Komputika: Jurnal Sistem Komputer
Volume 9, Nomor 1, April 2020, hlm. 43 - 50
Terakreditasi Peringkat 3, SK No. 28/E/KPT/2019
DOI: **10.34010/komputika.v9i1.2848**

ISSN: 2252-9039 (print)
ISSN: 2655-3198 (online)

Studi Perbandingan Performa Algoritma Penjadwalan untuk Real Time Data Twitter pada Hadoop

Sidik Prabowo[1*], Maman Abdurohman[2]

[1,2] Program Studi Informatika, Fakultas Informatika, Universitas Telkom
Jl. Telekomunikasi 1, Bandung, Indonesia 40257
**email: pakwowo@telkomuniversity.ac.id*

(Naskah masuk: *20 Februari 2020*; diterima untuk diterbitkan: *18 Maret 2020*)

ABSTRAK *–Hadoop merupakan sebuah framework software yang bersifat open source dan berbasis java. Hadoop terdiri atas dua komponen utama, yaitu MapReduce dan Hadoop Distributed File System (HDFS). MapReduce terdiri atas Map dan Reduce yang digunakan untuk pemrosesan data, sementara HDFS adalah tempat atau direktori dimana data hadoop dapat disimpan. Dalam menjalankan job yang tidak jarang terdapat keragaman karakteristik eksekusinya, diperlukan job scheduler yang tepat. Terdapat banyak job scheduler yang dapat di pilih supaya sesuai dengan karakteristik job. Fair Scheduler merupakan salah satu scheduler yang bekerja dengan cara memastikan suatu jobs akan mendapatkan resource yang sama dengan jobs yang lain, dengan tujuan meningkatkan performa dari segi Average Completion Time. Hadoop Fair Sojourn Protocol Scheduler adalah sebuah algoritma scheduling dalam Hadoop yang dapat melakukan scheduling berdasarkan ukuran jobs yang diberikan. Penelitian ini bertujuan untuk melihat perbandingan performa kedua scheduler tersebut untuk karakteristik data twitter. Hasil pengujian menunjukkan Hadoop Fair Sojourn Protocol Scheduler memiliki performansi lebih baik dibandingkan Fair Scheduler baik dari penanganan average completion time sebesar 9,31% dan job throughput sebesar 23,46%. Kemudian untuk Fair Scheduler unggul dalam parameter task fail rate sebesar 23,98%.*

Kata Kunci *– Hadoop, Scheduler, FS, HFSPS, Real-time Twitter*

Comparison Study of Scheduling Algorithm Performance for Real Time Data Twitter in Hadoop

ABSTRACT *– Hadoop is open source and java based software framework. Hadoop consists of two main components, namely MapReduce and Hadoop Distributed File System (HDFS). MapReduce consists of Map and Reduce which is used for data processing, while HDFS is a place or directory where Hadoop data can be stored. In carrying out a job that is not uncommonly diverse in its execution characteristics, a proper job scheduler is needed. There are many job schedulers that can be selected to match job characteristics. Fair Scheduler uses a scheduler where the principle ensures that jobs will get the same resources as other jobs, with the aim of improving performance in terms of Average Completion Time. Hadoop Fair Sojourn Protocol Scheduler is a scheduling algorithm in Hadoop that can do scheduling based on the size of jobs provided. This study aims to compare the performance of the two schedulers for Twitter data characteristics. The test results show the Hadoop Fair Sojourn Protocol Scheduler has a better performance than the Fair Scheduler both from handling average completion time of 9.31% and job throughput of 23.46%. Then the Fair Scheduler excels in the task fail rate parameter of 23.98%.*

Keywords *- Hadoop, Scheduler, FS, HFSPS, Real-time Twitter.*

1. PENDAHULUAN

Twitter adalah salah satu media sosial yang menerima jutaan tweet setiap harinya dan diperkirakan menghasilkan data mentah sebesar 1 zettabyte setiap tahunnya. Data mentah yang sangat besar ini dihasilkan dari tiap obrolan (*tweet*) yang kemudian dapat digunakan untuk kepentingan industri, sosial, ekonomi, pemerintahan, dan tujuan lainnya. Salah satu cara konvensional untuk mengolah *Big Data* dari twitter tersebut adalah menggunakan Apache Hadoop[1].

