Sistemas Operativos

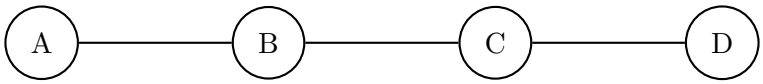
Práctica 7: Sistemas distribuidos

Notas preliminares

- Los ejercicios marcados con el símbolo ★ constituyen un subconjunto mínimo de ejercitación. Sin embargo, aconsejamos fuertemente hacer todos los ejercicios.

Ejercicio 1 ★

Un sistema distribuido tiene cuatro nodos: A, B, C y D, conectados de la siguiente forma:



Responda las siguientes preguntas justificando:

a) Dados los siguientes escenarios:

- El nodo B se cae
- El enlace entre A y B se cae.
- B está muy sobrecargado, y su tiempo de respuesta es 100 veces mayor a lo normal.

¿Puede A discernir entre cada uno de ellos?

b) Si A recibe un mensaje de D, a través de B, ¿Se puede asumir que D no está caído?.

c) Si B recibe un mensaje de A y uno de C, ¿Se puede saber si A envio su mensaje antes que C, o viceversa? ¿Por qué?

Ejercicio 2 ★

Suponer un sistema donde los n procesadores están comunicados mediante un bus ultra rápido de baja latencia, de manera tal que los tiempos de acceso a memoria remota son comparables con los tiempos locales. Imaginar que se cuenta además con un entorno de programación que permite manejar la memoria remota como si fuera local. ¿Consideraría a tal sistema como *distribuido* o *paralelo*? Justificar.

Ejercicio 3

Un algoritmo de *commit* distribuido funciona de la siguiente manera: opera sobre una red donde los paquetes pueden perderse o demorarse, y cuando un nodo quiere escribir sobre cierto archivo que está replicado en todos los nodos, envía un pedido de escritura. Si recibe confirmación de todos los nodos, escribe y le notifica a los demás que pueden hacer lo propio. Alguien nota que este algoritmo puede fallar si los nodos se caen entre la escritura y la notificación, y propone para solucionarlo el envío de mensajes de confirmación de parte de los nodos. ¿Este algoritmo modificado resuelve el problema? Justificar.

Ejercicio 4 ★

Se tiene un sistema distribuido donde los nodos están conectados bajo una topología de anillo, cada uno con un ID que los identifica. Se quiere implementar un algoritmo de elección de líder donde aquel elegido sea el que tenga el menor ID.

a) Proponga un protocolo para resolver este problema. Analize la complejidad de su protocolo con respecto a la cantidad de mensajes que se utilizan.
b) Si su solución tuvo $O(n^2)$ cantidad de mensajes utilizados, proponga otro protocolo que disminuya esa cantidad.

Ayuda: Considere un protocolo por fases y en elegir líderes en un vecindario local de cada nodo.

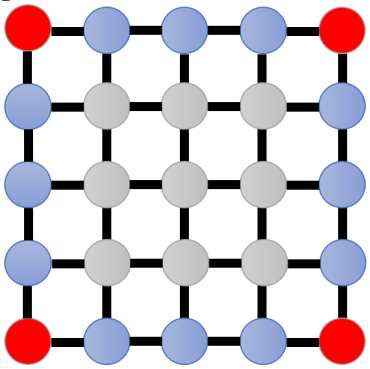
Ejercicio 5

Se tiene replicas de una base de datos en m nodos distintos, cada uno con su propio identificador. Estos nodos estan distribuidos en grupos en distintas partes del mundo. Todos los nodos estan conectados entre si y tienen un líder para poder organizarse. Supongamos que falla el líder

a) Proponga un algoritmo para elección de un nuevo líder, basándose en el menor ID.
b) Suponga que uno de los grupos queda desconectado de todos los demás, formandose una partición de la red. Después de un cierto tiempo, el grupo se puede volverse a conectar. Proponga un protocolo de elección de líder que contemple esta situación. Cada partición que se provoque tiene que elegir un nuevo líder y seguir funcionando bajo sus órdenes. En caso que se vuelvan a unir las particiones, deberan buscar un nuevo líder.

