Analisis Korelasi Pada Data Yahoo! Properties dan Instant Messaging dengan Menggunakan Hadoop

Mikiavonty Endrawati Mirabel[1], Henry Novianus Palit, Ph.D[2], Andreas Handojo, M.MT[3]
Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) – 8417658
E-Mail: mikiavonty@gmail.com[1], hnpalit@petra.ac.id[2], handojo@petra.ac.id[3]

ABSTRAK

Dalam beberapa tahun terakhir, *Big Data* menjadi *trend* dalam dunia teknologi informasi. Sejalan dengan berkembangnya organisasi semakin banyak data bisnis yang dimiliki, beberapa dapat mencapai skala Terabytes hingga Pentabytes. Ketika sebuah organisasi ingin melakukan analisa terhadap sebuah *Big Data* maka waktu analisa menjadi sangat lama dikarenakan keterbatasan CPU dan memori. Untuk mengatasi hal ini, munculah sebuah paradigma yaitu *distributed computing*. Ada banyak *tools* yang bertujuan untuk mengolah *Big Data* seperti Apache Hadoop, Apache Spark, dan sebagainya. Akan tetapi, yang akan dianalisa adalah Apache Hadoop.

Tingkat adopsi *Big Data* di Indonesia adalah 20% untuk 2 sampai 3 tahun ke depan [1]. Melihat fakta tersebut, semakin banyak perusahaan yang berencana mengadopsi *Big Data* untuk analisa datanya. Melihat sedang berkembangnya penggunaan *Big Data* dan Apache Hadoop, maka dilakukan eksplorasi analisis terhadap korelasi data pada Apache Hadoop. Selain itu, dilakukan juga pengujian Apache Hadoop dengan menggunakan jumlah *node* yang bervariasi, jumlah *mapper* dan *reducer*, dan menggunakan jumlah *block size* yang berbeda [10].

Eksplorasi analisis korelasi data pada Apache Hadoop dilakukan dengan membuat 4 jenis aplikasi analisa data, yaitu dua aplikasi pencarian nilai korelasi untuk data Yahoo! Messenger dan dua aplikasi pencarian pohon klasifikasi. Berdasarkan hasil pengujian didapatkan bahwa untuk pencarian nilai korelasi aplikasi R lebih cocok untuk ukuran data yang lebih kecil sedangkan Hadoop lebih cocok untuk ukuran data yang besar. Untuk data yang besar, aplikasi R menggunakan persentase CPU dan memori yang lebih tinggi daripada Hadoop. Kombinasi *mapper* dan *reducer* yang akan memberikan waktu eksekusi optimal adalah untuk *mapper* dalam rentang 2/3 dari total CPU *core*, sedangkan *reducer* adalah 1/3 dari total CPU *core*.

Kata Kunci

Hadoop, Big Data, *Mapreduce*, Analisa korelasi

ABSTRACT

In the past few years, Big Data has been a trend in the world of Information Technology. Alongside the growth of organizations, the bigger the data that is owned, some can reach the scale of Terabytes to Pentabytes. When an organization wants to do an analysis of a Big Data it takes time due to limited CPU and memory. To overcome this, there is a paradigm which is called distributed computing. There are many tools that aim to cultivate Big Data such as Apache Hadoop, Apache Spark, and etc. However, the tool that is going to be analyzed is the Apache Hadoop.

The adoption rate of Big Data in Indonesia is 20% for 2 to 3 years into the future [1]. Seeing these facts, more and more companies are planning to adopt Big Data to analyze their data. Due to the increasing use of Big Data and Apache Hadoop, it is conducted an exploratory analysis of data correlation on Apache Hadoop. In addition, testing was also done using the Apache Hadoop with varying number of nodes, mappers and reducers, and number of different block sizes [10].

Exploratory analysis of data correlation on Apache Hadoop is done by making four types of data analysis applications, namely two applications correlation values for the data search Yahoo! Messenger and two applications for creating classification trees. Based on test results obtained that the R application is more suitable for smaller data size while Hadoop is more suitable for large data size. For large data, application R uses higher percentage of CPU and memory than Hadoop. The combination of mapper and reducer that will provide the most optimal execution time is for mapper in the range of 2/3 of the total CPU cores, while the reducer is 1/3 of the total CPU cores.

Keywords

Hadoop, Big Data, Mapreduce, Correlation Analysis

1. PENDAHULUAN

Dalam beberapa tahun terakhir, *Big Data* menjadi *trend* dalam dunia teknologi informasi. Sejalan dengan berkembangnya organisasi semakin banyak data bisnis yang dimiliki, beberapa dapat mencapai skala *Terabytes* hingga *Pentabytes*. Kebanyakan dari data yang tersimpan tersebut awalnya ditujukan untuk kepentingan arsip. Beberapa dari organisasi ini melihat manfaat dari analisa terhadap data tersebut. Analisa tersebut dapat menghasilkan informasi yang berguna bagi organisasi.

