Universidad Nacional Autónoma de México

Facultad de Ciencias

Computación Distribuida

Tarea 1

Johann Ramón Gordillo Guzmán
418046090

Tarea presentada como parte del curso de **Computación Distribuida** impartido por la profesora **M.C Karla Rocío Vargas Godoy**.

05 de Octubre del 2020

Link al código fuente: https://github.com/JohannGordillo/

Actividades

1. ¿Cuál es la diferencia entre el cómputo paralelo y el cómputo distribuido?

   Respuesta.

   Como hemos visto a lo largo de estas dos semanas, en el **Cómputo Distribuido** tenemos múltiples entidades (también llamadas procesos, procesadores o computadoras) autónomas tal que cada una de ellas tiene solo conocimiento parcial de los parámetros involucrados en el problema a ser resuelto. Dichas entidades se comunican unas con otras por medio del paso de mensajes y trabajan en conjunto para la resolución de un único problema.
   Una dificultad de este enfoque es que una entidad no puede saber el estado actual de otra de manera instántanea, sino que únicamente sus estados pasados.

   Por otro lado, en el **Cómputo Paralelo** cada una de las entidades realiza múltiples tareas asignadas a ellas de manera simultánea.

   De esta manera, la **principal diferencia** entre el cómputo distribuido y el cómputo paralelo es que éste último permite que múltiples entidades (procesadores) ejecuten tareas de manera simultánea, mientras que en el cómputo distribuido se divide una única tarea entre varias entidades para resolver un problema en común.

2. ¿Por qué no hay un **único** modelo de cómputo distribuido?

   Respuesta.

   Dentro del **Cómputo Distribuido** tenemos distintos modelos, generalmente divididos en los siguientes grupos: Modelo de Paso de Mensajes, Modelo de Memoria Compartida, y otros modelos como los de Autoestabilización, de Robots Móviles y protocolos de poblamiento.

   No hay un único modelo de cómputo, pues cada uno de ellos provee ventajas y desventajas, además de que algunos son especializados para resolver cierto tipo de problemas computacionales. A continuación explico algunos de los modelos con base en [2]:

   En el **Modelo de Paso de Mensajes** se simulan redes y cualquier interacción entre procesadores físicamente separados requiere la transmisión de información de un lugar a otro.

   Los **Modelos de Memoria Copartida** describen el comportamiento de los procesos en un sistema multiprocesamiento. La comunicación se da a través de estructuras de datos compartidas.

   Un sistema distribuido es de **Autoestibilización** si dado un estado inicial, el sistema terminará en un estado correcto en un número finito de pasos.

   En el **Modelo de Robots Móviles**, los agentes (robots) están representados como puntos en $\mathbb{R}^2$ y no tienen habilidad para comunicarse, pero pueden observar las posiciones de los demás.

3. Explica con tus propias palabras qué es un *evento*.

Respuesta.

Para mi, un **evento** no es más que una acción desencadenada por las entidades de un programa o por un usuario, la cual es detectada por el programa. Algunos ejemplos de eventos son enviar y recibir mensajes.

Dentro del Algoritmo de Reconocimiento de una Gráfica, cuando un proceso envía a otro su conjunto de vecinos, estamos frente a un evento.

4. Diseña un algoritmo distribuido que calcule la distancia entre la raíz de una gráfica y el nodo que se está visitando.

Respuesta.

Algorithm 1 Broadcast para calcular la distancia entre p_i y la raíz de la gráfica

1: $neighbors_i = \{$conjunto de vecinos$\}$
2: $seen\_message_i = false$
3: $distance_i = 0$
4: **if** $p_i = root$ **then**
5: $\quad seen\_message_i = true$
6: $\quad$ **for** $j \in neighbors_i$ **do**
7: $\quad\quad$ `send` MYDISTANCE($distance_i$) to j
8: $\quad$ **end for**
9: **end if**

10: **when** MYDISTANCE($distance_j$) is received from neighbor p_j
11: **begin:**
12: $\quad$ **if** $seen\_message_i = false$ **then**
13: $\quad\quad seen\_message_i = true$
14: $\quad\quad distance_i = distance_j + 1$
15: $\quad\quad$ **for** $k \in neighbors_i \backslash \{j\}$ **do**
16: $\quad\quad\quad$ `send` MYDISTANCE($distance_i$) to k
17: $\quad\quad$ **end for**
18: $\quad$ **end if**
19: **end**

En esta implementación cada proceso p_i tendrá su distancia al nodo raíz en $distance_i$, inicialmente en cero, y los procesos que sean vecinos del nodo raíz *root* incrementarán esta distancia en 1 durante la primera ronda. Posteriormente, enviarán esta distancia a sus vecinos, que también incrementarán su contador en 1, y así sucesivamente hasta que todos los procesos conozcan su distancia al nodo raíz.

