

MapReduce – Aplikasi Fungsi dalam Pemrosesan Paralel

Ibrohim Kholilul Islam - 13513090¹
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
¹ibrohim@students.itb.ac.id

Abstract— Saat ini, perkembangan pengolahan data dalam jumlah besar sangat dibutuhkan. Hal ini disebabkan karena teknologi pengolahan informasi beberapa dekade terakhir baru mencapai tahap mengumpulkan data. MapReduce pada dasarnya merupakan proses fungsi map dan reduce pada paradigma pemrograman fungsional.

Keywords—MapReduce, pemrosesan paralel, sistem terdistribusi.

I. PENDAHULUAN

Pada dasarnya sebuah perkembangan teknologi didasarkan kepada kebutuhan manusia. Seperti perkembangan telepon seluler yang didasari kebutuhan komunikasi jarak jauh. Begitu juga yang terjadi di dunia ilmu komputer.

Saat ini, perkembangan pengolahan data dalam jumlah besar sangat dibutuhkan. Hal ini disebabkan karena teknologi pengolahan informasi beberapa dekade terakhir baru mencapai tahap mengumpulkan data.

Akibatnya ketika data sudah terkumpul dalam ukuran yang sangat besar, proses untuk mengolah data menjadi informasi tidak mampu dilakukan. Seharusnya dengan adanya data dengan ukuran yang besar dapat dilakukan ekstraksi informasi untuk kemudian diproses menjadi pengetahuan.

Namun, pemrosesan data dengan ukuran yang besar tidak mungkin dilakukan dengan satu perangkat saja. Di sinilah muncul alasan kebutuhan pemrosesan data secara paralel.

Pada dasarnya semua yang terjadi di dunia ini berlangsung secara paralel. Sebuah tumbuhan tumbuh secara bersamaan. Partikel-partikel air pada air terjun jatuh secara bersamaan. Begitu pula yang terjadi di otak manusia, pemrosesan data terjadi secara paralel pada sel-sel saraf.

Kebutuhan pemrosesan data secara paralel ini kemudian menuntun para peneliti untuk membuat model-model pemrosesan paralel. Beberapa model dalam pemrosesan paralel salah satunya adalah mapreduce.

Mapreduce pada awalnya merupakan fungsi dalam pemrograman dengan paradigma fungsional. Namun

kemudian berkembang menjadi sebuah kerangka kerja (*framework*) untuk dapat menangani reduksi dan toleransi kesalahan (*fault tolerant*).

II. SISTEM TERDISTRIBUSI

Sebelum tahun 1980 kesalahan perangkat keras merupakan faktor utama yang mempengaruhi sistem yang handal. Sampai pada akhirnya teknologi perkembangan perangkat keras seperti IC (*Integrated Circuits*) membuat kemajuan yang sangat pesat. Teknologi tersebut mengurangi radiasi panas dan faktor-faktor lain yang mempengaruhi kinerja elektronik. Sehingga pada dekade ini kesalahan perangkat keras tidak lagi menjadi faktor yang cukup berpengaruh.

Pada sistem yang cukup besar seperti Facebook, atau Google, keadaan down-time seperti pada saat upgrade sistem, kesalahan pendinginan, ataupun matinya aliran listrik dapat ditanggulangi dengan cara sistem terdistribusi. Sehingga, jika pada suatu saat sebuah perangkat mengalami kesalahan, terdapat perangkat lain yang dapat menyelesaikan fungsi perangkat tersebut.

Terdapat beberapa paradigma yang saat ini digunakan. Dari beberapa paradigma, paradigma fungsional merupakan paradigma yang secara alamiah sangat cocok dengan pemrosesan data secara paralel.

III. FUNGSI

Fungsi merupakan relasi biner yang memetakan dari daerah asal (*domain*) ke daerah hasil (*codomain*). Jika f merupakan fungsi yang memetakan dari himpunan A ke B maka dapat dituliskan sebagai

$$f: A \rightarrow B$$

yang artinya f memetakan dari daerah asal A ke daerah hasil B . Dari sebuah bahasan dalam dunia matematika inilah yang kemudian berkembang menjadi sebuah paradigma pemrograman, yakni paradigma fungsional.

IV. PARADIGMA FUNGSIONAL

Pada paradigma fungsional, sebuah proses akan menerima masukan untuk menghasilkan keluaran tertentu, tanpa mengubah keadaan (*state*) global. Artinya

pemrograman difokuskan pada penyelesaian masalah melalui dataflow.

