Comparación de sistema centralizado y distribuido para la distribución de libros en un sistema bibliotecario

Leonardo Sobarzo 201573584-0
Leandro Cerna 201573552-2

3 de diciembre de 2020

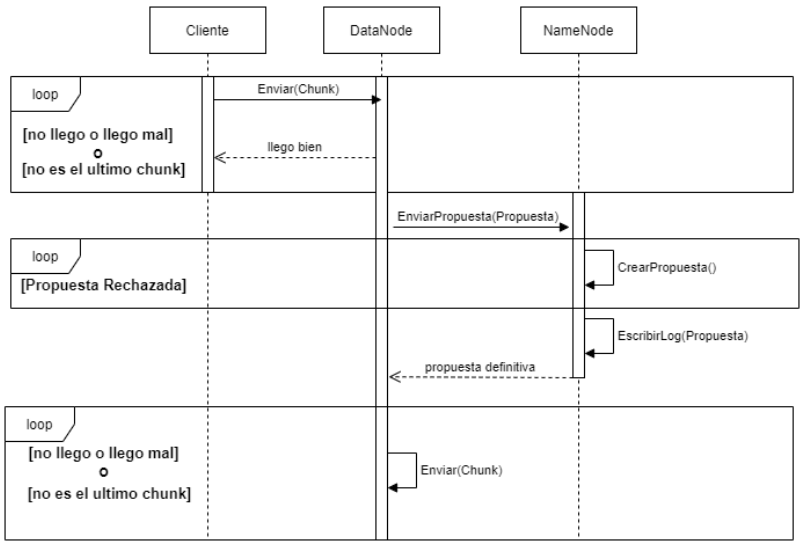
Que se hizo

Para la realización del experimento se utilizó el lenguaje de programación GO, con sus módulos de gRPC y protocolbuffer.

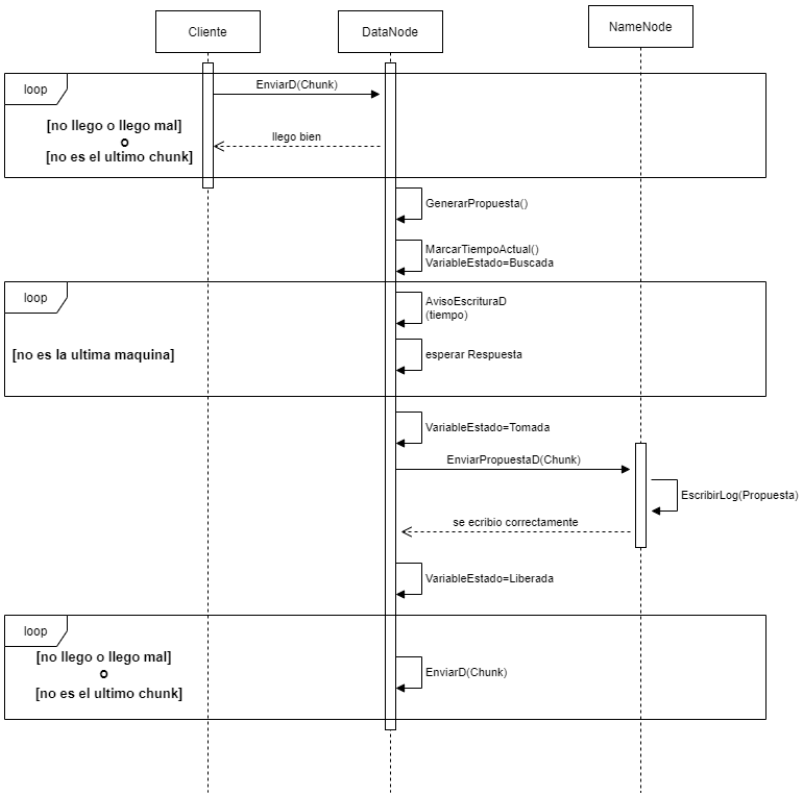
Se utilizó cuatro máquinas virtuales, que realizaron diferentes tareas dentro del sistema. Una máquina funcionó como guía (Namenode) en donde se guarda en memoria y en disco un registro de los libros que posee el sistema, además de donde encontrar sus partes que están almacenadas en otras máquinas, este trabajo se le asignó a la máquina llamada *dist41*. Las tres máquinas restantes fueron utilizadas como almacenamiento (Datanode) de partes de libros, como aclaración estas son: *dist42, dist43* y *dist44*. Finalmente el cliente, que puede subir o bajar libros del sistema, se ejecuta en cualquiera de las cuatro máquinas antes mencionadas.