Pude haber llamado al nodo raíz como p_s ya que es un proceso distinguido, pero decidí llamarlo *root*. El algoritmo no es eficiente, pues se están enviando mensajes de más. Para solucionar este problema, podemos usar un broadcast basado en un árbol generador con raíz, pero esto lo implementaremos esta semana.

5. Para el algoritmo propuesto en el ejercicio anterior, argumenta cuántos mensajes es necesario enviar para diseminar el mensaje y cuál es el tamaño de cada mensaje enviado.

   Respuesta.

   - **Número de mensajes.**

     En el peor de los casos, un mensaje se enviará por el mismo canal dos veces, a excepción del canal que conecta al nodo raíz, pues por este canal el mensaje se enviará una única vez, ya que en esta implementación un proceso no reenvía el mensaje al proceso del que lo recibió y en la primera ronda el único proceso que envía mensajes es la raíz.

     Entonces, el nodo raíz v_r envía un total de $\delta(v_r)$ mensajes durante la primera ronda, por lo que en el peor de los casos se enviarán un total de $2e - \delta(v_r)$, donde e es el tamaño de la gráfica y $\delta(v_r)$ el grado del nodo raíz.

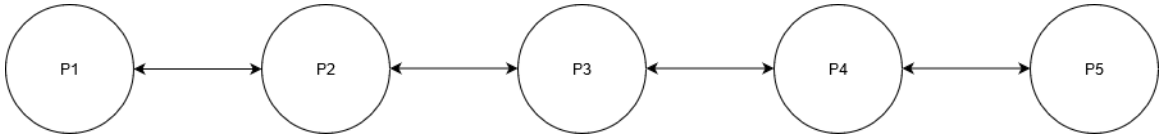
     $\therefore$ La cota superior es $2e - \delta(v_r)$.

   - **Tamaño de los mensajes.**

     Cada mensaje MYDISTANCE($distance_i$) enviado por un proceso p_i tiene un entero que es máximo n, y sabemos que se puede codificar con $\log n$ bits, de manera que el tamaño máximo de los mensajes será $\log n$ bits.

     $\therefore$ La cota superior es $\log n$ bits.

6. **(Punto Extra)** Realizar la ejecución detallada de la ejecución del algoritmo para aprender la gráfica de comunicación de la siguiente imagen:



Respuesta.

En la **inicialización** tenemos los siguientes valores para cada uno de los procesos:

$vecinos_1 = \{2\}$
$proc\_conocidos_1 = \{1\}$
$canales\_conocidos_1 = \{< 1, 2 >\}$

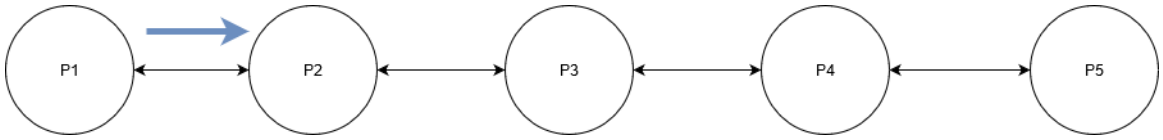
$vecinos_2 = \{1, 3\}$
$proc\_conocidos_2 = \{2\}$
$canales\_conocidos_2 = \{< 2, 1 >, < 2, 3 >\}$

$vecinos_3 = \{2, 4\}$
$proc\_conocidos_3 = \{3\}$
$canales\_conocidos_3 = \{< 3, 2 >, < 3, 4 >\}$

$vecinos_4 = \{3, 5\}$
$proc\_conocidos_4 = \{4\}$
$canales\_conocidos_4 = \{< 4, 3 >, < 4, 5 >\}$

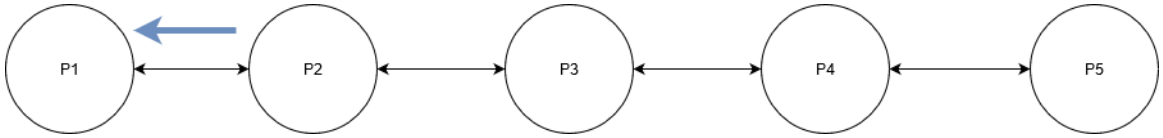
$vecinos_5 = \{4\}$
$proc\_conocidos_5 = \{5\}$
$canales\_conocidos_5 = \{< 5, 4 >\}$

Avanzamos a la **Ronda 1**. p_1 envía a p_2 su información y actualiza sus valores.