Ejercicio 6 ★

Se tiene un sistema distribuido conectado con una topología en malla. Proponga algún protocolo de elección de líder bajo esta topología.



Ejercicio 7 ★

Considere el siguiente algoritmo de consenso para sistemas distribuidos, donde todos los nodos están conectados entre sí:

Paso 1: Cuando un nodo se da cuenta de que el líder se ha bloqueado, envía un mensaje "ELECCIÓN" a todos los nodos que tienen id superior al suyo. En caso que no reciba respuestas de los nodos superiores, asume que es el líder y envía el mensaje "COORDINADOR", junto con su id a todos los nodos, incluidos los nodos con id inferiores.

Paso 2: Cuando un nodo con un id superior recibe un mensaje de "ELECCIÓN" de un nodo inferior, responde con su id de nodo junto con la respuesta "OK".

Paso 3: Cuando un nodo recibe la respuesta "OK" de muchos nodos (con sus respectivos ids), encuentra el id más alto, max(id), y envía mensajes "COORDINADOR" junto con el maximo id de todos los nodos. De esta forma se anuncia al nuevo líder.

Para asegurarse de que el líder recién elegido no se haya bloqueado, el nodo que envía el mensaje "COORDINADOR" junto con el id, espera un tiempo aleatorio para recibir la respuesta "ACEPTAR"

del nuevo líder. Si no recibe respuesta de "ACEPTAR", el nodo que envía el mensaje "COORDINADOR" inicia de nuevo todos los pasos. Estos pasos continúan hasta que un nuevo líder responde con la respuesta "ACEPTAR" o el nodo sigue el **Paso 1** donde anuncia al nuevo líder.

Se pide:

1. Indicar si se logra elegir un líder al usar este algoritmo. Justifique su respuesta .
2. Indicar a cuál de los algoritmos vistos en clase se asemeja más, indicando sus ventajas y/o desventajas en comparación al algoritmo seleccionado. Justifique su respuesta.

Ejercicio 8 ★

En una variante descentralizada del protocolo Two Phase Commit, los participantes se comunican directamente uno con otro en vez de indirectamente con un coordinador

- Fase 1: El coordinador manda su voto a todos los participantes
- Fase 2: Si el coordinador vota que no, los participantes abortan su transacción. Si vota que si, cada participante manda su voto al coordinador y al resto de participantes donde cada uno decide sobre el resultado acorde a el voto que le llega y lleva a cabo el procedimiento. Es decir, si llega un solo abort, deben cancelar su ejecución. En cambio, tiene que llegar la confirmación de todos los demás para poder continuar.

1. ¿Qué ventajas y desventajas encuentra esta variante con respecto a la variante centralizada? Hablar con respecto a la cantidad de mensajes, tolerancia a fallos, etc
2. ¿En qué casos usaría cada versión del protocolo?

Ejercicio 9

Un servidor usa ordenamiento por timestamps para el control de la concurrencia entre sistemas distribuidos. Este servidor funciona para que una app bancaria permita realizar operaciones de manera remota. Entre otras cosas, comprar y vender diferentes tipos de divisas. Un usuario puede comprar cualquier tipo de moneda dentro del sistema con cualquiera que posea en su cuenta. El banco cuenta con un sistema que recibe los pedidos, que a su vez se puede comunicar con otro sistema que controla el stock de los diferentes tipos de divisas y un tercer sistema que mantiene registro del estado de cuenta de los usuarios.

a) Describir un protocolo que permita a los usuarios comprar divisas manteniendo en todo momento la consistencia, aprovechando el uso de los timestamps.
b) Una persona nos planteó el uso del protocolo 2PC para resolver este problema. Sin embargo, alguien nos argumentó que el uso de este protocolo es redundante bajo estas condiciones. ¿Estaría de acuerdo con esta persona? Justifique