Big Data memiliki ukuran dan jumlah yang besar. Ketika sebuah organisasi ingin melakukan analisa terhadap sebuah *Big Data* dengan hanya menggunakan satu komputer maka waktu analisa menjadi sangat lama. Hal ini dikarenakan proses analisa terbatas oleh kapasitas dari komputer, seperti memori dan CPU. Untuk mengatasi hal ini, munculah sebuah paradigma yaitu *distributed computing*. *Distributed Computing* adalah proses komputasi yang dilakukan dengan memecah sebuah *task* kepada banyak komputer untuk diselesaikan. Ada banyak *tools* yang bertujuan untuk mengolah *Big Data* seperti *Apache Hadoop, Apache Spark*, dan sebagainya. Akan tetapi, menurut InfoWorld, alat yang paling populer untuk pelaksanaan *distributed computing* adalah Apache Hadoop yang dapat diimplementasikan pada *single computer* ataupun *multiple computer* dalam suatu jaringan tertentu. *Tool* ini memiliki peran yang sangat penting dalam proses pengolahan *Big Data*.

Tingkat adopsi *Big Data* di Indonesia adalah 20% untuk 2 sampai 3 tahun ke depan [1]. Melihat fakta tersebut, semakin banyak perusahaan yang berencana mengadopsi *Big Data* untuk analisa datanya. Melihat sedang berkembangnya penggunaan *Big Data* dan *Apache Hadoop*, maka dilakukan eksplorasi analisis terhadap korelasi data pada *Apache Hadoop*. Skripsi ini dibuat untuk melakukan eksplorasi analisis korelasi dengan menggunakan *Apache Hadoop*. Bagaimana mengimplementasikan analisa korelasi pada Apache Hadoop. Diharapkan melalui skripsi ini, dapat membuka lebih banyak penggunaan dan penelitian dengan menggunakan *Big Data* di Indonesia. Selain itu, organisasi maupun perusahaan dapat mencari informasi mengenai korelasi data yang dapat berguna bagi mereka. Data yang akan digunakan adalah data milik Yahoo! yang berisikan data komunikasi IM dari user, umur, jenis kelamin, Yahoo! Property use, dan karakteristik dari *mobile phone* yang digunakan. *Dataset* ini diambil dari *website Webscope* milik Yahoo yang memang disediakan untuk kepentingan penelitian.

2. DASAR TEORI

2.1 Big Data

Konsep "*Big Data*" pertama kali dicetuskan oleh Roger Magoulas di dalam media O'Riley pada tahun 2005. *Big Data* menjelaskan bahwa data yang ada begitu besar dan banyak sehingga manajemen data tradisional tidak dapat digunakan lagi. Pandangan IBM terhadap *Big Data* [10] dapat dibagi menjadi empat aspek, yaitu:

1. *Volume*

   Kuantitas data yang dimiliki oleh perusahaan. Data ini nantinya digunakan untuk menghasilkan pengetahuan yang penting.

2. *Velocity*

   Berapa lama waktu yang dibutuhkan untuk memproses *Big Data* tersebut. Beberapa aktivitas yang ada itu sangat penting dan membutuhkan respon yang cepat. Untuk itu proses yang dilakukan harus diefisienkan.

3. *Variety*

   Meliputi tipe data apa saja yang ada di dalam *Big Data*. Data yang digunakan dapat terstruktur atau tidak terstruktur.

4. *Veracity*

   Seberapa percaya pemimpin perusahaan mengambil keputusan menggunakan hasil olahan data yang ada. Jadi, mencari korelasi yang tepat sangat penting bagi masa depan perusahaan.

2.2 Metode Korelasi

2.2.1 Nilai R

Nilai R [3] adalah koefisien korelasi, yaitu derajat hubungan antar 2 variabel dan arah dari relasi tersebut, apakah relasi tersebut positif ataukah negatif. Koefisien korelasi dapat dilambangkan sebagai r. Interval dari r adalah $-1 < r < 1$. Semakin dekat angka r dengan angka 1 ataupun -1 maka dikatakan hubungan korelasi yang ada sangat dekat. Apabila angka mendekati 1 maka hubungan yang ada semakin positif, sedangkan semakin mendekati -1 maka hubungan korelasi yang ada semakin negatif. Sebaliknya apabila angka mendekati 0 maka hubungan semakin lemah dan bahkan tidak memiliki korelasi.

2.2.2 Metode Pearson

Untuk melihat kedekatan hubungan korelasi antar data dapat dilihat melalui koefisien korelasi seperti yang dijelaskan pada 2.2.1. Terdapat beberapa macam metode untuk menghitung koefisien korelasi ini. Metode yang akan dibahas di dalam skripsi ini adalah metode Pearson, Spearman, dan Point-Biserial.

Metode Pearson [3] menghitung korelasi secara linear antara 2 pasang variabel. Asumsi dari metode Pearson adalah variabel ditentukan dengan nilai level atau *ratio*, data yang ada terdistribusi normal. Metode ini dapat dituliskan sebagai berikut:

$$r = \frac{N \sum xy - \sum (x)(y)}{\sqrt{N \sum x^2 - \sum (x^2)][N \sum y^2 - \sum (y^2)]}} \quad (1)$$

Keterangan:

r = Koefisien korelasi Pearson
n = Jumlah variabel
∑xy = total sum perkalian variabel 1 dan 2
∑x = total *sum* variabel 1
∑y = total *sum* variabel 2
∑x2 = total *sum* kuadrat dari variabel 1
∑y2 = total *sum* kuadrat dari variabel 2

2.2.3 Metode Spearman

Jika data yang ada tidak memenuhi asumsi dari Pearson maka digunakan metode Spearman. Metode Spearman [3] adalah pengembangan dari Metode Pearson pada 2.2.2. Metode Spearman dapat digunakan dalam 2 kondisi, yaitu:

1. Kedua pasang data yang digunakan adalah variabel yang dihitung dalam skala ordinal. Skala ordinal adalah nilai didapat dari *ranking* data bukan melalui nilai data.
2. Kedua pasangan data yang digunakan adalah data *ratio* atau interval, tetapi tidak memiliki hubungan yang linear. Seperti contoh adalah hubungan antara waktu berlatih dengan tingkat performa seseorang. Hubungan yang ada tidak dalam garis linear melainkan membentuk kurva.

