Algoritmo Bizantino

Vera Catashunga, Alejandro Vidal

Programación Concurrente y Distribuida

November 28, 2021

Tabla de contenidos

Introducción

Bizancio era la capital del Imperio Romano del Este. Debido al vasto territorio en ese momento, la guarnición de cada ejército estaba muy separada, y los generales tuvieron que depender de mensajeros para transmitir las noticias. Cuando luchan, todos los generales del ejército bizantino deben llegar a un consenso para ganar mejor. Sin embargo, puede haber traidores en el ejército, lo que puede causar los siguientes problemas:

- Los traidores pueden engañar a algunos generales para que tomen acciones ofensivas.
- El traidor puede alentar a otros generales a actuar.
- Un traidor puede confundir a otros generales y hacer que reciban información inconsistente, lo que puede ser confuso.

Por lo tanto, los generales deben tener un acuerdo de método predeterminado para permitir que todos los generales leales estén de acuerdo, y unos pocos traidores no pueden hacer que los generales leales hagan el plan equivocado.

Problema General Bizantino (Tolerancia bizantina a fallas)

En un sistema distribuido, especialmente en un entorno de red blockchain, también es similar al entorno de los generales bizantinos, con nodos buenos (g. b. leales), nodos malos y nodos maliciosos (g. b. rebeldes). Generalmente, estos nodos fallidos se denominan nodos bizantinos, y los nodos normales son nodos no bizantinos.

La hipótesis de Bizancio es un modelo del mundo real: las computadoras y las redes pueden comportarse inesperadamente debido a errores de hardware, congestión o desconexión de la red y ataques maliciosos.

El núcleo del algoritmo de consenso es formar consenso entre los nodos normales.

Para resolver el problema general bizantino, se deben cumplir las dos condiciones siguientes:

- Cada general leal debe recibir el mismo valor de comando v_i (v_i es el orden del general i-ésimo)
- Si el i-ésimo general es leal, el orden que envía es el mismo que el recibido por cada general leal

El problema de los generales bizantinos puede describirse como: Un general que envía una orden envía un comando a los generales n-1 restantes, de modo que:

- Todos los generales leales que reciben órdenes obedecen las mismas órdenes
- Si el general al mando es leal, entonces otros generales leales obedecen la orden recibida

Problema y solución

En el caso clásico, podemos encontrar dos formas:

- Acuerdo oral
- Acuerdo escrito

Acuerdo oral

Información verbal Incluso si los generales envían a alguien para transmitir el mensaje, se deben cumplir las siguientes condiciones:

- Cada mensaje enviado puede ser entregado correctamente
- Los destinatarios saben quién envió el mensaje
- Capacidad para conocer mensajes perdidos

Para entender mejor, supongamos que tenemos 10 generales. Cualquiera de ellos envía un mensajero para enviar un mensaje. Los otros 2-10 generales reciben un mensaje de 10. Los generales se dicen directamente. En una ronda, cada general tiene información (+ propio mensaje), si hay traidores la información puede ser inconsistente, por lo tanto, cómo tomar una decisión, si más de la mitad de los generales deciden atacar, en resumen, el ataque es que la minoría obedece a la mayoría.

Acuerdo escrito

En comparación con los acuerdos verbales, los acuerdos escritos serán sellados (firmados) por los generales si acuerdan ejecutar el acuerdo.

- Este sello no se puede falsificar, una vez que se puede encontrar la manipulación
- Cualquier general puede verificar la fiabilidad de los sellos entre sí

En comparación con los acuerdos orales, los acuerdos escritos resuelven el problema de la trazabilidad, pero aún enfrentan algunos problemas:

- La transmisión de información es lenta
- No se puede evitar la falsificación de sellos
- Las cartas se guardan y nadie puede ser garantizado

