el almacenamiento de los bloques ya procesados y la historial, a fines de ser usada como repositorio de solo lectura de los bloques procesados.

Con respecto a las métricas, se propone crear dos workers procesadores, uno en CUDA y otro en secuencial, de manera tal que se pueda comprar el rendimiento de ambos a distintas escalas de challenges.

Además de propone crear una variable que sea máximas transacciones por bloque, de manera tal que si se la setea en 1, seria una blockchain tradicional y si se le aumenta el numero se pueda medir la ventaja en rendimiento de esta estructura.

[1] https://en.wikipedia.org/wiki/Blockchain_%28database%29

[2] https://www.rethinkdb.com/docs/sharding-and-replication/