Rumus dari metode Spearman adalah sama dengan Pearson pada persamaan 1, hanya saja nilai x dan y yang dimasukkan ke dalam pesamaan adalah hasil *ranking* data. Cara melakukan *ranking* dapat dilihat pada Gambar 1.

Person	X	Y	X−Rank	Y−Rank
A	4	9	3	3
B	2	2	1	1
C	10	10	4	4
D	3	8	2	2

Gambar 1. Perhitungan *Ranking* Nilai yang Berbeda [3]

Untuk perhitungan *ranking* data yang memiliki nilai yang sama dapat dilihat pada Gambar 2.

Scores	Rank Position	Final Rank	
3	1	1.5	Mean of 1 and 2
3	2	1.5	
5	3	3	
6	4	5	Mean of 4, 5, and 6
6	5	5	
6	6	5	
12	7	7	

Gambar 2. Perhitungan *Ranking* dengan Nilai yang Sama [3]

2.2.4 *Metode Point-Biserial*

Metode Point-Biserial [3] digunakan untuk menghitung korelasi antara 2 variabel yang satu adalah variabel dengan nilai numerik, yang kedua adalah variabel *dichtonomous* atau binomial. Variabel *dichtonomous* adalah variabel yang hanya memiliki 2 nilai. Seperti contoh adalah pria dan wanita. Untuk menghitung korelasi dengan metode Point-Biserial maka perlu dilakukan konversi dari kategori ke nilai 0 atau 1. Misal pria adalah 0 dan wanita adalah 1. Setelah dilakukan konversi maka dilakukan perhitungan dengan rumus Pearson pada persamaan 2.5, dimana nilai X dan Y adalah data yang telah dikonversikan.

2.3 Metode Classification Tree CART

Classification Tree adalah sebuah *decision tree* yang dibuat melalui perhitungan gini Index dari sebuah data. *Classification Tree* dengan algoritma CART [7] dapat dijelaskan sebagai berikut:

1. Algoritma dimulai dari *node root*
2. Dilakukan perhitungan Gini untuk *root*
3. Untuk setiap kemungkinan *split* data, dilakukan perhitungan Gini Indexnya. Kemudian dicari Gini Index yang paling minimum (Kriteria yang memberikan delta gini maksimum).
4. Pecah *node* berdasarkan kriteria yang didapatkan dari langkah 3. Jika *node* anak telah mencapai Gini Index 0 atau tidak ada kemungkinan lagi untuk melakukan perpecahan maka proses berhenti. Jika tidak, maka dilakukan kembali langkah nomor 2.

Gini Index adalah proporsi ketidakrataan data. Rumus Gini Index untuk satu *node* [6] dapat dilihat pada persamaan 2.

$$gini\ (D) = 1 - \sum_{j=1}^{n} p_j^2 \qquad (2)$$

Setelah dilakukan perhitungan Gini Index sebuah *node* maka dilakukan perhitungan Gini Index untuk hasil perpecahan [6], dengan menggunakan persamaan 3.

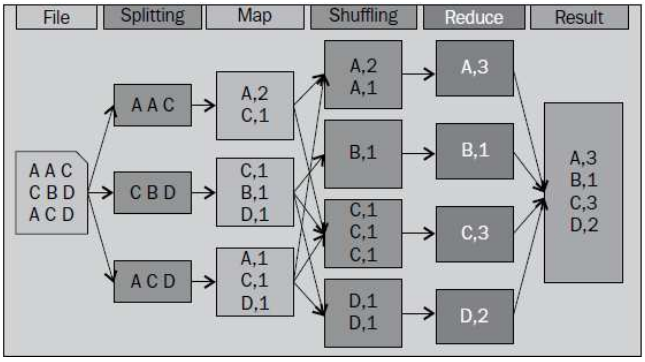
$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2) \qquad (3)$$

Setelah ditemukan nilai Gini Index dari persamaan 2 dan nilai Gini Index Split dari persamaan 3, maka dilakukan pencarian Delta Gini [6] yang didapatkan dari persamaan 4.

$$\Delta gini(A) = gini(D) - gini_A(D) \qquad (4)$$

2.4 *Mapreduce*

Mapreduce [11] adalah sebuah model *programming* sederhana untuk memproses data. Program yang mengimplementasikan *Mapreduce* dapat berjalan secara paralel sehingga dapat menyelesaikan analisis data dengan ukuran yang sangat besar. *Mapreduce* terdiri dari 2 bagian yaitu fase *map* dan fase *reduce*. Kedua fase memiliki sepasang *key* dan *value* sebagai *input* dan *output* yang tipenya dipilih oleh *programmer*. *Programmer* harus menentukan juga isi fungsi *mapper* dan fungsi *reducer*. Arsitektur *Mapreduce* pada Hadoop dapat dilihat pada Gambar 3.