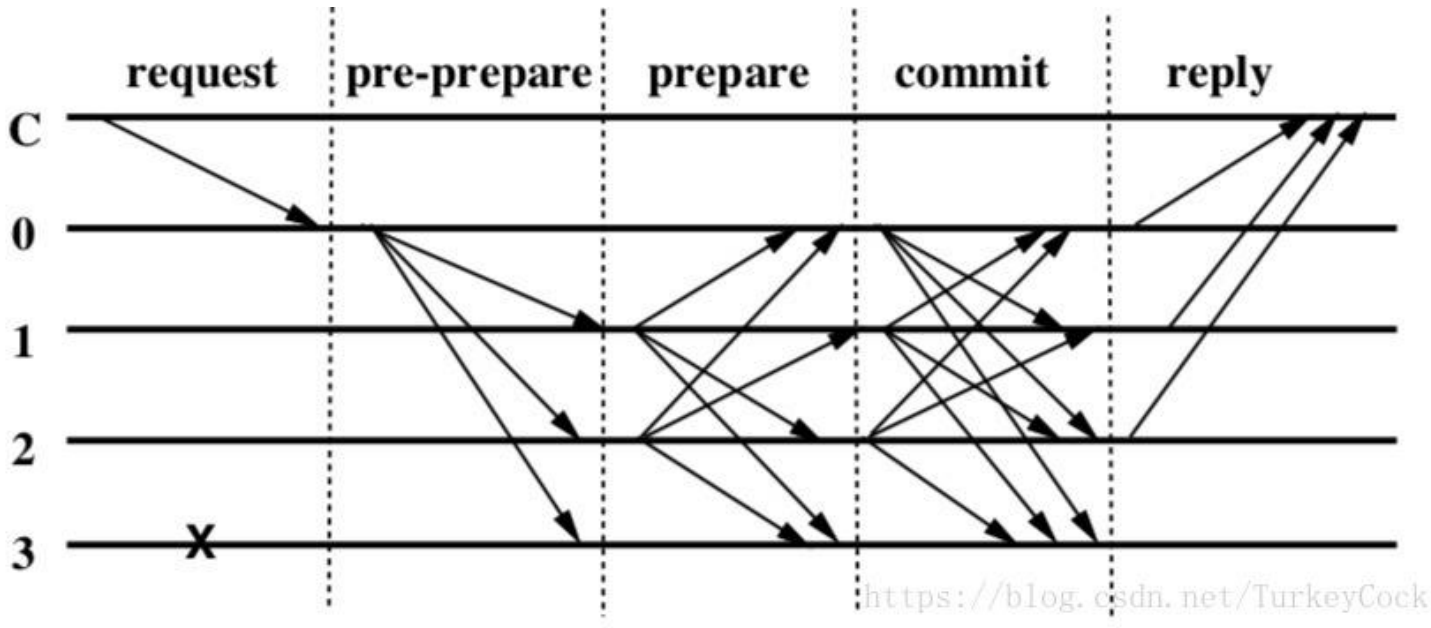
Algoritmo PBFT

Previamente, se deben comprender algunos conceptos básicos:

- **Réplica:** nodo Blockchain, proporciona un servicio de "réplica de réplica"
- **Cliente:** nodo del cliente que inicia la solicitud al primario, que se combina con el primario en la cadena de bloques
- **Primario:** iniciador de bloque, genera un nuevo bloque y lo transmite después de recibir la solicitud
- **Copia de seguridad:** el verificador de bloque verificará después de recibir el bloqueo y luego transmitirá el resultado de la verificación para consenso
- **Vista:** una copia de seguridad principal y múltiples forman una vista, y se hace un consenso sobre un bloque en la vista
- **Número de secuencia (n):** número especificado por el primario, que puede entenderse como la altura del bloque
- **Punto de control:** si el bloque correspondiente a un número de secuencia recibe más de 2/3 de la confirmación, se denomina punto de control