En el **algoritmo centralizado** el cliente al subir un libro primero lo debe partir en trozos de 250kB y envía cada uno de ellos a la máquina *dist43*. Este Datanode al obtener todos las partes del archivo genera una distribución de ellos y se la envía al Namenode para su revisión. La distribución sigue la planificación de Round-robin, es decir se entrega trozos de manera equitativa y racional a los Datanode comenzando por la máquina *dist42*, siguiento *dist43*, prosigue *dist44* y se vuelve a comenzar con la máquina *dist42*.

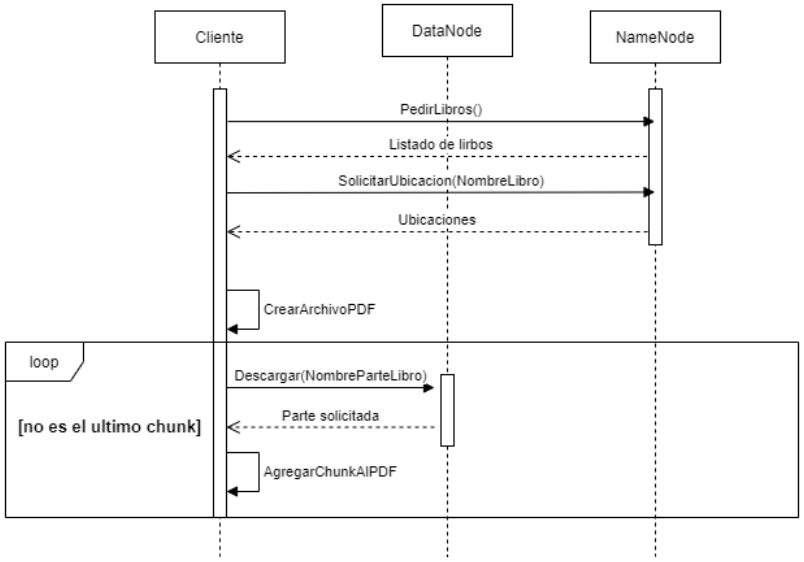
Luego de que *dist41* obtiene la distribución genera un número aleatorio, si este es mayor a 0.1 entonces la distribución es aceptada, sino se genera una distribución al azar con los tres Datanode siendo tomados en cuenta. Después se guarda en memoria el nombre del libro y la distribución de las partes, además de escribir en un archivo llamado Log.txt, el cual posee el nombre del libro, sus partes y la distribución de estas en las máquinas. Finalmente se envía la distribución aceptada a *dist43* para que ella distribuya las partes del libro a las máquinas respectivas.



En el **algoritmo distribuido** el cliente también debe particionar el archivo y enviar sus trozos a *dist43* y a diferencia del algoritmo anterior es esta máquina la encargada de generar una distribución de las partes, utilizando la misma lógica de Round-robin, después de generarla debe enviar la propuesta de distribución a los otros Datanodes y estos deben aceptarla, sino la aceptan generarán otra propuesta, de igual manera a como se explicó en el algoritmo centralizado la nueva propuesta será al azar y esta debe ser aceptada por todos los Datanodes sino se repetirá el proceso. Paralelamente el Datanode que envío la propuesta pide permiso para escribir en el log de *dist41*, si la propuesta es rechazada la escritura al log es rechazada, si la propuesta es aceptada el log es escrito y las partes son distribuidas.



