Examen 2 - Computación Distribuida

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE CIENCIAS

2026-1

Nombre: Carbajal Galicia Hilda Joana No. de cuenta 318087223

EJERCICIO 1. Explica por qué el algoritmo para construir un árbol generador puede producir más de un árbol (Algoritmo 5).

El algoritmo 5 puede producir más de un árbol diferente porque no existe un límite para el tiempo de entrega de los mensajes, lo que hace que el orden de llegada de los mensajes `GO()` pueda variar haciendo que se generen diferentes árboles en distintas ejecuciones.

EJERCICIO 2. En Monterrey, hay varias estaciones de monitoreo climático con software para recopilar datos meteorológicos. La estación central en la zona metropolitana necesita recabar los datos de todas las estaciones periféricas.

Escribe un algoritmo distribuido que permita a cada estación recopilar información de sus estaciones hijas y enviarla a la estación padre sin duplicados, y que la estación central reporte los datos meteorológicos de todas las estaciones.

Me basé en el algoritmo convergecast

Algorithm 1

```
1:  Initially do
2:   parent_i = estación central
3:   children_i = esatciones hijas
4:   data_i = datos_meteorológicos
5:   reports = todos los reportes
6:   expected_responses = |children_i|  // nº de estaciones perifericas
7: Si p_i es hoja (children_i vacío):      hijas que deben responder
8:   Enviar BACK(data_i) a parent_i
```

```
 9: Cuando reciba BACK(dataset) de un hijo p_j:
 10. received_data.append(dataset)  // Almacena los datos

 11:   expected_responses = expected_responses - 1
 12:  end if
 13: Si expected_responses == 0:
 14:   all=reports + data_i
 15:   Si p_i != 0:
 16:     Enviar BACK(all) a parent_i
 17:   else:
 18:     Reportar all
 19:  end if
 10: end
```

EJERCICIO 3. Explica con tus propias palabras el funcionamiento del siguiente algoritmo:

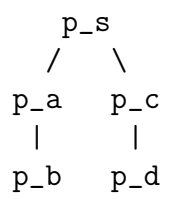
```
Algorithm 1 Algoritmo de Suma Distribuida - Código para el proceso pi

1: valori = valor local del proceso pi
2: vecinosi = conjunto de procesos vecinos de pi
3: sumai = valori   Suma parcial, inicializada con el valor local
4: conti = 0        Contador de mensajes recibidos
5: begin:
6:   if pi == ps then
7:     for each j  vecinosi do
8:       send SOLICITUD to pi
9:     end
10: when SOLICITUD is received from pi
11:   begin:
12:     send RESPUESTA(valori) to pi
13:   end
14: when RESPUESTA(valor) is received from pi
15:   begin:
16:     sumai = sumai + valor
17:     conti = conti + 1
18:     if conti == |vecinosi| then
19:       resultado = sumai   La suma total ha sido calculada
20: end
```

El algoritmo inicia con p_s enviando una solicitud a los vecinos, cuando estos la reciben responden con su valor y suma el mensaje, cuando todos ya respondieron termina despúes de calcular el resultado.

1. ¿Para qué tipo de gráficas está diseñado el algoritmo?
Me parece que los nodos son los que envían y reciben información a la raíz, asi que diría para gráficas que son árboles generadores enraizados.

2.¿Cómo es la topología de la gráfica? Da un ejemplo si es necesario y justifica.

No tiene ciclos por ser un árbol, tiene una raíz, y hay comunicación entre los vecinos (hijos) y padres. Como ejemplo en este árbol los procesos hoja (p_c, p_b, p_c, pd) envían al padre los valores sumados y hacia arriba.



3. ¿Cuántas rondas son necesarias para ejecutar el algoritmo? Justifica.
EL algoritmo necesita 3 rondas para ejecutarse, en la primera ronda la raíz p_s envía una solicitud a los vecinos, en la segunda ronda cada vecino recibe una solicitud y la responde, en la tercera ronda la raíz recibe la respuesta y actualiza la suma, cuando todas las respuestas son iguales al numero total de vecinos calcula la suma total.