Acuerdo de tres fases

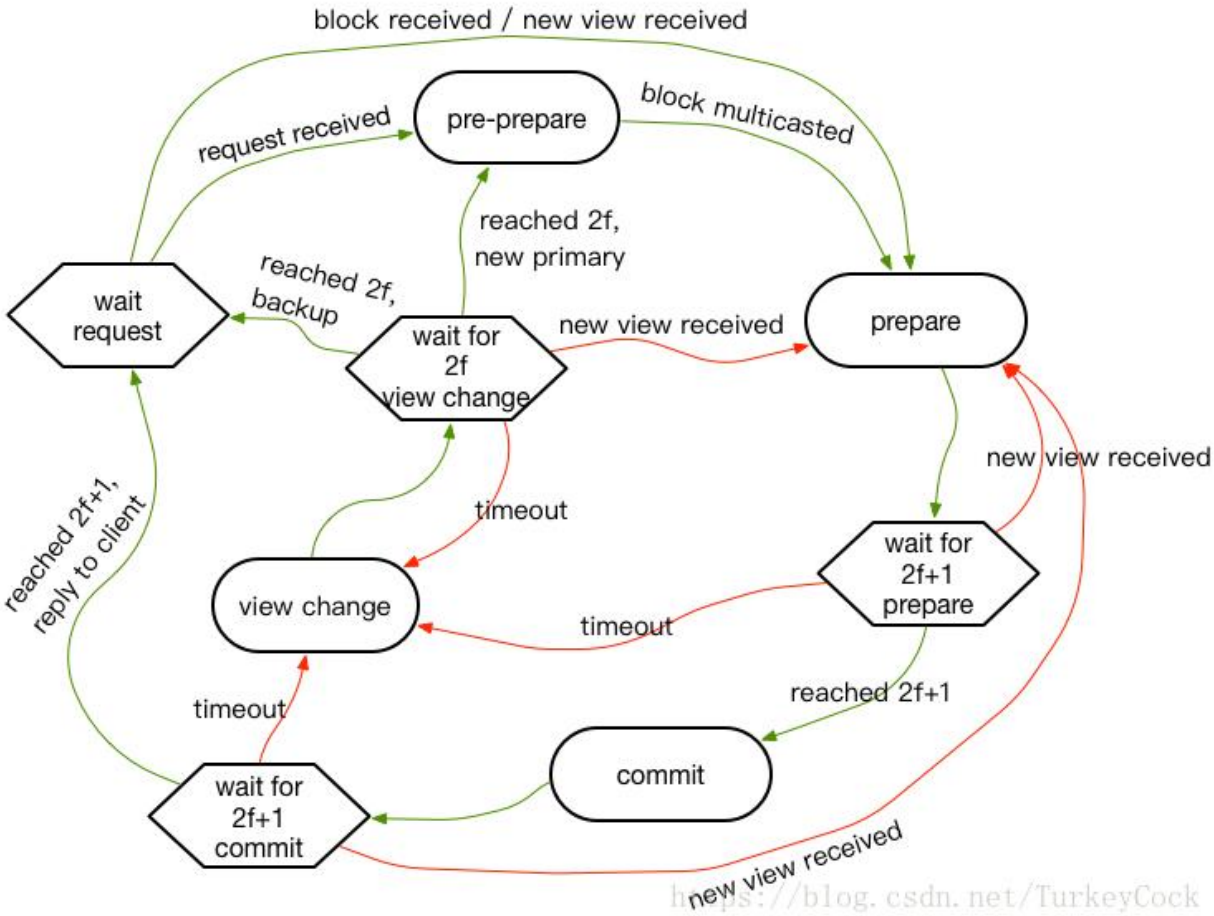
El acuerdo de tres fases es el núcleo de PBFT.



