DATOS MASIVOS I

UNIDAD II MODELO DE MAPEO Y REDUCCIÓN

MODELO COSTO – COMUNICACIÓN

Raúl Galindo Hernández

22 de Febrero de 2023

Costos en el Modelo Map – Reduce

- Costo de cómputo.
  - Trabajos de map (mapeo).
  - Trabajos de reduce (reducción).

Costos en el Modelo Map – Reduce

- Costo de comunicación.
  - Transmisión de los pares clave-valor de los nodos de map a los nodos reduce.
  - En las tareas de map usualmente no se incurre en costos de comunicación (se busca llevarlas a cabo en el mismo nodo donde se encuentran los datos, aunque se puede incurrir en costos de lectura de los datos de disco).

Costos en el Modelo Map – Reduce

- En un escenario común.
  - El costo de comunicación domina (es mayor que) al de cómputo.
  - El costo de los trabajos de Map es una pequeña fracción del costo de comunicación.
  - El costo que incurre el sistema por el ordenamiento es proporcional al costo de comunicación.

Costos en el Modelo Map – Reduce

- En los servicios de la nube (por ej. EC2) se paga tanto por el cómputo como por la comunicación, por lo que es importante tener un balance de ambos en el diseño de los algoritmos.

Modelo Costo – Comunicación

En Map – Reduce el costo de un algoritmo se calcula de la siguiente manera.

1) Costo de comunicación.

2) Costo de comunicación transcurrido.

3) Costo de computación transcurrido.

Costo de Comunicación

- El costo de comunicación es la suma del tamaño de todos los archivos pasados del proceso de mapeo hacia el proceso de reducción, agregado a la suma del tamaño de todos los archivos de salida en los procesos de reducción.

