# Problema de los Generales Bizantinos

Introducción

Los sistemas informáticos confiables deben manejar los componentes defectuosos que proporcionan información conflictiva a diferentes partes del sistema. Esta situación se puede expresar de manera abstracta en términos de un grupo de generales del ejército bizantino acompañados con sus tropas alrededor de una ciudad enemiga. Comunicándose solo por un mensajero, los generales deben ponerse de acuerdo sobre un plan de batalla común. Sin embargo, uno o más de ellos pueden ser traidores que tratarán de confundir a los demás. El problema es encontrar un algoritmo para asegurar que los generales leales lleguen a un acuerdo.

Los generales deben tener un algoritmo para garantizar que:

* Todos los generales leales deciden sobre el mismo plan de acción.
* Un pequeño número de traidores no puede hacer que los generales leales adopten un mal plan.

Para que se cumpla la primera condición debe cumplirse lo siguiente:

* Todo general leal debe obtener la misma información v (1) ...., v (n).

Tenemos el siguiente requisito para cada i:

Si el general i-esimo es leal, entonces el valor que envía debe ser utilizado por cada general leal como el valor de v (i).

Resultados de imposibilidad

Su dificultad está indicada por el sorprendente hecho de que si los generales solo pueden enviar mensajes orales, ninguna solución funcionará a menos que más de dos tercios de los generales sean leales.

Un mensaje oral es aquel cuyo contenido está completamente bajo el control del remitente, por lo que un remitente traidor puede transmitir cualquier mensaje posible.

Una solución con mensajes orales

Supone que cada general ejecuta algún algoritmo que involucra el envío de mensajes a los otros generales, y asumimos que un general leal ejecuta correctamente su algoritmo. La definición de un mensaje oral está incorporada en las siguientes suposiciones que hacemos para el sistema de mensajes de los generales:

* Todos los mensajes que se envían se entregan correctamente.
* El receptor de un mensaje sabe quién lo envió.
* Se puede detectar la ausencia de un mensaje.

Se define inductivamente los algoritmos de Mensajes Orales OM(m), para todos los enteros no negativos m, mediante los cuales un comandante envía una orden a n-1 tenientes. Mostramos que OM(m) resuelve el problema de los generales bizantinos para 3m + 1 o más generales en presencia de un máximo de m traidores.

Para probar la corrección del algoritmo OM(m) para m arbitraria, primero probamos el siguiente lema.

LEMMA 1. Para cualquier m y k, el algoritmo OM(m) satisface IC2 si hay más de 2k + m generales y como máximo k traidores.

El siguiente teorema afirma que el algoritmo OM(m) resuelve el problema de los generales bizantinos.

TEOREMA 1. Para cualquier m, el algoritmo OM(m) satisface las condiciones IC1 e IC2 si hay más de 3 m de generales y como máximo m traidores.

Una solución con mensajes firmados

El problema se vuelve más fácil de resolver si podemos restringir la habilidad de poder mentir. Una forma de hacer esto es permitir que los generales envíen mensajes firmados imperdonables. Más precisamente, agregamos

(a) La firma de un general leal no puede ser falsificada, y se puede detectar cualquier alteración del contenido de sus mensajes firmados.

(b) Cualquiera puede verificar la autenticidad de la firma de un general.

Perder rutas de comunicación

Suponemos que las barreras físicas imponen algunas restricciones sobre quién puede enviar mensajes a quién. Consideramos que los generales forman los nodos de un grafo G finito no dirigido simple, donde un arco entre dos nodos indica que esos dos generales pueden enviar mensajes

Extendemos OM (m) a un algoritmo que resuelve el problema de los generales bizantinos en presencia de m traidores si el grafo G de los generales es 3 m-regular. Para todos los enteros positivos m y p, definimos el algoritmo OM (m, p) de la siguiente manera cuando el grafo G de generales es p-regular.

Ahora probamos que OM (m, 3m) resuelve el problema de los generales bizantinos si hay a lo sumo m traidores. La prueba es similar a la prueba para el algoritmo OM (m) y solo se bosquejará. Comienza con la siguiente extensión de Lemma 1.

LEMMA 2. Para cualquier m> 0 y p>= 2k + m, el algoritmo OM (m, p) satisface IC2 si hay como máximo k traidores.

La corrección del algoritmo OM(m, 3m) es una consecuencia inmediata del siguiente resultado.

TEOREMA 3. Para cualquier m> 0 y cualquier p >= 3m, el Algoritmo OM (m, p) resuelve el problema de los generales bizantinos si hay como máximo m traidores.