Hadoop merupakan sebuah framework *software* berbasis java dan mempunyai sifat tidak berbayar

(*open-source*) yang banyak digunakan untuk pengolahan data dalam kuantitas yang besar dengan cara kerja terdistribusi[2]. Terdapat 2 arsitektur dasar yang membangun hadoop yaitu HDFS dan MapReduce[3]. Algoritma *job scheduller* yang terdapat dalam MapReduce berguna untuk mengatur setiap job yang berjalan pada sistem Hadoop. Job yang diolah pada *job scheduler* dapat bersifat *heterogen* (berbeda) maupun *homogen* (sejenis).

Hadoop Fair Sojourn Protocol Scheduler (HFSPS) dan *Fair Scheduler* merupakan job scheduler yang sudah ada dan berkembang pada sistem Hadoop, dan memiliki keunikan masing-masing dalam menangani karakteristik job yang diproses. HFSPS menerapkan metode *scheduling* berdasarkan ukuran penjadwalan dengan membangun sebuah *knowledge* (pengetahuan) mengenai informasi dari ukuran penjadwalan dari sebuah job selama eksekusi sehingga dapat mengoptimalkan *completion time*[4]. *Fair Scheduler* menerapkan sistem eksekusi *map* dan *reduces* secara bersamaan[5].

Pada penelitian ini membahas tentang performa HFSPS dan *Fair Scheduler* sebagai scheduler yang akan dipakai pada sistem Hadoop untuk real-time data dari twitter. Dalam melihat performansi nya akan diamati parameter Task Fail Rate, Job Throughput, dan Average Completion Time[6].

Tujuan dari penelitian yang ingin dicapai adalah dapat mengetahui perbandingan performansi antara *Fair Scheduler* dan *Hadoop Fair Sojourn Protocol Scheduler* (HFSP) pada sistem Hadoop khususnya dalam menangani real time data pada Twitter Batasan masalah dalam penelitian ini adalah server berjumlah tiga berbentuk virtual yang menjadi server utama pada Hadoop dan berbasis Linux Ubuntu 14.04, user berjumlah empat yang akan mengakses job, penggunaan algoritma penjadwalan hanya menggunakan *Fair Scheduler* dan HFSP, menggunakan data yang bersifat real time yang diperoleh dari streaming data Twitter menggunakan Apache Flume, menggunakan parameter *Task Fail Rate, Job Throughput,* dan *Average Completion Time*.

2. METODE DAN BAHAN

2.1. Studi Terkait

Penelitian [4] dan [7] berkaitan erat dengan penelitian ini. Pada penelitian tersebut telah dilakukan pengujian algoritma *Fair Scheduler* dan HFSPS, namun terdapat perbedaan dari segi metode pengujian, pada penelitian tersebut kedua algoritma digunakan untuk mengolah data dump. Sedangkan pada penelitian ini dilakukan pengujian algoritma *Fair Scheduler* dan HFSPS untuk digunakan pada data real time yang didapat melalui *twitter*. *Real time* data merupakan sebuah data yang tidak disimpan melainkan bahwa data tersebut diteruskan dari suatu *user* ke *user* lainnya.