Umumnya sebuah komputer didesain untuk melakukan langkah-langkah secara berurutan (*control flow*) berdasarkan model von Neumann. Sehingga perhitungan $D = B^2 - 4 * A * C$ dengan $C=1$, $A=1$, dan $B=-2$ akan dilakukan sebagai berikut

- 1 $C \leftarrow 1$
- 2 $A \leftarrow 1$
- 3 $B \leftarrow -2$
- 4 $T1 \leftarrow A * C$
- 5 $T2 \leftarrow 4 * T1$
- 6 $T3 \leftarrow B^3$
- 7 $D \leftarrow T3 - T2$

Sedangkan pada model dataflow, proses perhitungan akan dilakukan berdasarkan ketersediaan data untuk diproses, alur kerja akan dilakukan sebagai berikut:

- 1 $C \leftarrow 1; \quad A \leftarrow 1; \quad B \leftarrow -2$
- 2 $T1 \leftarrow A * C; \quad T3 \leftarrow B^3$
- 3 $T2 \leftarrow 4 * T1$
- 4 $D \leftarrow T3 - T2$

Sehingga paradigma ini secara alami merupakan paradigma yang sangat cocok untuk pemrosesan secara paralel.

V. FUNGSI SEBAGAI PARAMETER FUNGSI

Misalkan sebuah pada pemrosesan penambahan 1 untuk tiap elemen pada list.

$$sumlist(x) = \begin{cases} nil, & x = nil \\ conso(head(x) + 1, inclist(tail(x))), & x \neq nil \end{cases}$$

x	inclist(x)
[1, 2, 3]	[2, 3, 4]
[0, -1, -2]	[1, 0, -1]

Kemudian pada pemrosesan perkalian dengan 7 untuk tiap elemen pada list.

$$mullist(x) = \begin{cases} nil, & x = nil \\ conso(head(x) * 7, mullist(tail(x))), & x \neq nil \end{cases}$$

x	inclist(x)
[1, 2, 3]	[7, 14, 21]
[0, -1, -2]	[0, -7, -14]

Dengan melihat pola kedua fungsi dimungkinkan fungsi pemrosesan list dengan abstraksi yang lebih tinggi.

Fungsi yang menggunakan fungsi lain sebagai pemroses elemen pada tiap elemen list.

$$map(x, f) = \begin{cases} nil, & x = nil \\ conso(f(head(x)), map(tail(x))), & x \neq nil \end{cases}$$

Atau pada fungsi penjumlahan tiap elemen list

$$sumlist(x) = \begin{cases} 0, & x = nil \\ head(x) + sumlist(tail(x)), & x \neq nil \end{cases}$$

x	sumlist(x)
[1, 2, 3]	6
[0, -1, -2]	-3

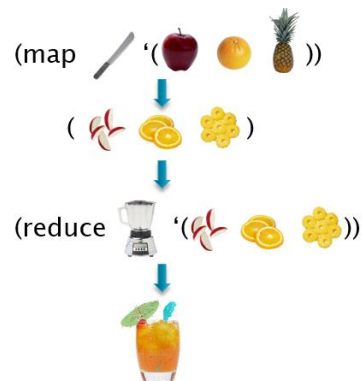
Melihat pola fungsi penjumlahan tiap elemen list, dapat dibuat sebuah fungsi yang lebih abstrak.

$$reduce(x, f, a) = \begin{cases} a, & x = nil \\ f(head(x) + reduce(tail(x), f, a)), & x \neq nil \end{cases}$$

VI. MAPREDUCE

Seperti yang telah dibahas sebelumnya, mapreduce pada awalnya merupakan model yang berawal dari paradigm fungsional yang secara alamiah merupakan paradigma yang sangat cocok dengan pemrosesan paralel. MapReduce merupakan salah satu model yang menggunakan kelebihan paradigma ini.

MapReduce pada dasarnya merupakan proses fungsi map dan reduce seperti yang telah dibahas sebelumnya. Sebagai contoh, pada pembuatan jus.