Probamos el siguiente resultado más general, donde el diámetro del grafo es el número más pequeño d, de manera que dos nodos están conectados por una ruta que contiene como máximo d arcos.

TEOREMA 4. Para cualquier m y d, si hay a lo sumo m traidores y el subgrafo de generales leales tiene un diámetro d, entonces el algoritmo SM (m + d - 1) resuelve el problema de los generales bizantinos.

Sistemas confiables

La única forma que se conoce para implementar un sistema informático confiable es usar varios "procesadores" diferentes para calcular el mismo resultado y luego realizar una mayoría de votos en sus resultados para obtener un solo valor.

El uso de la votación por mayoría para lograr confiabilidad se basa en el supuesto de que todos los procesadores no defectuosos producirán el mismo resultado.

Además, diferentes procesadores pueden obtener diferentes valores incluso desde una unidad de entrada no defectuosa si leen el valor mientras está cambiando.

Para que la votación por mayoría produzca un sistema confiable, se deben cumplir las dos condiciones siguientes:

* Todos los procesadores no defectuosos deben usar el mismo valor de entrada (para que produzcan la misma salida).
* Si la unidad de entrada no está defectuosa, entonces todos los procesos no defectuosos utilizan el valor que proporciona como entrada (para que produzcan la salida correcta).

Estas son solo nuestras condiciones de consistencia interactivas IC1 e IC2, donde el "comandante" es la unidad que genera la entrada, los "tenientes" son los procesadores y "leal" significa que no es defectuoso.

Ahora consideramos estas suposiciones en orden.

A1. Indica que todos los mensajes enviados por un procesador que no está defectuoso se entregan correctamente. En sistemas reales, las líneas de comunicación pueden fallar. Para los algoritmos de mensajes orales OM (m) y OM (m, p), el fallo de la línea de comunicación que une dos procesadores es indistinguible del fallo de uno de los procesadores. Por lo tanto, solo podemos garantizar que estos algoritmos funcionarán en presencia de hasta m fallas, ya sean fallas del procesador o de la línea de comunicación.

A2. Indica que un procesador puede determinar el originador de cualquier mensaje que recibió. Lo que realmente es necesario es que un procesador defectuoso no pueda hacerse pasar por uno defectuoso.

A3. Requiere que se pueda detectar la ausencia de un mensaje. La ausencia de un mensaje solo puede detectarse si no llega dentro de un período de tiempo fijo.

Se puede demostrar que ningún algoritmo de este tipo puede resolver el Problema de los Generales Bizantinos si los mensajes pueden transmitirse arbitrariamente rápidamente, incluso si hay es un límite superior en el retardo de transmisión del mensaje. Además, ninguna solución es posible, incluso si restringimos a los traidores para que el único comportamiento incorrecto que se les permita sea el hecho de no enviar un mensaje.

No hay dos relojes que funcionen exactamente a la misma velocidad, por lo que no importa la precisión con la que estén sincronizados inicialmente los relojes de los procesadores, finalmente se separarán arbitrariamente a menos que se vuelvan a sincronizar periódicamente. Por lo tanto, tenemos el problema de mantener todos los relojes de los procesadores sincronizados con una cantidad fija, incluso si algunos de los procesadores están defectuosos.

A4. Requiere que los procesadores puedan firmar sus mensajes de forma tal que no se pueda falsificar la firma de un procesador que no sea defectuoso. Para cumplir con las partes (a) y (b) de A4, la función Si debe tener las siguientes dos propiedades:

(a) Si el procesador i no está defectuoso, entonces ningún procesador defectuoso puede generar Si(M).

(b) Dados M y X, cualquier proceso puede determinar si X es igual a Si(M).

La propiedad (a) nunca se puede garantizar, ya que Si (M) es solo un elemento de datos, y un procesador defectuoso podría generar cualquier elemento de datos.

Cómo se hace esto depende del tipo de fallas que esperamos encontrar. Hay dos casos de interés:

* Mal funcionamiento aleatorio. Al hacer de Si una función "aleatoria" adecuada, podemos hacer que la probabilidad de que un mal funcionamiento aleatorio en un procesador genere una firma correcta sea esencialmente igual a la probabilidad de que lo haga a través de un procedimiento de elección aleatoria, es decir, el recíproco del número de posibles firmas.

Inteligencia maliciosa. Si el procesador defectuoso está siendo guiado por una inteligencia maliciosa, por ejemplo, si se trata de un procesador perfectamente bueno operado por un humano que intenta interrumpir el sistema, entonces la construcción