Hal yang perlu diperhatikan pada *real time* data adalah bahwa data yang sedang diteruskan tidak sepenuhnya sampai ke pengguna secara langsung, mungkin ada beberapa hambatan yang terkait dengan infrastruktur pengumpulan data, *bandwidth* antara beberapa pihak, atau lainnya[8]. Selain itu, pengembangan yang penulis lakukan pada penelitian sebelumnya ialah pemanfaatan *Apache flume.* Pada penelitian sebelumnya pengujian algoritma pada karakteristik *job* tertentu hanya menggunakan arsitektur Hadoop, pada pengujian kali ini arsitektur Hadoop akan digabung dengan arsitektur flume

2.2. Hadoop System

Hadoop memiliki dua lapisan utama yang digunakan dalam melakukan proses pengolahan data, lapisan itu diantaranya[3]:

a. MapReduce

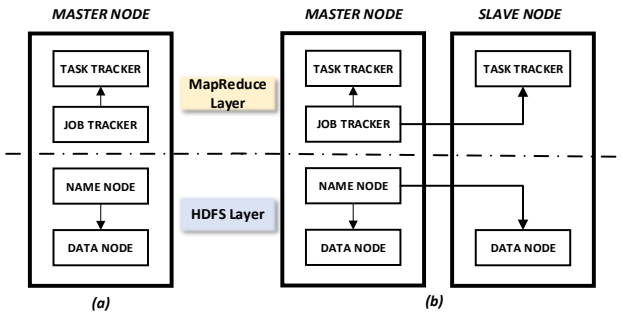
Lapisan pada Hadoop yang memiliki dua proses utama yaitu proses *Map* dan *Reduce.* MapReduce berguna untuk memproses setiap *job* yang masuk ke dalam master. Dalam proses ini diperlukan job *scheduler* yang berguna untuk memanajemen penjadwalan dari eksekusi *job.* Terdapat beberapa algoritma yang diciptakan untuk menjamin bahwa proses ini berjalan secara akurat dan cepat[9].

b. *Hadoop File Distributed System* (HDFS)

Lapisan HDFS merupakan sebuah tempat atau direktori dimana data hadoop dapat disimpan. Lapisan ini memiliki dua komponen utama yaitu Name node yang bertugas sebagai node utama guna mengatur penempatan data pada *cluster*, dan data node yang merupakan tempat dimana data ditempatkan dalam *cluster* [10].

2.3. Hadoop Single-Node dan Multi-Node

Hadoop *single-node* hanya memiliki satu server yang bekerja menjadi master tetapi tidak bekerja sebagai slave. Pada server hadoop *single-node* semua proses dilakukan dalam satu server[15].



Gambar 1. Arsitektur Hadoop Single node (a) dan Multinode (b)

Hadoop *multi-node* merupakan gabungan 2 server single-node yaitu 1 untuk server master dan 1 untuk server slave. Server master dapat bekerja juga sebagai server slave dan server slave hanya bekerja sebagai server slave. Gambar 1 menunjukkan arsitektur Hadoop *single node* dan *multi node*.

2.4. Flume

Flume merupakan sebuah layanan yang diberikan oleh Apache untuk mengumpulkan, menggabungkan, dan memindahkan sejumlah data streaming langsung ke dalam directory HDFS. Layanan yang diberikan oleh Apache bersifat gratis (open source). Apache Flume juga dapat digunakan untuk melakukan penarikan real time data pada twitter yang kemudian digunakan dalam pengujian job pada Hadoop[7].

2.5. Job

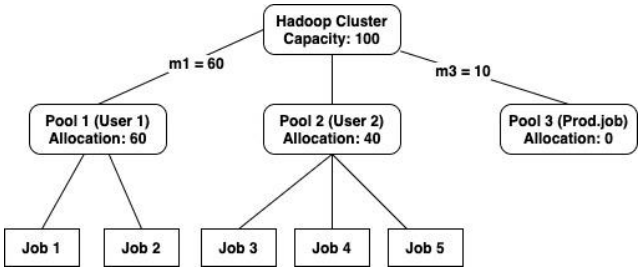
Job merupakan suatu pekerjaan yang diberikan kepada Hadoop dan dapat dikerjakan oleh Hadoop. *Job* dalam kategorisasinya dapat dikelompokkan menjadi homogen (*job* yang sama) atau dilakukan secara bersamaan antar *job* yang homogen (*job* yang berbeda)[14].