Desde el inicio de la solicitud hasta la recepción final de la respuesta, el proceso de consenso en el medio debe pasar por 3 etapas:

- **Pre-prepare:** el primario recibe la solicitud, genera un nuevo bloque y lo transmite
- **Preparar:** después de que todas las réplicas reciben el bloque, transmiten el resultado de la verificación del bloque mientras esperan recibir transmisiones de más de 2/3 nodos
- **Confirmar:** después de recibir 2/3 de la transmisión del nodo o el tiempo de espera, envíe la transmisión nuevamente y vuelva a esperar para recibir la transmisión de más de 2/3 nodos

Máquina de estado PBFT



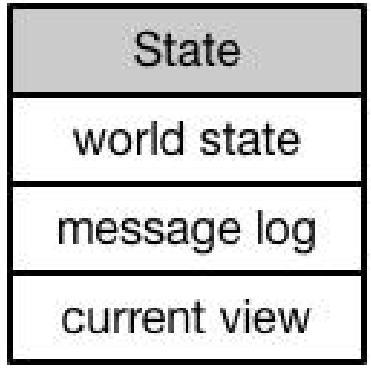
El rectángulo redondeado representa el estado, el hexágono representa la etapa de espera, la línea verde representa el proceso normal y la línea roja representa el proceso anormal.

- **wait request**
  - Esperando el estado de la solicitud, todos los nodos están inicialmente en este estado.
  - Una vez que el primario recibe el mensaje REQUEST, cambiará al estado de preparación previa
  - Una vez que la copia de seguridad recibe el bloque, cambiará al estado de preparación
- **pre-prepare**
  - Este estado es exclusivo del primario. Después de que el primario genera un bloque y difunde el mensaje PRE-PREPARE, pasa al estado de preparación
- **prepare**
  - Después de entrar en este estado, transmita el mensaje PREPARE y espere a que los nodos $2f + 1$ confirmen
- **wait for 2f+1 prepare**
  - Si espera que los nodos $2f + 1$ confirmen (aceptar o rechazar), cambie al estado de confirmación
  - Si se agota el tiempo de espera, cambie a la vista de cambio de estado
- **commit**
  - Después de entrar en este estado, transmita el mensaje COMMIT y espere a que los nodos $2f + 1$ confirmen

- **wait for 2f+1 commit**
  - Si espera que los nodos 2f + 1 confirmen (aceptar o rechazar), envíe un mensaje de RESPUESTA y vuelva al estado de solicitud de espera
  - Si se agota el tiempo de espera, cambie a la vista de cambio de estado
- **view change**
  - Después de entrar en este estado, transmita el mensaje VIEW-CHNAGE de v + 1, y espere recibir el mensaje VIEW-CHANGE de los nodos 2f
- **wait for 2f view change:** Este estado es más complicado y se puede dividir en las siguientes 4 situaciones:
  - Recibió el mensaje VIEW-CHANGE de los nodos 2f, y es un nuevo primario, transmita el mensaje NEW-VIEW y cambie al estado de preparación previa
  - Recibió el mensaje VIEW-CHANGE de los nodos 2f, y es una copia de seguridad, cambie al estado de solicitud de espera
  - Recibir tiempo de espera, volver para ver el cambio de estado, transmitir v + 2 VIEW-CHANGE mensaje
  - Antes de recibir el mensaje VIEW-CHANGE de los nodos 2f, si se recibe el mensaje NEW-VIEW, cambiará al estado de preparación

Estructura de datos

A continuación, se presenta la estructura de datos relacionada, principalmente el estado y los mensajes.



El estado de un nodo consta principalmente de tres partes:

- Estado mundial (es decir, la información de bloque más reciente)
- Registro de mensajes
- Vista actual

Three Phase Protocol

REQUEST
operation
timestamp
client id
signature

PRE-PREPARE
block
view id
seq number
digest(block)
signature

PREPARE
view id
seq number
digest(block)
replica index
signature

COMMIT
view id
seq number
digest(block)
replica index
signature

REPLY
view id
timestamp
replica index
result

La estructura del mensaje relacionada con el protocolo de tres fases se enumera aquí. El mensaje PRE-PREPARE contiene el bloque recién generado y los otros mensajes contienen id, número de secuencia, resumen del contenido del bloque y firma.