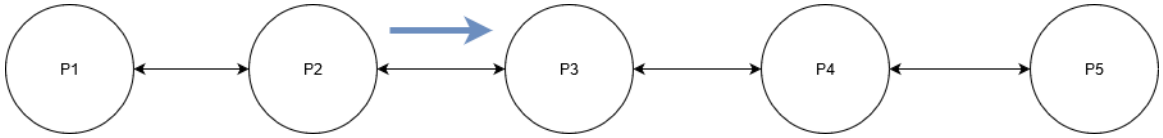
$vecinos_2 = \{1, 3\}$
$proc\_conocidos_2 = \{2, 1\}$
$canales\_conocidos_2 = \{< 2, 1 >, < 2, 3 >\}$

p_2 envía a p_1 su información y actualiza sus valores.



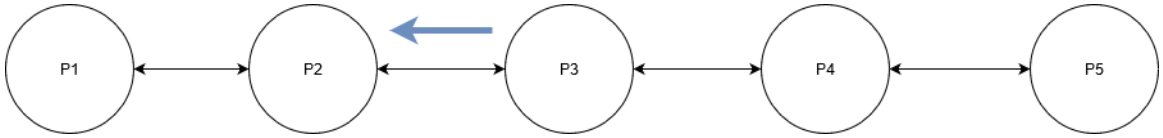
$vecinos_1 = \{2\}$
$proc\_conocidos_1 = \{1, 2\}$
$canales\_conocidos_1 = \{< 1, 2 >, < 2, 3 >\}$

p_2 envía a p_3 su información y actualiza sus valores.



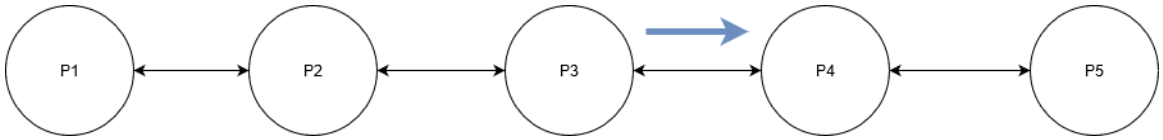
$vecinos_3 = \{2, 4\}$
$proc\_conocidos_3 = \{3, 2\}$
$canales\_conocidos_3 = \{< 3, 2 >, < 3, 4 >, < 2, 1 >\}$

p_3 envía a p_2 su información y actualiza sus valores.



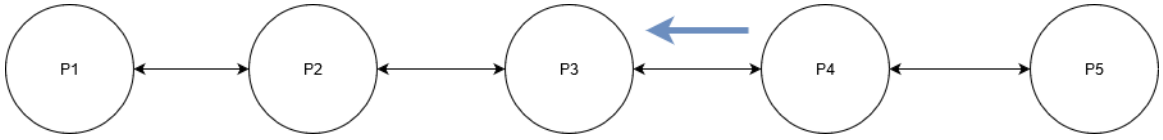
$vecinos_2 = \{1, 3\}$
$proc\_conocidos_2 = \{2, 1, 3\}$
$canales\_conocidos_2 = \{< 2, 1 >, < 2, 3 >, < 3, 4 >\}$

p_3 envía a p_4 su información y actualiza sus valores.



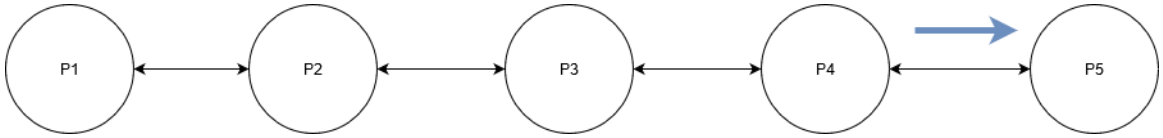
$vecinos_4 = \{3, 5\}$
$proc\_conocidos_4 = \{4, 3\}$
$canales\_conocidos_4 = \{< 4, 3 >, < 4, 5 >, < 3, 2 >\}$

p_4 envía a p_3 su información y actualiza sus valores.