2.6. Algoritma Fair Scheduler

Fair Scheduler menggunakan sebuah metode yang menentukan suatu *jobs* akan mendapatkan resource yang sama dengan *jobs* yang lain. Maka setiap pekerjaan akan mendapatkan resource yang sama. Dengan demikian tiap tiap *jobs* tidak harus menunggu lama pada resource yang sedang digunakan pada *jobs* yang lain[14]. Gambar 2 dan gambar 3 menunjukkan pembagian pool pada *Fair Scheduler*.

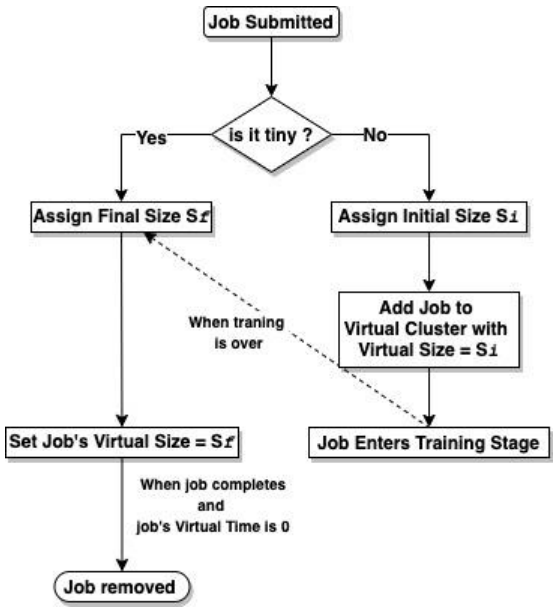
2.7. Algoritma HFSPS

HFSPS adalah sebuah algoritma *scheduling* dalam Hadoop yang dapat melakukan scheduling berdasarkan ukuran *jobs* yang diberikan. HFSPS sendiri merupakan sebuah pengembangan dari Hadoop *Fair Scheduler*.



Gambar 2. Pembagian pool pada *Fair Scheduler* [11]

Algoritma HFSPS ini dapat mengalokasikan sebuah cluster untuk memberikan info mengenai besarnya *jobs* yang akan dikerjakan. Semakin kecil ukuran sebuah *jobs*, maka antrian *jobs* itu akan diselesaikan terlebih dahulu dengan menggunakan resource secukupnya, jadi pada saat ini resource yang tidak terpakai akan dialokasikan untuk menghitung informasi *jobs* selanjutnya yang akan dikerjakan.



Gambar 3. Pembagian pool pada *Fair Scheduler* [11]

Pada HFSPS semua resource dapat dipakai untuk mengerjakan suatu *jobs* [4].

2.8. Parameter Pengujian

Parameter yang digunakan yaitu *Task Fail Rate, Job Throughput,* dan *Average Completion Time*. Rincian parameter adalah sebagai berikut:

a. Task fail rate

Pada parameter ini diukur jumlah dari *job* yang gagal dan telah diulang kembali pekerjaannya karena kesalahan dari resource pada server Hadoop. Satuan yang dipakai pada parameter ini adalah *job fail* per jumlah / total job[14].

b. Job Throughput

Pada parameter ini diukur jumlah job yang telah berjalan dan berhasil dalam satuan waktu[8]. Nilai yang dihasilkan diperoleh dari awal suatu job itu berjalan hingga suatu job tersebut selesai dan dibagi jumlah menit yang dibutuhkan untuk menyelesaikan job tersebut. Satuan pada parameter ini adalah job per menit[14].

c. Average Completion Time

Parameter ini diukur rata-rata waktu yang dibutuhkan dari suatu job. Nilai ini diperoleh dari jumlah keseluruhan waktu yang dibutuhkan untuk menyelesaikan suatu job dibagi dengan jumlah keseluruhan job yang masuk pada satu skenario.