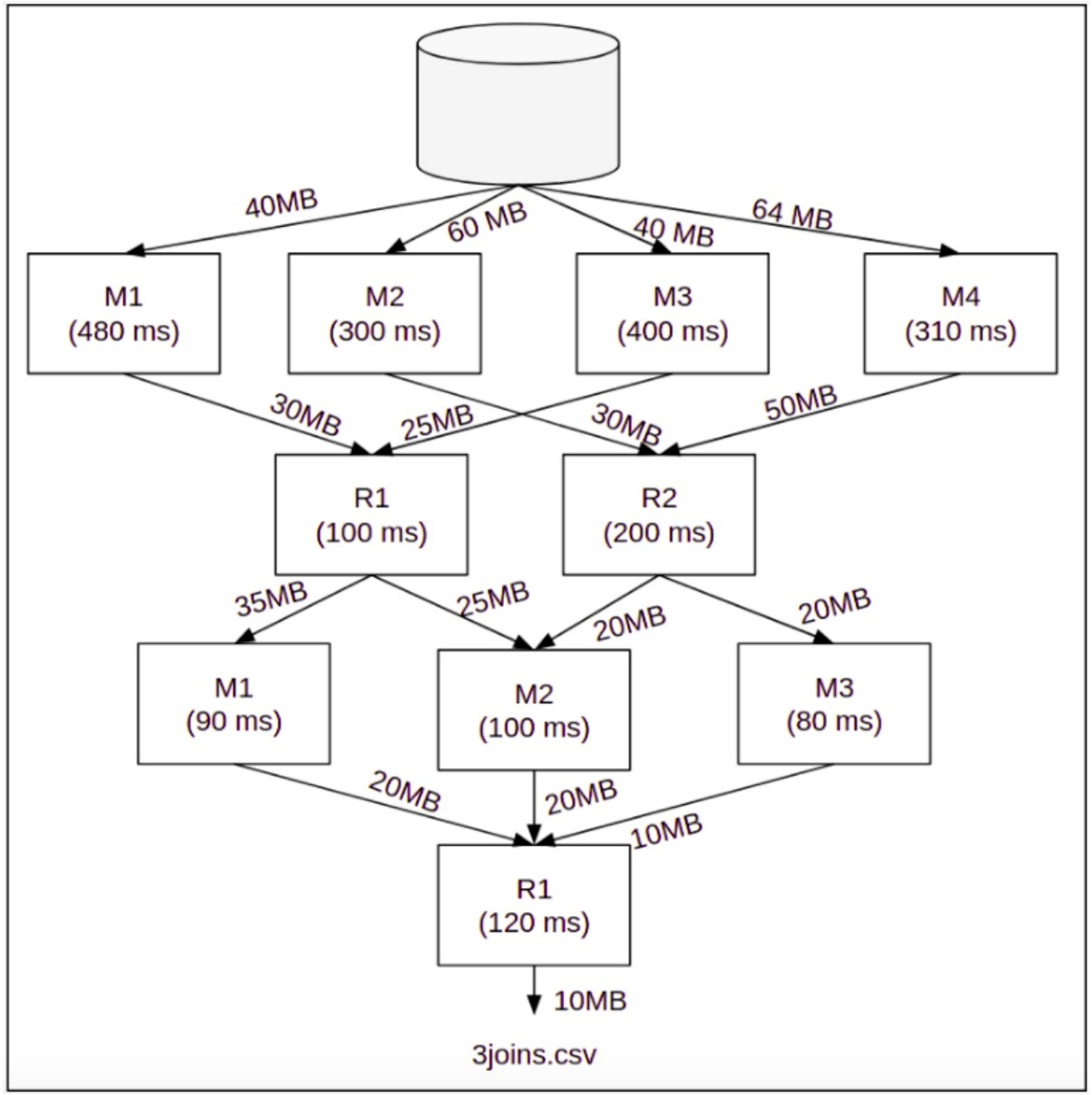
Costo de Comunicación Transcurrido

- El costo de comunicación transcurrido es la suma de la salida y de la entrada más grande para el proceso de mapeo, agregado a la suma de la salida y de la entrada más grande para el proceso de reducción.

Costo de Computación Transcurrido

- El costo de computación transcurrido es la suma únicamente del tiempo de ejecución de los procesos.

Modelo Costo – Comunicación: Ejercicio



Modelo Costo – Comunicación

- El costo de comunicación de un algoritmo es la suma de los costos de todas las tareas que requiere ejecutar.

Modelo Costo – Comunicación

- La meta es medir la eficiencia de un algoritmo basado en su costo de comunicación.
  - El cómputo que realiza cada tarea es usualmente muy simple y con complejidad lineal en el tamaño de la entrada.
  - Transmitir una entrada sobre la red puede ser muy lento y requerir tiempo para poder recibirla de otros nodos debido al tráfico en la red.

Unión de Dos Relaciones: Ejemplo

Problema: $R(A, B) \triangleright\triangleleft S(B, C)$

- Las tareas de mapeo tienen fragmentos de cualquiera de las dos.

Función de mapeo.

- Recibe tupla *t* de *R* o *S*.
- Produce par clave – valor.

Unión de Dos Relaciones: Ejemplo

Función de reducción.

- Recibe llave con lista de valores, la cual puede contener la tupla de *R*, la de *S* o ambas.
- Identifica tuplas que son de *R* de aquellas que son de *S* y las empareja.
- Produce *par(t, t)*

Costo de Comunicación de la Unión de Dos Relaciones: Map

- Supongamos que los tamaños de *R* y *S* son *r* y *s*.
- La entrada a las tareas de mapeo es un fragmento de los archivos que contienen a *R* o a *S*.

Costo de Comunicación de la Unión de Dos Relaciones: Map

- La suma del costo de comunicación de cada tarea de mapeo es $r + s$.

- El costo de comunicación del mapeo es $O(r + s)$.

Costo de Comunicación de la Unión de Dos Relaciones: Reduce

- La reducción se aplica a uno o más atributos.
- Cada reductor toma su entrada y la divide entre las tuplas que vienen de *R* y las que vienen de *S*.