Gambar 3. Aristektur Mapreduce pada Hadoop [11]

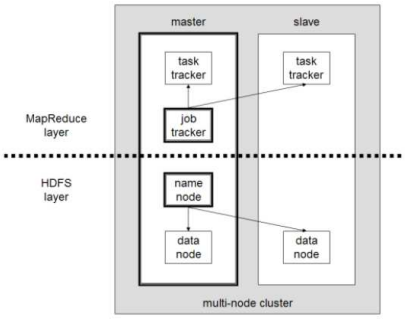
2.5 Distributed Systems

Distributed System pada sistem komputer adalah kumpulan proses yang terotomatisasi, berkomunikasi di atas sebuah jaringan yang memiliki fitur tidak ada proses *clock* secara fisik, tidak ada *shared memory*, prosesor yang digunakan adalah *loosely coupled* (komponen prosesor dapat bekerja tanpa harus mengetahui secara detail komponen yang lain), otomatis dan heterogen [5]. Motivasi untuk menggunakan *distributed system* adalah untuk:

1. Komputasi terdistribusi yang inheren
2. Berbagi *resource*
3. Akses ke data dan *resource* yang jauh
4. Meningkatkan *reliability* dari sistem
5. Meningkatkan performa
6. Skalibilitas
7. *Modularity* (Kemudahan menambah prosesor tanpa mempengaruhi sistem)

2.6 Apache Hadoop

Hadoop dibuat oleh Doug Cutting, pembuat Apache Lucene [4]. Hadoop adalah sebuah *framework* yang memungkinkan distributed processing di berbagai *cluster* yang terpisah. Hadoop bekerja dalam *environment* Java. Arsitektur dari Hadoop dapat dilihat pada Gambar 4.



Gambar 4. Arsitektur Hadoop Ward [4]

Hadoop terdiri dari 2 bagian, yaitu HDFS (*Hadoop Distributed File System*) *file system* dan *Mapreduce*. HDFS [2] terdiri dari *name node* dan *data node*, sedangkan *Mapreduce* terdiri dari *job tracker* dan *task tracker*. Pada saat ini, Hadoop adalah sistem tercepat untuk mengolah data dalam Terabyte. Hadoop [8] dapat dijalankan dengan 3 macam cara, yaitu:

- *Standalone mode*: *default mode* yang disediakan oleh Hadoop. Semua dijalankan melalui sebuah proses Java.

- *Pseudo – distributed mode*: Hadoop dikonfigurasi untuk berjalan pada sebuah mesin, akan tetapi dengan beberapa daemon Hadoop yang berjalan sebagai proses Java yang berbeda.
- *Fully distributed atau cluster mode*: Di sini sebuah mesin di sebuah *cluster* yang dilabel *namenode* dan yang lain sebagai *Jobtracker*. Hanya satu *namenode* yang diletakkan dalam sebuah *cluster*. *Namenode* ini bertugas me-*manage namespace*, *metadata filesystem*, dan kontrol akses. *Secondary namenode* bisa diletakkan untuk proses *handshaking* secara periodik dengan *namenode* untuk toleransi kegagalan. Semua mesin di dalam *cluster* bekerja sebagai *datanode* dan *tasktracker*. *Datanode* menyimpan data sistem. Tiap *datanode* menyimpan *local storage* masing – masing. *Tasktracker* yang bertugas menjalankan operasi *Map* dan *Reduce*.

Pada penelitian ini, Hadoop dijalankan dengan menggunakan *fully distributed mode*.

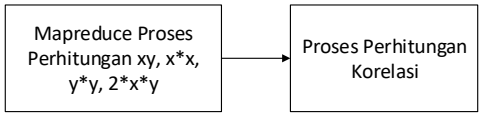
2.7 Data Yahoo! Property dan Instant Messenger

Dataset ini didapatkan dari Webscope milik Yahoo! Research bagian G7. *Dataset* ini merupakan data dari 1 Oktober 2007 – 30 Oktober 2007 [12]. *Dataset* ini terdiri dari 31 *file* yang di-*host* di AWS (*Amazon Web Service*)[1]

3. DESAIN SISTEM

3.1 Aplikasi *Mapreduce* Korelasi

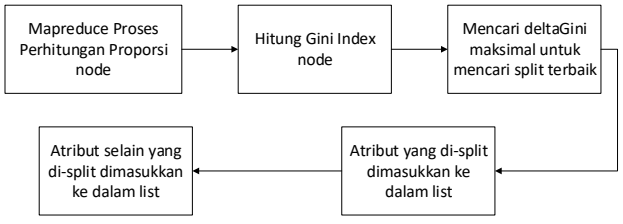
Aplikasi ini bertugas untuk menghitung korelasi Pearson, Spearman, dan Point-Biserial. Proses perhitungan secara umum terdiri dari *mapreduce* yang digunakan untuk menghitung x, y, xy, x*x, y*y, dan x*y. Setelah itu nilai x dan y dihitung ke dalam rumus Pearson, Spearman, maupun Point-Biserial. Proses dapat digambarkan pada Gambar 5.