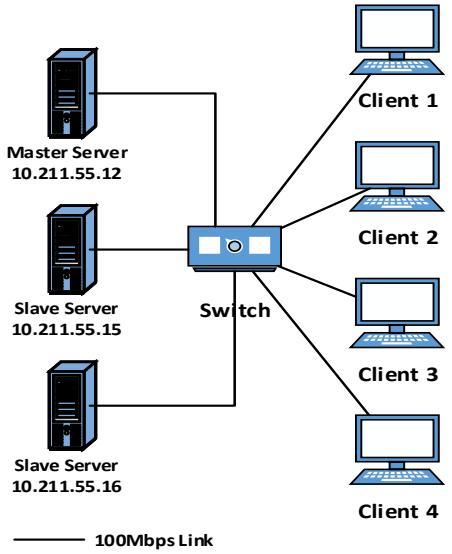
Satuan yang dipakai pada parameter ini adalah menit[14].

d. I/O

Parameter ini mengukur kecepatan i/o (baca/tulis) pada cluster hadoop. Parameter ini dapat dihasilkan dengan cara memberikan *job* pada MapReduce untuk membaca dan menulis file secara paralel pada Hadoop system yang telah dibuat.[12].

2.9. Perancangan sistem

Pada Penelitian ini eksperimen dilakukan di atas arsitektur *MultiNode* seperti pada gambar 4.



Gambar 4. Topologi Penelitian

Dalam lingkungan uji coba, terdiri atas tiga buah server, dimana salah satunya berfungsi sebagai *Master* node dan yang lainnya berfungsi sebagai *Slave Node*. Dalam konfigurasi ini *Master* dapat berfungsi sebagai *Master server* dan juga dapat bertindak sebagai *slave server,* sementara *Slave* hanya dapat bertindak sebagai *slave* saja. Fungsi *client* adalah mengirimkan *job* yang selanjutnya dieksekusi oleh *server.* Jenis *job* yang dikerjakan dalam penelitian ini adalah seperti pada tabel 1 berikut.

Tabel 1 Karakteristik Job

Jenis job	Karakteristik Job
Wordcount	Menghitung jumlah kata yang unik dalam sebuah data text berukuran besar. Output dari *job* ini adalah daftar kata beserta jumlah kemunculan kata tersebut dalam sebuah file.
Grep	Mencari kata yang ditentukan oleh user, sehingga data hasil dari grep hanya kata yang dicari user saja.

Skenario yang dijalankan terdiri atas 4 skenario job utama, yang kemudian dikombinasikan berdasarkan karakteristik job yang berbeda. Hal ini ditujukan untuk melihat pengaruh scheduler terhadap tipe job yang homogen maupun heterogen. Tabel 2 menunjukkan kombinasi job yang digunakan dalam penelitian ini[14].

Tabel 2 Skenario Job

Skenario	*Job*	Jenis *Job*	*Jobs (FS * HFSPS)*			
1	Wordcount	Homogen	10	20	30	50
2	Grep		10	20	30	50
3	Wordcount (Spesific Case)		10	20	30	50
4	Grep (Spesific Case)		10	20	30	50
5	Wordcount-Grep	Heterogen	10	20	30	50
6	Wordcount-Grep (Spesific Case)		10	20	30	50

Resource data dalam pengujian ini diambil dari hasil *streaming* data yang dilakukan oleh Apache Flume terhadap server twitter sehingga data yang diperoleh bersifat *real time*. Pada pengujian skenario 3, 4, dan 6 *(Wordcount Spesific Case dan Grep Spesific Case)* data twitter yang distream bersifat spesifik *(keyword search: Prabowo, Jokowi, capres, cawapres, 2019gantipresiden, cebong)*, kemudian file *jar* yang digunakan untuk *job wordcount* dan *grep* memiliki spesifikasi hanya mencari dan menghitung huruf dengan *keyword search:* Prabowo, Jokowi, capres, cawapres, 2019gantipresiden, cebong. Pengambilan data dilakukan pada periode juni-agustus 2018. Tujuan pengambilan data yang bersifat real time merupakan bentuk pengembangan dari studi kasus yang belum pernah dilakukan sebelumnya. Tabel 3 menunjukkan resource data.

