Informe algoritmo pagerank distribuido y centralizado

Importante:
En los archivos de los grafos quitar las líneas con #, ya que nosotros los leemos como si estas no existirán.

Modelo centralizado:

Para el modelo centralizado se escogieron las siguientes estructuras; 2 diccionarios tipo :

- **map<int,vector<int>> nodos** : El cual contiene como primer elemento (llave) los nodos del grafo, y de segundo, los nodos a los cuales apunta (vector).
- **map<int,vector<int>> padres** : En su llave, van todos los nodos del grafo, y el valor (vector), contiene todos los nodos padre (nodos que apuntan a la llave) del nodo actual.

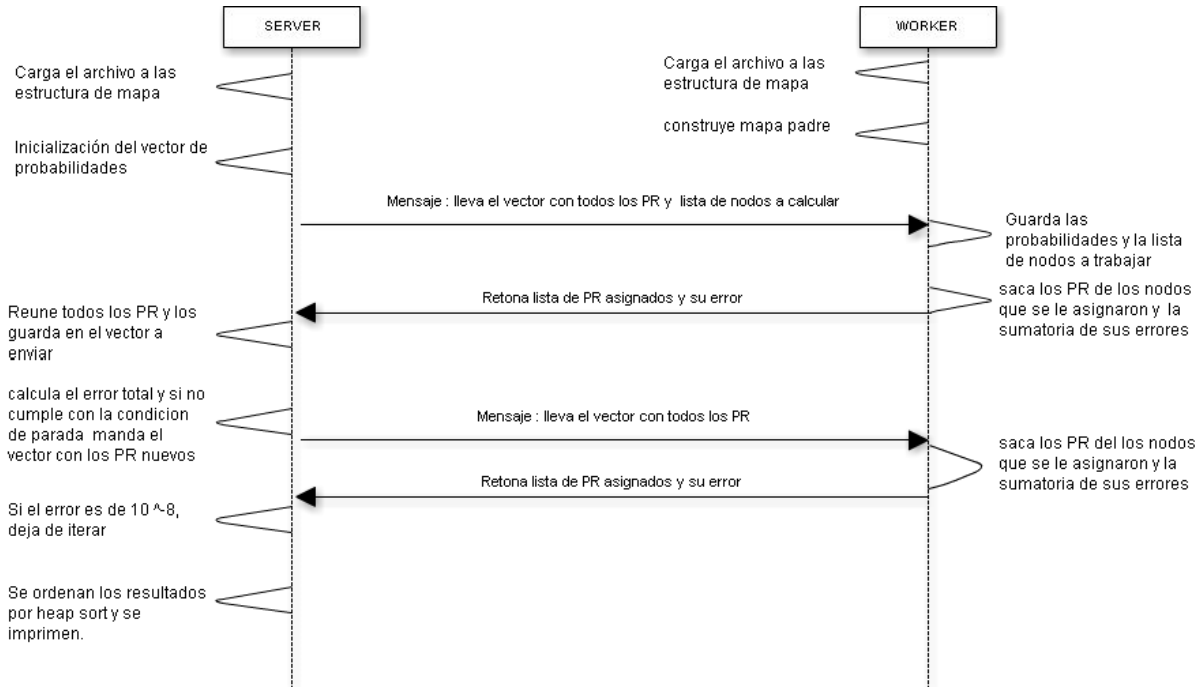
El almacenamiento de los PR se se hace en otros dos mapas diferentes, uno que se inicializa con los primeros valores de PR para que arranque el algoritmo el cual es **map<int,double> probabilidades**; y el segundo es **map<int,double> probabilidades2** que contiene los nuevos PR calculados durante la iteración, luego probabilidades2 es vaciado en **probabilidades**
y continua el algoritmo hasta llegar al criterio de parada que es de 1x10^-8. Esta condicion de parada es la sumatoria de los errores en cada nodo, el cual es la resta del nuevo PG con el antiguo.

Al ejecutar el algoritmo para reducir la complejidad se hace una intersección de los dos conjunto (Los dos mapas o diccionarios) y esto da origen al nuevo vector de probabilidades (en cada iteración se suman las probabilidades para comprobar que su valor sea 1 o cercano a este).

Por último y algo importantes para la convergencia del algoritmo es que al cargar el archivo en el mapa nodos recorremos este llenando las cabezas vacías con el ID de todos los demás nodos excepto el mismo, esto con el fin de evitar que estos nodos se conviertan en una especie de sumidero de la probabilidad de los demás (PR).

Modelo distribuido:

Diagrama de secuencia para nuestra aplicación distribuida:



Teniendo en cuenta nuestro modelo centralizado se decide distribuir los proceso de la siguiente manera:

server:

- El servidor es el encargado de repartir el grafo y su carga de trabajo entre los workers suscritos a su puerto d (para esto se debe ejecutar los workers primero y todo los que se vayan a utilizar). esto se hace solo la primera vez únicamente y en las iteraciones posteriores solo se envia la lista de probabilidades

- El no saca la lista de padres, pero a las cabezas nulas las corrige para que funcione bien el algoritmo.
- Luego de repartir el trabajo se espera a que lleguen todas las particiones con los PR nuevos para proceder a verificar si se cumplio la condicion de parada, si no se cumple envía los nuevos pr para la siguiente iteración.
- Cuando la condición de parada se da, el servidor ordena y muestra los nodos según su PR.

worker: diferencia del server recibe la dirección a suscribirse y dirección a enviar resultados: estas están por defecto y deben ser estas (obviamente cuando es en máquinas distantes cambia la ip) ./worker localhost:5220 localhost:5450.

- Debe tener el archivo con los nodos para la carga ya que este genera su propio mapa para poder trabajar sobre la lista que le manda el servidor. esta lista no es más que el id de las cabeza del diccionario. esto se ejecuta cuando se inicia el worker.
- crea el mapa de padres para a partir del de nodos
- Espera a que haya un mensaje en la pila del puerto y carga la información de este llenado el diccionario de PR para la iteración.
- hace el cruce entre el diccionario de padres y la lista de nodos enviada para hallar el pr nuevo.
- luego de esto lo envía de regreso al servidor

por razones de tiempo no se implementó un sistema de archivo compartido entre los workers pero sabemos de que esto hubiera sido mucho mejor que tener el archivo en todos lo workers.

TIEMPOS:

Grafo	**Modelo distribuido:**	**Modelo centralizado:**
Node 7115	2:34 minutos	4:15 minutos