Examen 1

Ricardo Montiel Manriquez

28 de Octubre del 2021

1. Preguntas de las tareas:

1. Da un algoritmo distribuido para contar el número de vértices en un árbol enraízado T, iniciando en la raíz.

   Nos basaremos en el cómputo por agregación y en el algoritmo de BroadConvergeCastTree que como revisamos en clase, es un algoritmo que nos ayuda a recorrer el árbol iniciando desde la raíz hasta las hojas, posteriormente confirmarmos la información estando desde las hojas recorrer todo el árbol nuevamente pero comenzando desde las hojas y terminando en la raíz, que termina recibiendo la información de que el recorrido se ha completado además agregaremos a este algoritmo una función de agregación $f(x, y)$ que es la funcion que regresa como resultado $x + y$, Asi:

   Algoritmo broadConvergecastTree

```
BroadConvergecastTree(ID, soyRaiz):

PADRE, HIJOS, #noVecientos = 0, acc = 1

Ejecutar inicialmente: // tiempo cero
    Si soyRaiz entonces:
        send(<START>) a todos en HIJOS

    Al recibir <START> de PADRE:
        Si |HIJOS| ≠ 0 entonces:
            send(<START>) a todos en HIJOS
        Sino
            send(<ok, acc>) a PADRE

Al recibir <ok, accum> de algun puerto en HIJOS:
    #noVecinos++
    acc  = f(acc, accum)
    Si #noVecinos == |HIJOS| entonces:
        Si soyRaiz entonces:
            return acc
        Sino
            send(<ok, acc>) a PADRE

```

   Al recibir $<ok, accum>$ notemos que la variable accum que se pasa como argumento, es un dato de tipo Int que posteriormente podemos ver que llevará guardado el número de los vértices que se van acumulando de abajo hacia arriba, al final la raíz tendrá la cuenta de los vértices que tiene como hijos y se agregara a él mismo a la cuenta para posteriormente terminar el algoritmo regresando un número entero que representa la cantidad de vértices en el árbol.

2. Generaliza el algoritmo convergecast (figura 1), para recolectar toda la información del sistema. Esto es, cuando el algoritmo termine, la raíz debería de tener todas las entradas de todos los procesos.

Algoritmo Convergecast(ID):

```
1
2       PADRE, HIJOS, soyRaiz = soyRaiz, #recibidos = 0
3           list [] informacion
4
5           informacion=[ID]
6
7           Ejecutar inicialmente: // tiempo cero
8               Si |HIJOS| == 0 entonces:
9               send(<ok,informacion>) a PADRE
10
11          Al recibir <ok,info> de algun puerto en HIJOS:
12              #recibidos++
13              informacion = f(informacion,info)
14              Si #recibidos == |HIJOS| entonces:
15              send(<ok,informacion>) a PADRE
16
```

donde f = función de concatenación de listas:

$f(l1, l2)$ que regresa una lista, resultado de la concatenación de dos listas en el mismo orden,ejemplo: $f([1, 2].[3, 4]) = [1, 2, 3, 4]$

Al final del algoritmo la raiz tendra una lista con los ID de todos los elementos icluida ella, por lo tanto dicha lista contendra todos la informacion: Todos los vertices del arbol T.

Podemos ver que se envían mensajes desde las hojas hacia la raíz todos los nodos envían un mensaje a su padre menos la raíz pues el no tiene padre, entonces la cantidad de mensajes que se envían es de $|v| - 1$ y si suponemos que cada mensaje toma kbits en enviarse, eso quiere decir que $|v|k - k$ seria la cantidad de mensajes si tomaran k-bits.

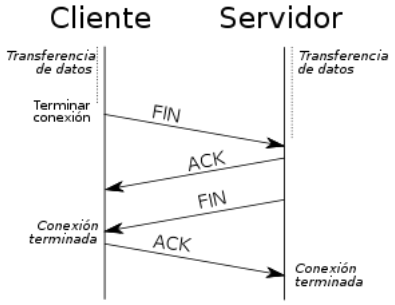
5. Describe brevemente el protocolo TCP. ¿Es posible resolver el problema de los dos amantes si hay un canal TCP confiable entre ambos amantes?

El *Protocolo de Control de Transmisión* o TCP (por sus siglas en inglés) es un sistema reglamentado que permite la comunicación en tres fases: el establecimiento de una conexión, la transferencia de los datos y el fin de la conexión. En particular, TCP destaca por poder transferir los datos de forma ordenada, re-transmitir paquetes perdidos o con errores y, el control del flujo y la congestión de los datos.

Lo relevante para el problema de los dos amantes, sin embargo, es la fase que finaliza la conexión, la cual consiste cuatro pasos:

a) La máquina **A** envia el mensaje para finalizar la conexión a la máquina **B**.
b) **B** envía un mensaje de confirmación al recibir el anterior.
c) **B** envía el mensaje final de que ha cerrado la conexión.
d) **A** cierra la conexión.

Es decir que para terminar una conexión esta es la información que se intercambia:



Entonces si tenemos a los enamorados Alicia y Bob, Alicia puede proponer una hora para el encuentro, luego Bob indica que ha leído su mensaje, confirma la hora, Alicia indica que ha leído su confirmación y se diregen a la cita. El problema dice explícitamente que suponemos que hay un canal TCP confiable entre ambos amantes, si esto garantiza que los mensajes son recibidos correctamente siempre, entonces el escenario anterior esta garantizado y el problema ha sido resuelto, en caso contrario siempre existe la posibilidad de que alguno de los mensajes se pierda (por ejemplo, en el que Bob cofirma la hora) y ambos no se puedan coordinar correctamente; en el caso del TCP esto ocurre cuando se deja un canal semi-cerrado.

2. Preguntas nuevas:

2. ¿Qué es sincronización? y ¿Por qué se dice que la sincronización es difícil?

   La sincronización es cuando buscamos que diferentes procesos dentro de nuestro sistema distribuido con diferentes maquinas se ejecuten de forma cronologica y respetando el orden de los eventos que ocurren dentro del sistema, haciendo diferencias incluso por milesimas de segundo.

3. ¿A que nos referimos cuando hablamos de un algoritmo distribuido?

   Es un algoritmo que es utilizado en el software de la computadora que consta de varios procesadores que estan interconectados y se tiene la responsabilidad de ejecutar diferentes partes del algoritmo entre estos procesadores al mismo tiempo.

   La sincronnzacion resulta compleja debido a que la información y el procesamiento se mantienen en diferentes nodos y la comunicacin estre estos su tiempo y si la red es muy grande la comunicacion puede ser muy tardada.

4. ¿A que nos referimos cuando hablamos del problema de indistinguibilidad en un sistema distribuido?

   A que no sabemos cuando el mensaje que fue enviado fue recibido con exito. Pudo suceder que se perdio en el camino y este nunca llego pero nunca lo sabremos.