Tabel 3 Resource Data

No	Name	*Size*	*Interval Time (Minute)*
1	FlumeData.1518089727686	1.47GB	205
2	FlumeData.1518100261589	2.07GB	240
3	FlumeData.1518118630126	3.37GB	300
4	FlumeData.1518128630868	3.30GB	300
5	FlumeData.1534196127183	1.64GB	1380

3. HASIL DAN PEMBAHASAN

Dari percobaan yang telah dilaksanakan, perbandingan performa 2 algoritma penjadwalan yang diimplementasikan pada hadoop, akan dibandingkan menjadi 7 skenario yang telah ditentukan sebelumnya.

3.1. Skenario 1 Homogen job (Wordcount)

Skenario 1 ini dilakukan pengujian terhadap satu tipe *job* yaitu wordcount terhadap algoritma.

Fair Scheduler dan HFSPS. Pada skenario ini jumlah job berpengaruh terhadap performasi kedua algoritma dengan dilihat dari nilai *average completion time, task fail rate*, dan *job throughput*. Resource data yang dipakai dalam pengaksesan file disetiap job berasal dari real time data twitter variasi ukuran yaitu 1.47 GB, 2.07 GB, 3.37 GB, dan 3.30 GB serta variasi jumlah job 10, 15, 20, 25, 30.

Pada skenario 1 dilakukan pengujian terhadap algoritma Fair dan HFSP, total nilai pada parameter average completion time dengan algoritma Fair menghasilkan nilai sebesar 1339,32 menit dan HFSP sebesar 648,23 menit dengan demikian algoritma HFSP lebih unggul sebesar 691,09 menit pada parameter average completion time. Total nilai pada parameter job throughput dengan algoritma Fair menghasilkan nilai sebesar 13,9 job/menit dan HFSP sebesar 29,99 job/menit dengan demikian algoritma HFSP lebih unggul sebesar 16,09 job/menit pada parameter job throughput. Total nilai pada parameter Task Fail Rate dengan algoritma Fair menghasilkan nilai sebesar 8,98% dan HFSP sebesar 6,61% dengan demikian algoritma HFSP lebih unggul sebesar 2,37% pada parameter task fail rate. Dalam melakukan kecepatan baca tulis pada cluster (I/O stats) algoritma HFSP unggul sebesar 6.44 mb/s dibandingkan algoritma Fair.

3.2. Skenario 2 Homogen job (Grep)

Pada Skenario 2 ini dilakukan pengujian terhadap satu jenis job yaitu job grep menggunakan algoritma *Fair Scheduler* dan Hadoop Fair Sojourn Protocol Scheduler. Pada skenario ini jumlah job berpengaruh terhadap performasi kedua algoritma dengan dilihat dari nilai average completion time, task fail rate, dan job throughput. Resource data yang dipakai dalam pengaksesan file disetiap job berasal dari data twitter yang bersifat real time dengan ukuran yang beragam yaitu 1.47 GB, 2.07 GB, 3.37 GB, dan 3.30GB. Skenario ini bercirikan dengan jenis job yang bersifat homogen dan mempunyai resource data yang beragam dengan jumlah job 10 *jobs*, 15 *jobs*, 20 *jobs*, 25 *jobs*, 30 *jobs*.

Terlihat pada hambar 6 bahwa hasil pengujian untuk scenario 2 pada algoritma Fair dan HFSP, total nilai pada parameter average completion time dengan algoritma Fair menghasilkan nilai sebesar 612,63 menit dan HFSP sebesar 193,19 menit dengan demikian algoritma HFSP lebih unggul sebesar 419,44 menit pada parameter average completion time. Total nilai pada parameter job throughput dengan algoritma Fair menghasilkan nilai sebesar 30,37 job/menit dan HFSP sebesar 96,95 job/menit dengan demikian algoritma H FSP lebih unggul sebesar 66,58 job/menit pada parameter job throughput. Total nilai pada parameter Task Fail Rate dengan algoritma Fair menghasilkan nilai sebesar 6,73% dan HFSP sebesar 9,24% dengan demikian algoritma Fair lebih unggul sebesar 2,51% pada parameter task fail rate. Dalam melakukan kecepatan baca tulis pada cluster (I/O stats) algoritma HFSP unggul sebesar 6.44 mb/s dibandingkan algoritma Fair.