Gambar 5. Proses perhitungan korelasi secara umum

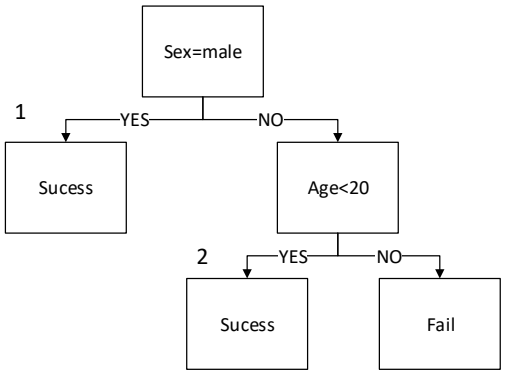
3.2 Aplikasi *Mapreduce* CART

Proses pencarian pohon klasifikasi secara umum dapat dilihat pada Gambar 6.



Gambar 6. Proses CART Secara Umum

Proses CART dimulai dengan melakukan proses *mapreduce* untuk melakukan perhitungan proporsi *node*. Setelah itu dilanjutkan dengan menghitung Gini Index *node* sekarang. Setelah didapatkan nilai Gini Index sekarang maka dicarilah untuk tiap atribut manakah atribut yang menghasilkan delta gini paling maksimal. Atribut yang menghasilkan delta gini maksimal dipilih sebagai kategori *split*. Kategori dimasukkan ke dalam *list* sebagai *node* selanjutnya, kemudian kategori selain yang dipilih dimasukkan sebagai *node* selanjutnya yang kedua. Contoh hasil *decision tree* CART dapat dilihat pada Gambar 7.



Gambar 7. Contoh *Decision Tree* Hasil CART

4. IMPLEMENTASI SISTEM

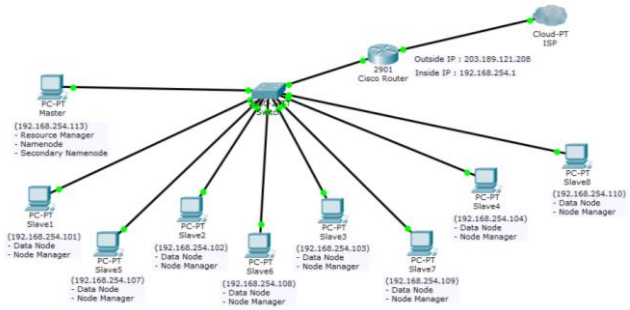
Hadoop diimplementasikan dengan jumlah 9 komputer. Satu komputer sebagai *master* dan 8 sisanya adalah *slave*. Spesifikasi komputer yang digunakan adalah sebagai beikut:

- Prosesor : Intel ® *Core* ™ i5-4570CPU @ 3.20GHz (4 CPUs)
- Memori yang ter*install* (RAM) 8 GB
- Tipe sistem : 64-bit
- Fast Ether*net* 100Mbps

Pada tiap komputer di-*install*:

- Ubuntu Server 16.04.1 LTS
- Hadoop 2.7.2
- Java default-jdk
- SSH-Server
- Procps 2:3.3.9

Topologi jaringan yang digunakan dapat dilihat pada Gambar 8.



Gambar 8. Topologi Jaringan

5. ANALISA DAN PENGUJIAN

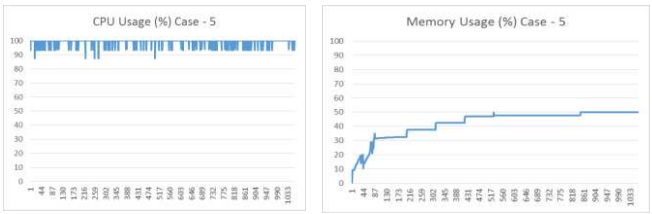
5.1 Aplikasi Korelasi antara Jenis Kelamin dengan Jenis *Page*

Data yang digunakan untuk aplikasi ini adalah data jenis kelamin dan jenis *page* yang dibuka. Besar data yang digunakan untuk kedua aplikasi adalah yaitu 1.106 GB.

[1] https://webscope.sandbox.yahoo.com/catalog.php?datatype=g

5.1.1 *Pengujian dengan R*

Pengujian pada R dilakukan sebanyak 10 kali. Dari hasil pengujian didapatkan rata penggunaan CPU pada aplikasi R adalah sebesar 98.17%, penggunaan memori sebesar 42.09%. Rata – rata waktu eksekusi 179.9 detik. Grafik penggunaan CPU dapat dilihat pada Gambar 9 (a) dan grafik penggunaan memori pada Gambar 9 (b).

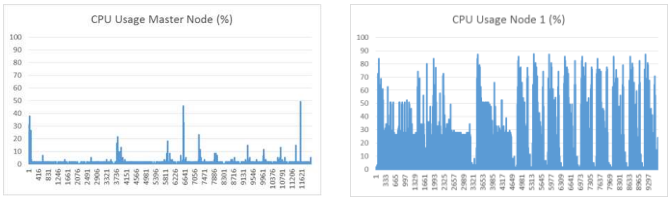


(a) (b)

Gambar 9. Grafik Penggunaan CPU dan Memori pada R (a) Grafik Penggunaan CPU (b) Grafik Penggunaan Memori