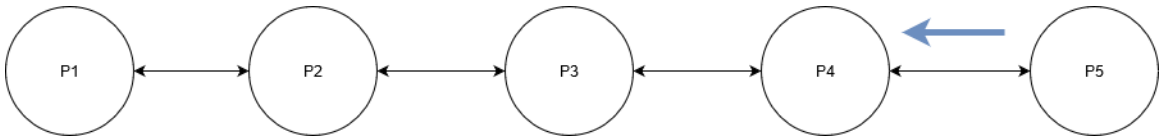
$vecinos_3 = \{2, 4\}$
$proc\_conocidos_3 = \{3, 2, 4\}$
$canales\_conocidos_3 = \{< 3, 2 >, < 3, 4 >, < 2, 1 >, < 4, 5 >\}$

p_4 envía a p_5 su información y actualiza sus valores.



$vecinos_5 = \{4\}$
$proc\_conocidos_5 = \{5, 4\}$
$canales\_conocidos_5 = \{< 5, 4 >, < 4, 3 >\}$

p_5 envía a p_4 su información y actualiza sus valores.

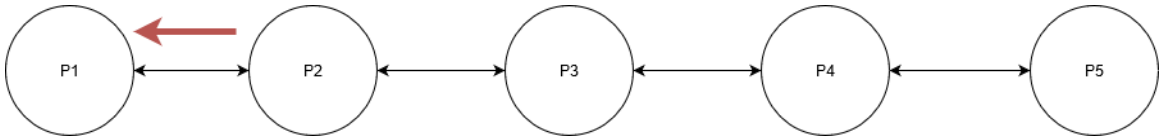


$vecinos_4 = \{3, 5\}$
$proc\_conocidos_4 = \{4, 3, 5\}$
$canales\_conocidos_4 = \{< 4, 3 >, < 4, 5 >, < 3, 2 >\}$

Fin de la **Ronda 1**. Comienza la **Ronda 2**.

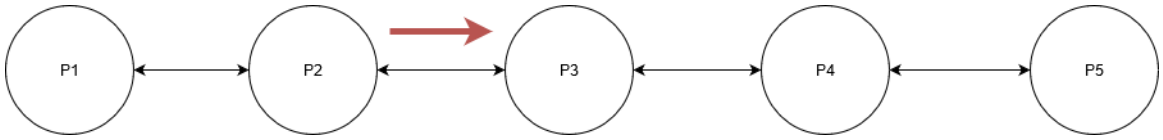
Hasta este punto, p_1 y p_5 conocen 2 canales; $p2$ y p_4, 3 canales; y p_3 conoce todos los canales.

p_2 hace el reenvío de información a p_1 y actualiza sus valores.



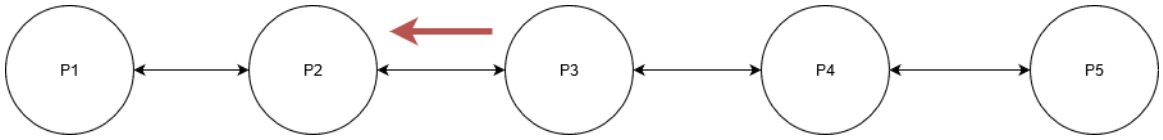
$vecinos_1 = \{2\}$
$proc\_conocidos_1 = \{1, 2, 3\}$
$canales\_conocidos_1 = \{< 1, 2 >, < 2, 3 >, < 3, 4 >\}$

p_2 hace el reenvío de información a p_3 y actualiza sus valores.



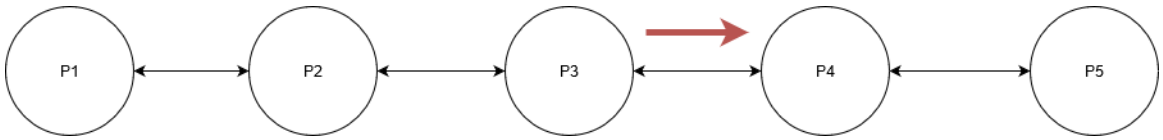
$vecinos_3 = \{2, 4\}$
$proc\_conocidos_3 = \{3, 2, 4, 1\}$
$canales\_conocidos_3 = \{< 3, 2 >, < 3, 4 >, < 2, 1 >, < 4, 5 >\}$

p_3 hace el reenvío de información a p_2 y actualiza sus valores.