Finalmente la descarga de libros es idéntica para ambos algoritmos, por lo que el cliente al iniciar la comunicación de descarga pide la lista de libros que está actualmente almacenada en sistema, esta es entregada por *dist41*. Luego de esto el cliente debe ingresar el nombre del libro que desea descargar y el Namenode le entregará la distribución de las partes del libro. Posteriormente el cliente habla con los respectivos Datanode en orden de la distribución y descarga los trozos del libro. Por último el cliente une el archivo.



Resultados

Para la toma de datos de ambos algoritmos se utilizó Pro Git segunda edición (licencia creative commons) con un tamaño de 18.7MB, por lo tanto sus particiones fueron 75. Se tomó el tiempo de inicio de proceso de envió de trozos y cuando el último trozo es recibido, se restó y se obtuvo el tiempo transcurrido para el envío total del libro, solo se hizo la descarga de datos puesto que la carga es el mismo proceso inverso y se omitió la partición y unión del libro ya que esto no se necesita una conexión entre máquinas.
Los datos obtenidos se muestran en las siguientes tablas:

Tabla 1: Centralizado

Intento	Tiempo
1	5.894929924s
2	5.653248537s
3	6.560481919s
4	5.588250575s
5	5.88030812s
6	6.41607848s
7	5.99043673s
8	6.42776884s
9	5.88951218s
10	6.43944378s
Promedio	6.0740459085s

Tabla 2: Distribuido

Intento	Tiempo
1	6.028306803s
2	7.872337856s
3	5.811903149s
4	6.923402748s
5	5.768757192s
6	6.268033159s
7	5.681181444s
8	5.189626984s
9	6.985159627s
10	6.280300632s
Promedio	6.2809009594s

Análisis

Con las tablas previamente presentadas se puede generar un promedio de los tiempos, como se muestra bajo ellas, dando un resultado aproximado de 6.0741s para el algoritmo centralizado y 6.2809s para el algoritmo distribuido.

Por otro lado se puede analizar la cantidad de mensajes que requieren ambos algoritmos, tomando en cuenta que n es la cantidad de particiones que se generó para el libro:

- Centralizado:
    - n, recibir n partes del cliente
    - 1, por enviar distribución a Namenode
    - n, al enviar n partes a Datanodes
    - **Total**: 2n+1
- Distribuido:
    - n, recibir n partes del cliente
    - 3n, aviso de escrito y aceptación de propuesta por parte de otros Datanode
    - 3f, avisos adicionales cada vez que falla la propuesta
    - 1, enviar distribución a Namenode
    - n, enviar n partes a Datanodes
    - **Total**: 5n+3f+1

Discusión

Se esperaba que el algoritmo centralizado fuese más lento ya que al enviar una distribución el Namenode debe revisarla y generar una nueva de ser el caso, haciendo un posible cuello de botella si existiesen muchas distribuciones enviadas, pero con los datos obtenidos de tiempos y mensajes se puede ver que no es así. Por otro lado el algoritmo distribuido necesita más procesamiento y comunicación ya que cada Datanode debe revisar la distribución y aceptarla, sino generar otra y esperar la respuesta de sus pares, haciendo que exista una cantidad de D-1 respuestas esperadas, siendo D la cantidad de Datanodes en el sistema, por lo que este problema se ven intensificado al agregar más Datanodes al sistema. Todo esto mientras siguen llegando libros al sistema y esos aun no están almacenados.

Por como está generado el criterio de rechazo en ambos algoritmos, en el caso distribuido teniendo una posibilidad de rechazo del 10 % por cada Datanode, se debe tener en cuenta que si se intenta seguir la opción distribuida es necesario generar un mejor criterio de rechazo.

Conclusión

Aunque el algoritmo centralizado en teoría podría ser más lento que su contra parte distribuida, en la práctica esto es ínfimo, la diferencia de los promedios de tiempos que se registró fue de menos de un segundo por lo que si el tiempo es un factor importante ambos algoritmos pueden ser utilizados, sin embargo, si el tráfico de la red preocupa entonces la mejor opción es utilizar el algoritmo centralizado ya que aunque falle la primera proposición de distribución la segunda siempre será realizada, por otro lado en el algoritmo distribuido esto no es así, puede existir múltiples distribuciones y la cantidad de mensajes escala rápidamente llenando la red.