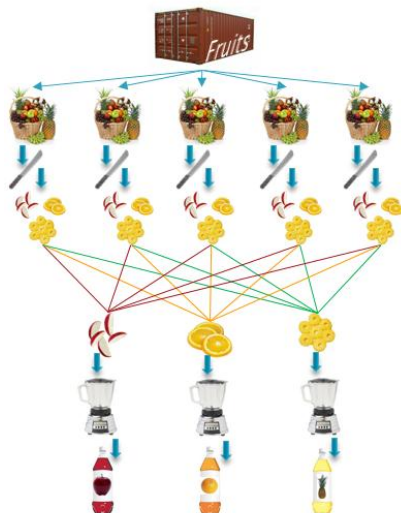


Gambar 1 Ilustrasi MapReduce (sumber: <http://salsahpc.indiana.edu>)

Pada dasarnya, proses map dilakukan untuk memroses informasi masukan menjadi sebuah informasi berupa tuple <kunci, nilai> berdasarkan masukan. Keluaran dari proses map ini kemudian dilakukan proses pengurutan. Kemudian dilakukan proses reduce sesuai dengan kunci menjadi sebuah nilai.

Pada skala yang lebih besar, proses pembuatan jus ini dapat dibuat menjadi dua buah proses. Proses pertama yakni proses map yang dilakukan secara paralel. Setelah proses map tersebut dilakukan, untuk sementara data

keluaran akan disimpan sampai akhirnya dilakukan proses reduce. Seperti yang dilakukan pada gambar 2.



Gambar 2 Ilustrasi MapReduce (sumber: <http://salsahpc.indiana.edu>)

Pada sistem cluster yang sebenarnya sebuah program pengguna akan melakukan pemecahan masukan menjadi beberapa potongan. Setelah itu program pengguna akan melakukan *fork* untuk menyediakan pekerja untuk map maupun reduce. Disaat itu pula program akan melakukan *fork* untuk melakukan penugasan kepada pekerja untuk melakukan map dari masukan sekaligus untuk menugaskan pekerja untuk melakukan reduce. Proses ini terjadi seperti gambar 3.

Model ini dapat digunakan untuk memecahkan beberapa persoalan. Salah satunya seperti pemrosesan teman yang sama pada sebuah jejaring sosial. Contoh lain, model ini dapat digunakan untuk menghitung banyaknya semua kata dengan jumlah huruf tertentu.

VI. HITUNG KATA

Pada proses hitung kemunculan setiap kata dengan panjang tertentu. Map ditentukan sebagai sebuah fungsi untuk melakukan perhitungan panjang kata. Kemudian pada proses pengurutan didefinisikan sebagai penggabungan tiap hasil dengan kunci yang sama menjadi sebuah list dan proses reduce merupakan proses untuk menghitung anggota list.

Misalkan masukan program adalah “dengan menentukan map adalah sebuah”. Program akan melakukan pemecahan kalimat masukan menjadi kata untuk kemudian dilakukan proses map.

w	map(w)
dengan	<6, dengan>
menentukan	<10, menentukan>
map	<3, map>
adalah	<6, adalah>
sebuah	<6, sebuah>

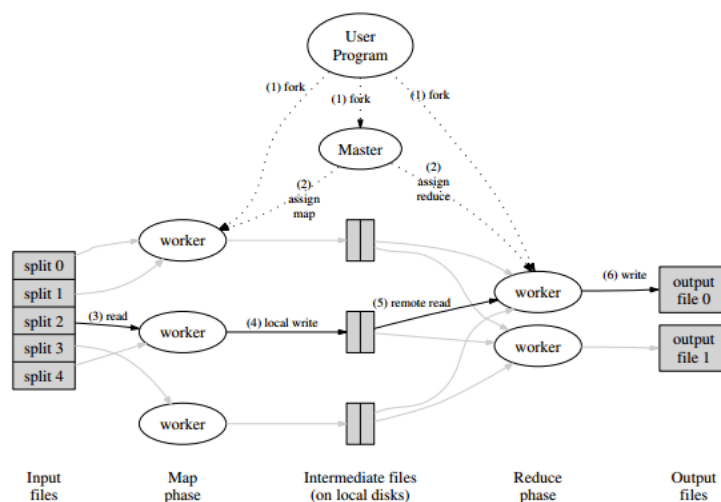
keluaran proses pengurutan dapat dilihat pada tabel berikut.

Hasil Pengurutan
<6, [dengan, adalah, sebuah]>
<10, [menentukan]>
<3, [map]>

keluaran proses reduce adalah sebagai berikut.

Hasil Pengurutan
<6, 2>
<10, 1>
<3, 1>

artinya pada terdapat 2 kata dengan panjang 6, 1 kata dengan panjang 10, dan 1 kata dengan panjang 3.



Gambar 3 MapReduce pada sebuah cluster (sumber: <http://research.google.com/archive/mapreduce-osdi04.pdf>)

VII. PENCARIAN TEMAN

Pada jejaring sosial seperti Facebook, terdapat sistem untuk mencari teman yang sama. Proses ini dilakukan pada data yang sangat besar, sehingga perlu dilakukan pemrosesan secara paralel. Salah satu model penyelesaian dengan map reduce yakni dengan menentukan fungsi map, sort, dan reduce sebagai berikut.