3.3. Skenario 3 Homogen job (Wordcount-SpecificCase)

Pada Skenario 3 ini dilakukan pengujian terhadap satu jenis job yaitu job wordcount yang bersifat spesifik (keyword count: Prabowo, Jokowi, capres, cawapres, 2019gantipresiden, cebong) menggunakan algoritma *Fair Scheduler* dan *Hadoop Fair Sojourn Protocol Scheduler*. Pada skenario ini jumlah job berpengaruh terhadap performasi kedua algoritma dengan dilihat dari nilai *average completion time*, *task fail rate*, dan *job throughput*. Resource data yang dipakai dalam pengaksesan file disetiap job berasal dari data *twitter* yang distream secara spesifik (*keyword search*: Prabowo, Jokowi, capres, cawapres, 2019gantipresiden, cebong) dengan ukuran 1.64 GB. Skenario ini bercirikan dengan jenis job yang bersifat homogen dan mempunyai resource data yang beragam dengan jumlah *job* 10 *jobs*, 15 *jobs*, 20 *jobs*, 25 *jobs*, 30 *jobs*.

Pada skenario 3, bahwa total nilai pada parameter average completion time dengan algoritma Fair menghasilkan nilai sebesar 680,7 menit dan HFSP sebesar 505,14 menit dengan demikian algoritma *HFSP* lebih unggul sebesar 175,56 menit pada parameter *average completion time*. Total nilai pada parameter *job throughput* dengan algoritma *Fair* menghasilkan nilai sebesar 17,62 *job*/menit dan *HFSP* sebesar 23,73 *job*/menit dengan demikian algoritma *HFSP* lebih unggul sebesar 6,11 *job*/menit pada parameter *job throughput*. Total nilai pada parameter *Task Fail Rate* dengan algoritma *Fair* menghasilkan nilai sebesar 4,75% dan *HFSP* sebesar 10,41% dengan demikian algoritma *Fair* lebih unggul sebesar 5,66% pada parameter task fail rate. Dalam melakukan kecepatan baca tulis pada cluster (I/O stats) algoritma HFSP unggul sebesar 6.44 mb/s dibandingkan algoritma Fair.

3.4. Skenario 4 Homogen job (Grep-Specific Case)

Pada Skenario 4 ini dilakukan pengujian terhadap satu jenis job yaitu job grep yang bersifat spesifik (keyword search: Prabowo, Jokowi, capres, cawapres, 2019gantipresiden, cebong) menggunakan algoritma *Fair Scheduler* dan Hadoop Fair Sojourn Protocol Scheduler. Pada skenario ini jumlah job berpengaruh terhadap performasi kedua algoritma dengan dilihat dari nilai average completion time, task fail rate, dan job throughput. Resource data yang dipakai dalam pengaksesan file disetiap job berasal dari data twitter yang distream secara spesifik (keyword search: Prabowo, Jokowi, capres, cawapres, 2019gantipresiden, cebong) dengan ukuran 1.64 GB. Skenario ini bercirikan dengan jenis job yang bersifat homogen dan mempunyai resource data yang beragam dengan jumlah job 10 *jobs*, 15 *jobs*, 20 *jobs*, 25 *jobs*, 30 *jobs*. Pada skenario 4 dilakukan pengujian terhadap algoritma Fair dan HFSP, total nilai pada parameter average completion time dengan algoritma Fair menghasilkan nilai sebesar 116,66 menit dan HFSP sebesar 87,05 menit dengan demikian algoritma HFSP lebih unggul sebesar 29,61 menit pada parameter average completion time. Total nilai pada parameter job throughput dengan algoritma Fair menghasilkan nilai sebesar 102,81 job/menit dan HFSP sebesar 137,58 job/menit dengan demikian algoritma HFSP lebih unggul sebesar 34,77 job/menit pada parameter job throughput. Total nilai pada parameter Task Fail Rate dengan algoritma Fair menghasilkan nilai sebesar 1,61% dan HFSP sebesar 4,33% dengan demikian algoritma Fair lebih unggul sebesar 2,72% pada parameter task fail rate. Dalam melakukan kecepatan baca tulis pada cluster (I/O stats) algoritma HFSP unggul sebesar 6.44 mb/s dibandingkan algoritma Fair.