Costo de Comunicación de la Unión de Dos Relaciones: Reduce

- Cada tupla de *R* se empareja con cada tupla de *S* para producir una salida.

- El tamaño de la salida puede ser mayor o menor a $r + s$, dependiendo de qué tanto se unan las tuplas.

Al diseñar un algoritmo para un clúster es importante también tomar en cuenta el tiempo que toma un algoritmo en ejecutarse en paralelo.

Complejidad para el Modelo Map – Reduce

- En Map – Reduce, el paralelismo es generado debido a que cada tarea de mapeo y reducción se ejecutan al mismo tiempo en diferentes nodos.
- El programador define el número de nodos para cada tarea.
- El modelo se encarga de planificar los pasos de mapeo, agrupación, reducción, rutas de los datos y tolerancia a fallos.

Rounds en el Modelo Map – Reduce

- Un *round* consiste de las tareas de *mapear – agrupar – reducir.*

- La salida de una tarea de reducción puede ser la entrada a otro *round*.

Rounds en el Modelo Map – Reduce

- En aplicaciones reales, el modelo Map – Reduce es una secuencia de tareas que se ejecutan a través de un número de *rounds*.

- Idealmente se espera que el número de *rounds* sea una constante.

Consulta Triangular: Ejemplo

M = tamaño de datos de entrada en bits

- Input: three tables
  R(X, Y), S(Y, Z), T(Z, X)