5.1.2 *Pengujian dengan Hadoop*

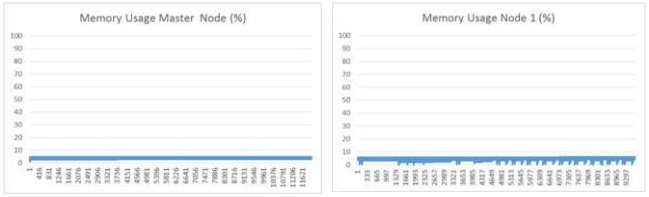
Setelah dilakukan pengujian terhadap perhitungan nilai korelasi pada R, maka dilakukan pengujian data pada aplikasi perhitungan korelasi yang telah dibuat pada *Mapreduce* Hadoop pada 5.1.1 Data yang digunakan sama dengan yang digunakan pada R. Pengujian dilakukan sebanyak 5 kali pada Hadoop. Dari hasil pengujian didapatkan bahwa rata – rata penggunaan CPU adalah 24.408% dan penggunaan memori adalah 16.308%. Rata – rata waktu yang dibutuhkan untuk menyelesaikan satu *job* tersebut adalah 494.964 detik dengan jumlah rata – rata sampel 5.278 per detik.



(a) (b)

Gambar 10. Grafik Penggunaan CPU (a) Master (b) *Node* 1

Melihat Gambar 10 (a) dan Gambar 10 (b), tingkat penggunaan CPU oleh Hadoop lebih rendah dibandingkan penggunaan CPU dengan aplikasi R. Hal ini dikarenakan Hadoop menggunakan *multicore* sedangkan aplikasi R menggunakan *single core* CPU.



(a) (b)

Gambar 11. Grafik Penggunaan Memori (a) Master (b) Slave

Dari Gambar 11, diketahui bahwa penggunaan memori pada Hadoop jauh lebih rendah dibandingkan aplikasi R. Untuk waktu eksekusi aplikasi pada R lebih cepat dibandingkan pada Hadoop dengan kondisi proses komputasi dilakukan sama – sama pada 1 komputer.

5.1.3 *Perbandingan R dengan Hadoop*

Melihat dari hasil percobaan pada 5.1.1 dan 5.1.2, didapatkan Tabel 1. Dari Tabel 1, terlihat bahwa penggunaan CPU dan memori pada R jauh lebih tinggi dibandingkan pada Hadoop, akan tetapi waktu eksekusi pada Hadoop 4 kali lebih lama daripada R. Dari pengujian ini dibandingkan dengan pengujian pada 5.2 didapatkan bahwa dengan jumlah data yang lebih besar, waktu eksekusi Hadoop semakin minimum.

Tabel 1 Tabel Perbandingan R dengan Hadoop

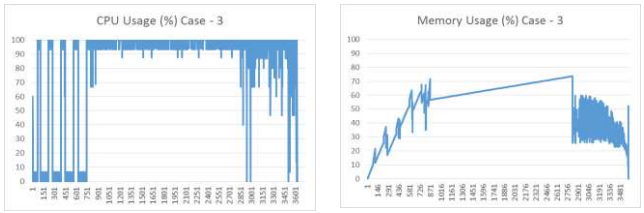
Jenis	CPU (%)	Memori (%)	Waktu (detik)
R	98.17	42.09	179.9
Hadoop			
Node 1	24.41	16.31	494.9
Node 2	13.27	14.93	442.5
Node 4	10.97	12.16	444.4
Node 8	11.909	11.76	366.1

5.2 Aplikasi Korelasi antara Umur dan Frekuensi Komunikasi

Data yang digunakan untuk aplikasi ini adalah data jenis kelamin dan frekuensi komunikasi. Besar data yang digunakan untuk kedua aplikasi adalah yaitu 5.37 GB.

5.2.1 *Pengujian dengan R*

Setelah dilakukan pengujian, secara rata – rata aplikasi R menggunakan 84.08% dari total CPU. Untuk penggunaan memori rata – rata menggunakan 52.54%. Total rata – rata waktu yang dibutuhkan untuk menyelesaikan aplikasi tersebut adalah 616.7 detik. Jumlah sampel per detik dari aplikasi ini adalah rata – rata 5.92 sampel per detik.



(a) (b)

Gambar 12. Grafik Penggunaan CPU dan Memori pada R (a) Grafik Penggunaan CPU (b) Grafik Penggunaan Memori

Melihat Gambar 12 (a), bahwa penggunaan CPU R sangat tinggi yaitu rata – rata 90 – 100%. Hal ini dikarenakan aplikasi R secara *default* hanya menggunakan satu *core* untuk prosesnya. Untuk Gambar 12 (b), tingkat penggunaan memori termasuk tinggi yaitu memakan 52.54% dari total memori. Penggunaan memori pada aplikasi ini telah dioptimasi dengan menggunakan *garbage collector* dimana memori yang sudah tidak dipakai dilepas dari proses.

5.2.2 *Pengujian dengan Hadoop*

Setelah dilakukan pengujian terhadap perhitungan nilai korelasi pada R, maka dilakukan pengujian data pada aplikasi perhitungan korelasi yang telah dibuat pada *Mapreduce* Hadoop. Data yang digunakan sama dengan yang digunakan pada R. Dilakukan percobaan sebanyak 5 kali pada Hadoop. Hasil percobaan dapat

dilihat pada tabel pada Tabel 2. Berdasarkan Tabel 2, didapatkan bahwa rata – rata penggunaan CPU adalah 11 % dan penggunaan memori adalah 13 %. Rata – rata waktu yang dibutuhkan untuk menyelesaikan satu *job* tersebut adalah 380-480 detik.