VIEW-CHANGE

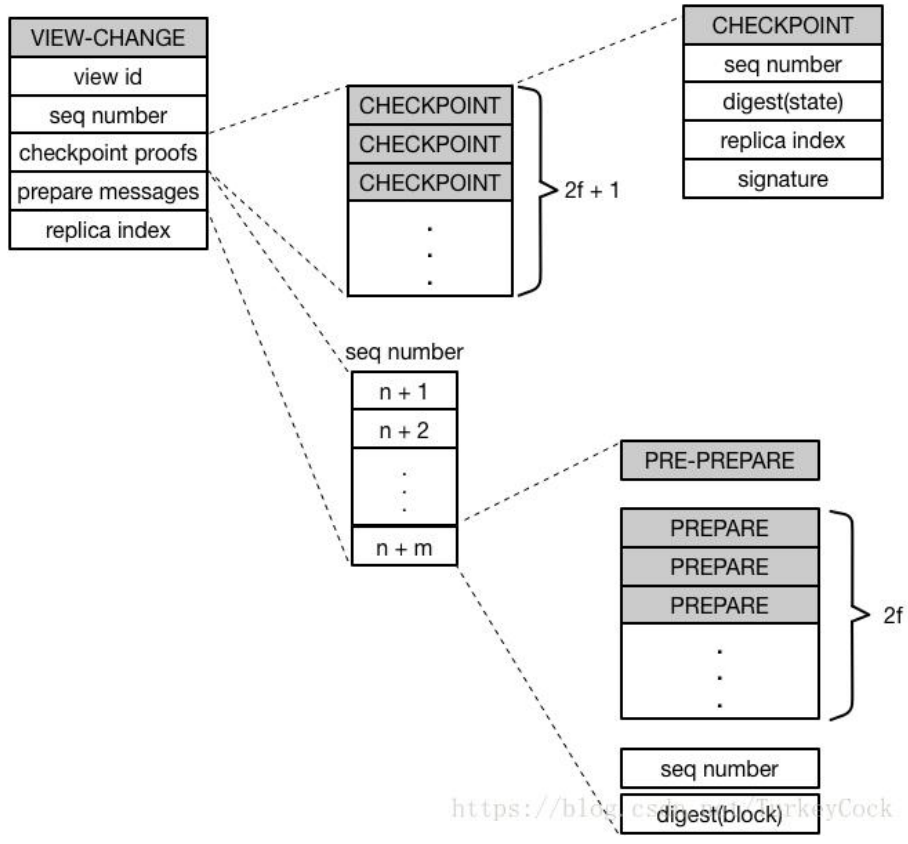
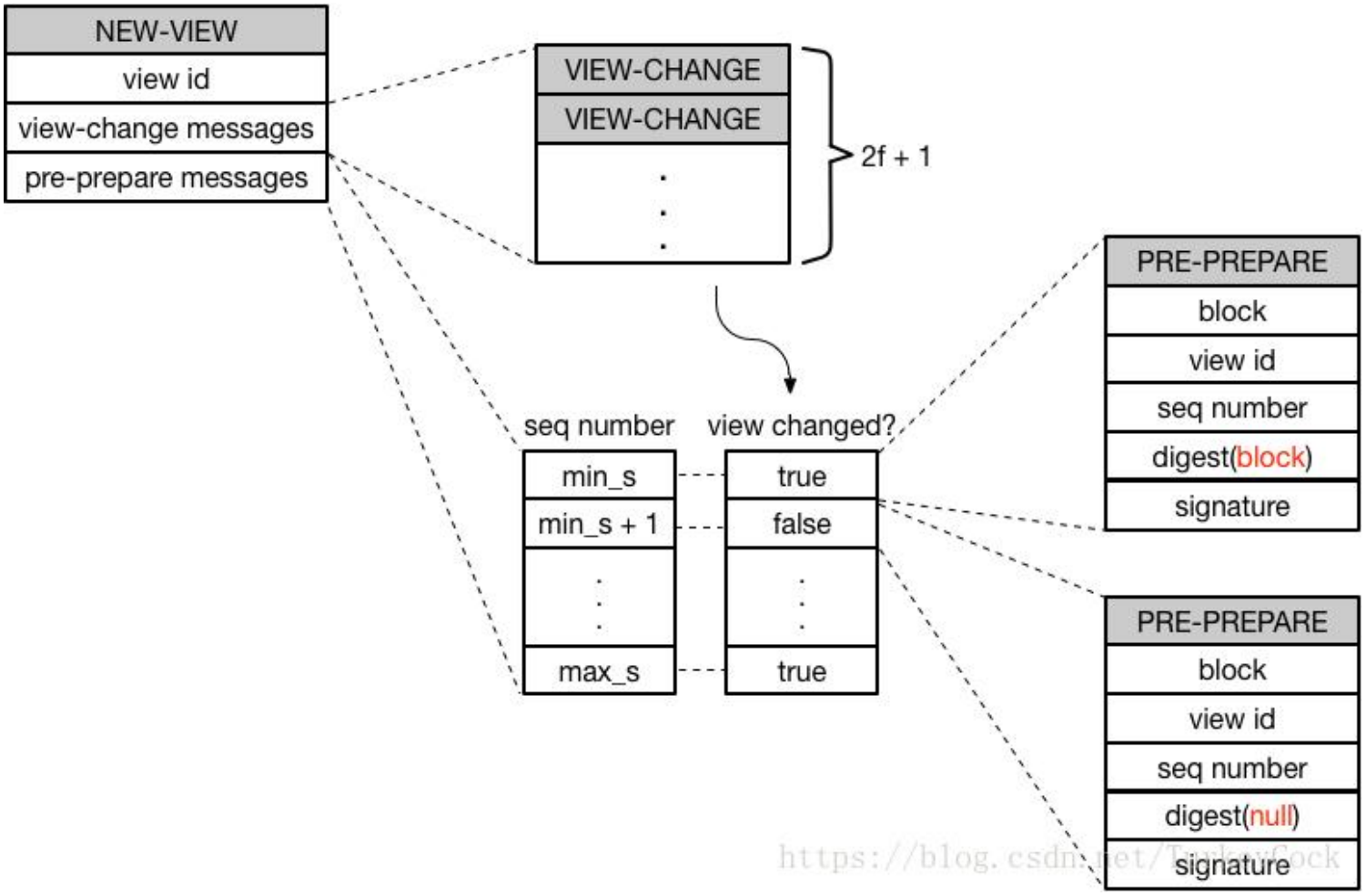