  size(R)+size(S)+size(T)=M

- Output: compute

  $Q(x,y,z) = R(x,y) \bowtie S(y,z) \bowtie T(z,x)$

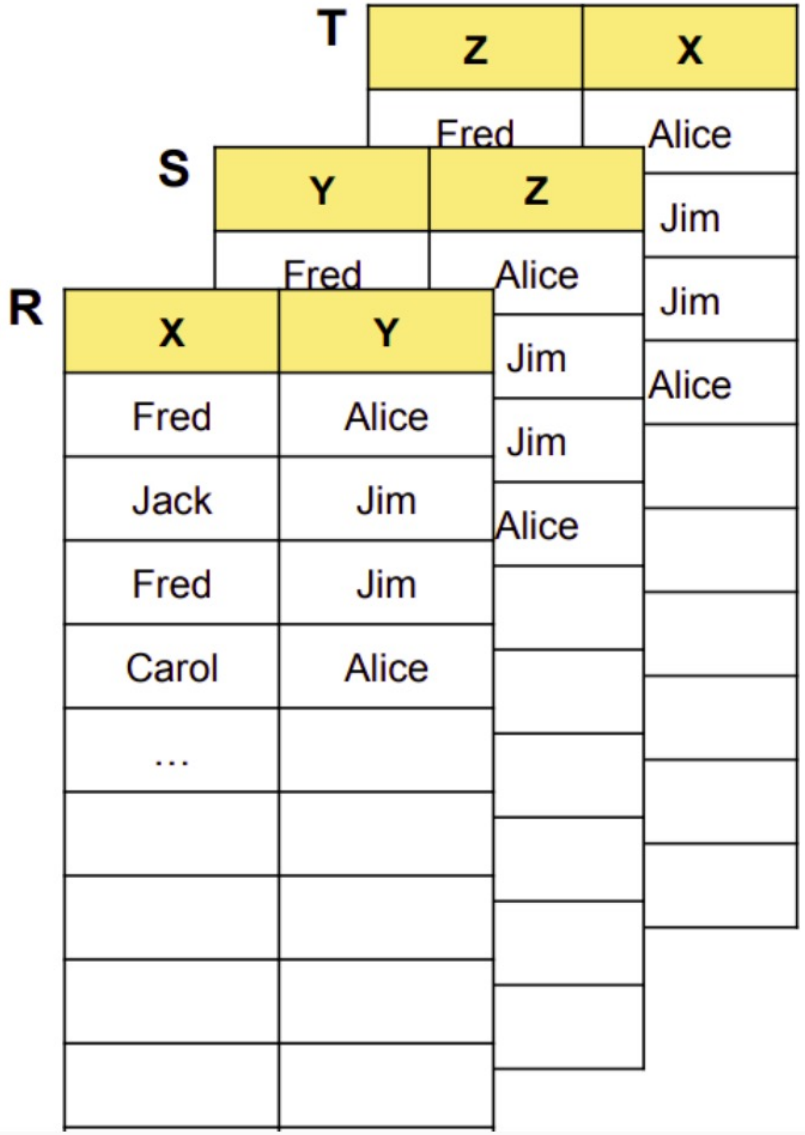


Imagen tomada de Dan Suciu, University of Washington

Ejemplo: Realizando la Unión en Dos Pasos

1) Calcular la primera unión (almacenarlo en *temp*).
temp(X, Y, Z) = *R(X, Y)* ⋈ *S(Y, Z)*

2) Calcular un segunda unión.
Q(X, Y, Z) = temp(X, Y, Z) ⋈ *T(Z, X)*

Ejemplo: Realizando la Unión en Dos Pasos

1) Calcular la primera unión (almacenarlo en *temp*).
temp(X, Y, Z) = $R(X, Y) \bowtie S(Y, Z)$

2) Calcular un segunda unión.
$Q(X, Y, Z) = temp(X, Y, Z) \bowtie T(Z, X)$

Notas.

- *temp* puede generar largas relaciones intermedias.
- El tiempo de comunicación y computación transcurrido puede verse afectadas (dependerán del tamaño de *temp*).

Eficiencia en Map – Reduce

Para medir la eficiencia del algoritmo Map – Reduce sobre el transcurso de su ejecución se debe obtener.

- R = Número de rounds que el algoritmo usará.
- C_r = La complejidad de comunicación del *round* r.
- t_r = El tiempo de ejecución interna.

Complejidad de Comunicación C_r del *Round r*

Se denota n_r , i como el tamaño de las I/O de las tareas de mapeo y reducción i en el *round r*.

$$C_r = \sum_i n_{r,i}$$ complejidad de comunicación del round r

$$C = \sum_{r=0}^{R-1} C_r$$ complejidad de comunicación para todo el algoritmo

Cálculo del Tiempo de Ejecución Interno t_r Para Cada *Round r*

Denotamos a t_r como el tiempo máximo de ejecución interno para la tarea de mapeo o reducción en el *round r*.

$$t = \sum_{r=0}^{R-1} t_r \text{ tiempo total de ejecución interno}$$

Parámetros Adicionales

Para calcular la eficiencia total de Map – Reduce se debe considerar:

- Latencia de la red (L). Se refiere al número de pasos que la tarea de mapeo o reducción tienen que esperar para recibir su primera entrada en una *ronda*.

Parámetros Adicionales

Para calcular la eficiencia total de Map – Reduce se debe considerar:

- Ancho de banda (B). Es el número de elementos que puede ser entregado por la red en una unidad de tiempo.

Tiempo Total de Ejecución

El tiempo total de ejecución para la implementación de un algoritmo se define de la siguiente forma (best case: Omega).

$$T = \Omega(\sum_{r=0}^{R-1}(t_r + L + C_r/B)) = \Omega(t + RL + C/B)$$

Ω representa el conjunto de todas las funciones de complejidad para un algoritmo (cota inferior).

Número de *Rounds*

- El número de *rounds* está proporcionalmente relacionado con el tiempo total de comunicación y la complejidad del algoritmo.

The less the better.

Retos en Map – Reduce

- Si nos enfocamos a tener un solo *round*, tendríamos un algoritmo ineficiente.

- Comunicación limitada. Existe una estricta modularidad que prohíbe que los nodos de reducción se comuniquen entre sí.

Retos en Map – Reduce

- Porción limitada de datos. Para cada nodo de mapeo y de reducción.

- Dado lo anterior, frecuentemente hay más de un *round*.

Paralelización en Map – Reduce

- Limitar la tasa de replicación.
- Cálculo de la complejidad previo al algoritmo.
- Uso de grafos.