Tabel 2 Tabel Hasil Pengujian Waktu, CPU, dan *Memory* Analisa Korelasi Umur dan Frekuensi Komunikasi pada *Hadoop*

Jenis	256 MB		
	CPU (%)	Memori (%)	TIME (seconds)
Node 6	11.717	13.748	474.73
Node 8	11.054	11.389	380.82
Jenis	**512 MB**		
	CPU (%)	**Memori (%)**	**TIME (seconds)**
Node 6	10.672	17.628	507.044
Node 8	9.11	15.502	438.17

5.2.3 Perbandingan R dengan Hadoop

Pada percobaan korelasi antara umur dan frekuensi komunikasi, pada Hadoop, data tidak dapat dijalankan pada jumlah *node* <=4. Hal ini dikarenakan memori java *(Java Heap space*) tidak cukup untuk menampung hasil olahan *mapreduce*. Akan tetapi, ketika jumlah *node* dinaikkan maka data dapat diolah dengan baik. Hasil percobaan dapat dilihat pada Tabel 3.

Tabel 3 Hasil Pengujian antara R dengan Hadoop

Jenis	CPU (%)	Memori (%)	TIME (seconds)
R	84.08	52.54	616.7
Hadoop			
Node 6	11.1945	15.688	490.887
Node 8	10.082	13.4455	409.495

Dari Tabel 3, dapat dilihat bahwa pengolahan data dengan Hadoop menghasilkan waktu eksekusi kurang lebih 2 kali lebih cepat daripada menggunakan R. Untuk mencari tahu konfigurasi optimal maka dilakukan pengujian terhadap kombinasi jumlah *mapper* dan *reducer*. Didapatkan hasil pada Tabel 4 dan Tabel 5.

Berdasarkan hasil pengujian, didapatkan bahwa kombinasi yang optimal adalah jumlah *mapper* adalah 2/3 dari jumlah total *core* CPU, sedangkan untuk jumlah *reducer* berjumlah 1/3 dari jumlah total *core* CPU.

5.3 Aplikasi *Classification Tree* antara Jenis OS Device dengan Kode Negara

Data yang digunakan adalah Umur dan Jenis OS *Device*. Besar data yang digunakan untuk kedua aplikasi adalah yaitu 1.105 GB.

5.3.1 Pengujian dengan R

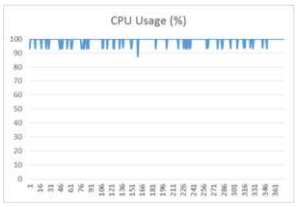
Dari hasil pengujian didapatkan bahwa secara rata – rata aplikasi R menggunakan 98.91 % dari total CPU. Untuk penggunaan memori rata – rata menggunakan 17.46 %. Total rata – rata waktu yang dibutuhkan untuk menyelesaikan aplikasi tersebut adalah 64.3 detik. Jumlah sampel per detik dari aplikasi ini adalah rata – rata 5.851 sampel per detik.

Tabel 4 Top 5 Konfigurasi dengan kombinasi *mapper* dan *reducer* untuk 6 *node*

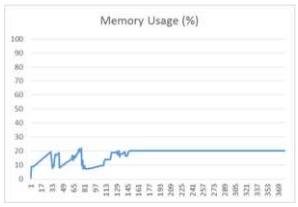
Konfigurasi	256 MB			
	CPU	Memori	Waktu	Total Waktu (YARN)
20-4	11.338	11.68	463.23	456.81
17-9	11.94	14.21	465.25	458.47
15-7	12.007	13.75	472.31	465.92
15-11	12.1	14.73	489.74	483.3
14-10	11.198	14.35	515.63	509.14
Konfigurasi	**512 MB**			
	CPU	**Memori**	**Waktu**	**Total Waktu (YARN)**
20-4	10.91	18.86	497.37	466.14
17-9	9.97	16.62	503.86	498.39
15-7	11.13	16.04	640.37	634.56
15-11	10.44	17.76	542.97	537.88
14-10	10.91	18.86	497.37	398.25

Tabel 5 Top 5 Konfigurasi dengan kombinasi *mapper* dan *reducer* untuk 8 *node*

Konfigurasi	256 MB			
	CPU	Memori	Waktu	Total Waktu (YARN)
21-9	11.91	11.76	371.98	366.13
20-12	12.39	12.36	373.4	367.64
20-10	11.65	12.47	377.63	372
16-16	11.24	10.68	417.18	395.28
16-5	8.08	9.7	408.93	403.05
Konfigurasi	**512 MB**			
	CPU	**Memori**	**Waktu**	**Total Waktu (YARN)**
21-9	9.76	14.93	408.03	402.41
20-12	8.47	19.28	468.53	461.88
20-10	9.16	14.99	438.47	432.87
16-16	10	16.2	423.98	418.63
16-5	8.16	12.11	480.85	475.06



Gambar 1 Grafik Penggunaan CPU



Gambar 2 Grafik Penggunaan Memori

dilihat bahwa penggunaan CPU maksimum pada aplikasi adalah sebesar 100% dengan jumlah sampel rata – rata 5.851 sampel per detik. Pada Gambar 16 dapat dilihat bahwa penggunaan memori maksimum pada aplikasi adalah sebesar 20% dengan jumlah sampel rata – rata 5.851 sampel per detik.