Map didefinisikan sebagai fungsi dengan input dengan kunci adalah identitas pengguna dan nilai adalah sebuah list pertemanan. Map tersebut menghasilkan tuple <kunci, nilai>. Kunci keluaran tersebut adalah sebuah himpunan dengan 2 anggota yakni identitas pengguna dan salah satu dari temannya. Nilai keluaran tersebut adalah list teman selain yang menjadi anggota pada kunci.

Pengurutan dalam proses ini dilakukan dengan menghimpun keluaran fungsi map dengan kunci yang sama. Keluaran proses ini yang kemudian diteruskan untuk dilakukan proses reduce.

Reduce didefinisikan sebagai fungsi yang menerima keluaran dari proses pengurutan. Untuk setiap masukan kunci-> nilai dengan nilai adalah list teman, keluaran berupa kunci-> nilai' dengan nilai' adalah sublist yang merupakan dari sublist dari kedua list pada nilai.

Misalkan input program berupa:

A -> B C D

B -> A C D E

C -> A B D E

D -> A B C E

map(A -> B C D):

(A B) -> B C D

(A C) -> B C D

(A D) -> B C D

map(B -> A C D E):

(A B) -> A C D E

(B C) -> A C D E

(B D) -> A C D E

(B E) -> A C D E

map(C -> A B D E):

(A C) -> A B D E

(B C) -> A B D E

(C D) -> A B D E

(C E) -> A B D E

map(D -> A B C E):

(A D) -> A B C E

(B D) -> A B C E

(C D) -> A B C E

(D E) -> A B C E

map(E -> B C D):

(B E) -> B C D

(C E) -> B C D

(D E) -> B C D

Hasil pengurutan:

(A B) -> (A C D E) (B C D)

(A C) -> (A B D E) (B C D)

(A D) -> (A B C E) (B C D)

(B C) -> (A B D E) (A C D E)

(B D) -> (A B C E) (A C D E)

(B E) -> (A C D E) (B C D)

(C D) -> (A B C E) (A B D E)

(C E) -> (A B D E) (B C D)

(D E) -> (A B C E) (B C D)

Sehingga setelah dilakukan proses reduce, keluaran adalah sebagai berikut:

(A B) -> (C D)

(A C) -> (B D)

(A D) -> (B C)

(B C) -> (A D E)

(B D) -> (A C E)

(B E) -> (C D)

(C D) -> (A B E)

(C E) -> (B D)

(D E) -> (B C)

Sehingga, ketika A membuka halaman profil D, maka dengan proses ini A akan menemukan bahwa A dan D keduanya berteman dengan B dan C.

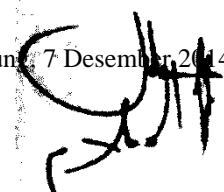
REFERENCES

- [1] Dawei Jiang, Beng Chin Ooi, Lei Shi, and Sai Wu. 2010. *The performance of MapReduce: an in-depth study*. Proc. VLDB Endow. 3, 1-2 (September 2010), pp. 472-483.
- [2] Jeffrey Dean and Sanjay Ghemawat, *MapReduce: Simplified Data Processing on Large Clusters*. Google, Inc. 2004. (diakses pada 7 Desember 2014) <http://research.google.com/archive/mapreduce-osdi04.pdf>.
- [3] Kenneth P. Birman, *Reliable Distributed System: Technologies, Web Services, and Applications*. Cornell University, Ithaca, NY: Departement of Computer Science, 2010, pp. 237-240.
- [4] Peter Henderson, *Functional Programming: Application and Implementation*. London: Prentice-Hall 1980, pp.242-246
- [5] Steve Krenzel, *MapReduce: Finding Friends*. 2010. (diakses pada 7 Desember 2014) <http://stevkrenzel.com/articles/finding-friends>
- [6] Ted G. Lewis, Hesham El-Rewini. *Introduction to Parallel Computing*, London: Prentice-Hall, 1992, pp. 215-218
- [7] Xiao Yang; Jianling Sun, "An analytical performance model of MapReduce," Cloud Computing and Intelligence Systems (CCIS), 2011 IEEE International Conference on , vol., no., pp.306,310, 15-17 Sept. 2011

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 7 Desember 2014


Brohim Kholilul Islam
13513090