Figure 1: Debe basarse en un punto de control estable, por lo que debe contener mensajes 2f+1 CHECKPOINT para demostrar que el punto de control es válido. Entonces, los números de secuencia por encima del punto de control deben incluir el mensaje PRE-PREPARE correspondiente y los mensajes 2f PREPARE.

NEW-VIEW


NEW-VIEW
view id
view-change messages
pre-prepare messages
VIEW-CHANGE
VIEW-CHANGE
2f + 1
seq number
view changed?
min_s
min_s + 1
max_s
true
false
true
PRE-PREPARE
block
view id
seq number
digest(block)
signature
PRE-PREPARE
block
view id
seq number
digest(null)
signature

El mensaje NEW-VIEW primero debe contener mensajes 2f + 1 VIEW-CHANGE para demostrar que más de 2/3 de los nodos están de acuerdo con una nueva ronda de consenso sobre una vista superior.

Luego, de acuerdo con la información del punto de control en todos los mensajes VIEW-CHANGE recibidos, encuentre el mínimo mins y el máximo maxs, y empaque el PRE-PREPARE correspondiente a cada número de secuencia en el intervalo Noticias.

En particular, para reducir la verificación repetida, si nunca se ha realizado un cambio de vista en un determinado número de secuencia (es decir, se llega a un consenso en la primera ronda), PRE-PREPARE contiene un Información resumida de la solicitud nula.

Referencias

- https://programmerclick.com/article/50941591477/
- http://pmg.csail.mit.edu/papers/osdi99.pdf
- https://programmerclick.com/article/6581269295/
- https://programmerclick.com/article/650447965/
- https://www.youtube.com/watch?v=1QUmFGmeSOQ