5.4 Aplikasi *Classification Tree* antara Jenis OS Device dengan Umur

5.4.1 Pengujian dengan R

Untuk pencarian pohon klasifikasi antara kode negara dengan Jenis OS Device, aplikasi R tidak dapat memproses data ke dalam bentuk pohon klasifikasi dikarenakan ukuran data yang besar.

6. KESIMPULAN

Setelah dilakukan perancangan sistem, pengimplementasian, dan pengujian terhadap aplikasi yang telah dibuat, dapat ditarik kesimpulan sebagai berikut:

1. Berdasarkan hasil pengujian, didapatkan bahwa dari data Yahoo:
   a. Korelasi antara jenis kelamin dengan jenis page yang dibuka, umur dan frekuensi komunikasi memiliki tingkat yang rendah.
   b. Pohon klasifikasi yang dihasilkan dari perhitungan CART untuk jenis OS dengan umur adalah jika umur pengguna di antara 28 – 32 tahun maka mereka menggunakan Symbian, sedangkan apabila pengguna >= 32 maka menggunakan PDA, jika pengguna < 28 maka menggunakan Symbian.
   c. Pohon klasifikasi yang dihasilkan dari perhitungan CART untuk jenis OS dengan umur adalah jika kode negaar = 143 maka banyak yang menggunakan Windows selain itu Symbian.
2. Berdasarkan hasil pengujian, aplikasi menggunakan non Hadoop (R) menggunakan CPU dan memori yang lebih besar dibandingkan dengan menggunakan Hadoop.
3. Berdasarkan hasil pengujian, untuk menghasilkan hasil eksekusi yang optimal maka jumlah *mapper* yang digunakan adalah sekitar 2/3 dari jumlah total *core* CPU dan jumlah *reducer* yang digunakan adalah 1/3 dari jumlah total *core* CPU.
4. Berdasarkan hasil pengujian, lama waktu eksekusi *job* Hadoop dipengaruhi oleh faktor banyaknya *job* yang dijalankan. Semakin banyak *job* yang dijalankan waktu eksekusi semakin lambat.
5. Berdasarkan hasil pengujian, semakin banyak *node* yang digunakan untuk mengolah data, maka waktu eksekusi *job* pada Hadoop semakin pendek.
6. R lebih cocok digunakan untuk perhitungan data dalam jumlah sedikit, sedangkan Hadoop tidak cocok untuk pengolahan data dalam jumlah akan tetapi bagus dalam pengolahan data dalam ukuan yang besar.

Saran untuk pengembangan kedepannnya adalah:

- Modul korelasi dan *classification tree* dapat dikembangkan kedepannya untuk menyongkong Data Analytics untuk *Business Intelligence* pada Hadoop.
- Untuk kedepannya dapat digunakan aplikasi lain yang berjalan di atas Hadoop seperti Hive, Pig, Spark, dan HBase, untuk pembuatan modul analisa data guna menyongkong *Business Intelligence*
- Untuk kedepannya dapat digunakan oozie untuk menjalankan berbagai macam *job* secara berurutan. Dengan oozie *job* yang dijalankan tidak hanya *mapreduce* Hadoop biasa tetapi juga *job* aplikasi lain seperti Hive, Pig, Spark, dan HBase.

7. DAFTAR PUSTAKA

[1] Aggarwal, A. 2015. Managing Big Data Integration in the Public Sector. IGI Global.

[2] Apache. 2016. Apache Hadoop 2.7.2 @2013; HDFS Architecture. URI = https://Hadoop.apache.org/docs/r2.7.2/Hadoop-project-dist/Hadoop-hdfs/HdfsDesign.html

[3] Gravetter, F. J., & Wallnau, L. B. 2013. Statistics for the Behavioral Sciences. Canada: Jon-David Hague.

[4] Harshawardhan S. Bhosale, P. D. 2014. A Review Paper on Big Data and Hadoop. International Journal of Scientific and Research Publications, 4(10), 1-7.

[5] Kshemkalyani, A. D., & Singhal, M. 2011. Distributed Computing: Principles, Algorithms, and Systems. New York: Cambridge University Press.

[6] Leszek, R., Maciej, J., & Lena, P. 2014. Classification and Regression Trees (CART) Theory and Applications. Information Sciences, 266, 1-15.

[7] Loh, W.-Y. 2011. Classification and regression trees. WIREs Data Mining and Knowledge Discovery, 14-23.

[8] Lublinsky, B., Smith, K. T., & Yakubovich, A. 2013. Professional Hadoop® Solutions. Indianapolis: John Wiley & Sons, Inc.

[9] Maitreya, S., & Jhab, C. 2015. MapReduce: Simplified Data Analysis of Big. Procedia Computer Science 57, 563 – 571.

[10] Marr, B. 2015. Why only one of the 5 Vs of big data really matters | IBM Big Data Analytics and Hub. URI=http://www.ibmbigdatahub.com/blog/why-only-one-5-vs-big-data-really-matters

[11] Turkington, G. 2013. Hadoop Beginner's Guide. Birmingham: Packt Publishing Ltd.

[12] Yahoo! About Us | research.yahoo.com. URI = https://research.yahoo.com