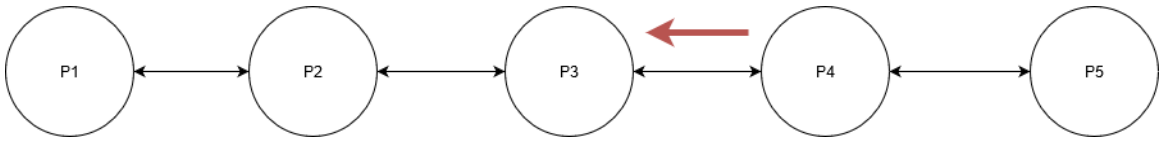
$vecinos_2 = \{1, 3\}$
$proc\_conocidos_2 = \{2, 1, 3, 4\}$
$canales\_conocidos_2 = \{< 2, 1 >, < 2, 3 >, < 3, 4 >, < 4, 5 >\}$

p_3 hace el reenvío de información a p_4 y actualiza sus valores.



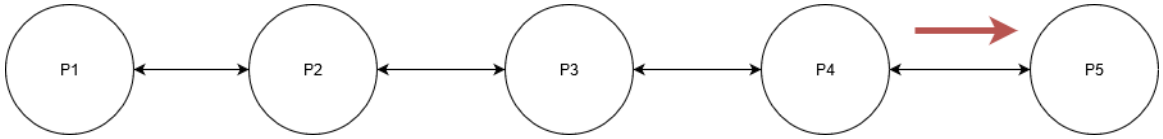
$vecinos_4 = \{3, 5\}$
$proc\_conocidos_4 = \{4, 3, 5, 2\}$
$canales\_conocidos_4 = \{< 4, 3 >, < 4, 5 >, < 3, 2 >, < 2, 1 >\}$

p_4 hace el reenvío de información a p_3 y actualiza sus valores.



$vecinos_3 = \{2, 4\}$
$proc\_conocidos_3 = \{3, 2, 4, 1, 5\}$
$canales\_conocidos_3 = \{< 3, 2 >, < 3, 4 >, < 2, 1 >, < 4, 5 >\}$

p_4 hace el reenvío de información a p_5 y actualiza sus valores.



$vecinos_5 = \{4\}$
$proc\_conocidos_5 = \{5, 4, 3\}$
$canales\_conocidos_5 = \{< 5, 3 >, < 4, 3 >, < 3, 2 >\}$

Tras un par de rondas más, p_5 y p_1 se conocerán mutuamente, pues toma d rondas a dos procesos p_i y p_j conocerse, donde d es la distancia entre p_i y p_j.

Finalmente, los valores de los procesos quedan como:

$vecinos_1 = \{2\}$
$proc\_conocidos_1 = \{1, 2, 3, 4, 5\}$
$canales\_conocidos_1 = \{< 1, 2 >, < 2, 3 >, < 3, 4 >, < 4, 5 >\}$

$vecinos_2 = \{1, 3\}$
$proc\_conocidos_2 = \{1, 2, 3, 4, 5\}$
$canales\_conocidos_2 = \{< 1, 2 >, < 2, 3 >, < 3, 4 >, < 4, 5 >\}$

$vecinos_3 = \{2, 4\}$
$proc\_conocidos_3 = \{1, 2, 3, 4, 5\}$
$canales\_conocidos_3 = \{< 1, 2 >, < 2, 3 >, < 3, 4 >, < 4, 5 >\}$

$vecinos_4 = \{3, 5\}$
$proc\_conocidos_4 = \{1, 2, 3, 4, 5\}$
$canales\_conocidos_4 = \{< 1, 2 >, < 2, 3 >, < 3, 4 >, < 4, 5 >\}$

$vecinos_5 = \{4\}$
$proc\_conocidos_5 = \{1, 2, 3, 4, 5\}$
$canales\_conocidos_5 = \{< 1, 2 >, < 2, 3 >, < 3, 4 >, < 4, 5 >\}$

Y el algoritmo para descubrimiento de la gráfica habrá finalizado.

Cabe añadir que la cota superior para el número de mensajes de este algoritmo es $2ne$ donde e es el número de canales y n el número de procesos. Además, la cota para el tamaño de los mensajes es $(g + 1)(\log n)$, donde g es el grado máximo de la gráfica.

Referencias

[1] Raynal, M. *Distributed Algorithms for Message-Passing Systems.* Springer, 2013.

[2] Aspnes, J. *Notes on Theory of Distributed Systems.* Yale University, 2017.

[3] Kshemkalyani, A. & Singhal, M. *Distributed Computing: Principles, Algorithms, and Systems.* Cambridge University Press, 2008.