3.5. Skenario 5 Heterogen job (Wordcount-Grep)

Pada Skenario 5 ini dilakukan pengujian terhadap dua jenis job yaitu job wordcount dan job grep menggunakan algoritma *Fair Scheduler* dan Hadoop Fair Sojourn Protocol Scheduler. Pada skenario ini jumlah job berpengaruh terhadap performasi kedua algoritma dengan dilihat dari nilai average completion time, task fail rate, dan job throughput. Resource data yang dipakai dalam pengaksesan file disetiap job berasal dari data twitter yang bersifat real time dengan ukuran yang beragam yaitu 1.47 GB, 2.07 GB, 3.37 GB, dan 3.30GB. Skenario ini bercirikan dengan jenis job yang bersifat heterogen dan mempunyai resource data yang beragam dengan jumlah job 10 *jobs*, 15 *jobs*, 20 *jobs*, 25 *jobs*, 30 *jobs*.

Pada skenario 5 dilakukan pengujian terhadap algoritma Fair dan HFSP, total nilai pada parameter average completion time dengan algoritma Fair menghasilkan nilai sebesar 1018,5 menit dan HFSP sebesar 347,42 menit dengan demikian algoritma HFSP lebih unggul sebesar 671,08 menit pada parameter average completion time. Total nilai pada parameter job throughput dengan algoritma Fair menghasilkan nilai sebesar 18,37 job/menit dan HFSP sebesar 53,95 job/menit dengan demikian algoritma HFSP lebih unggul sebesar 35,58 job/menit pada parameter job throughput. Total nilai pada parameter Task Fail Rate dengan algoritma Fair menghasilkan nilai sebesar 5,68% dan HFSP sebesar 6,91% dengan demikian algoritma Fair lebih unggul sebesar 1,23% pada parameter task fail rate. Dalam melakukan kecepatan baca tulis pada cluster (I/O stats) algoritma HFSP unggul sebesar 6.44 mb/s dibandingkan algoritma Fair.

3.6. Skenario 6 Heterogen job (Wordcount-Specific Case dan Grep-Specific Case)

Pada Skenario 6 ini dilakukan pengujian terhadap dua jenis job yaitu job wordcount dan grep yang bersifat spesifik (keyword search, count: Prabowo, Jokowi, capres, cawapres, 2019gantipresiden, cebong) menggunakan algoritma *Fair Scheduler* dan Hadoop Fair Sojourn Protocol Scheduler. Pada skenario ini jumlah job berpengaruh terhadap performasi kedua algoritma dengan dilihat dari nilai average completion time, task fail rate, dan job throughput. Resource data yang dipakai dalam pengaksesan file disetiap job berasal dari data twitter yang distream secara spesifik (keywor d search: Prabowo, Jokowi, capres, cawapres, 2019gantipresiden, cebong) dengan ukuran 1.64 GB. Skenario ini bercirikan dengan jenis job yang bersifat homogen dan mempunyai resource data yang beragam dengan jumlah job 10 *jobs*, 15 *jobs*, 20 *jobs*, 25 *jobs*, 30 *jobs*. Pada skenario 6 dilakukan pengujian terhadap algoritma Fair dan HFSP, total nilai pada parameter average completion time dengan algoritma Fair menghasilkan nilai sebesar 406,93 menit dan HFSP sebesar 303,39 menit dengan demikian algoritma HFSP lebih unggul sebesar 103,54 menit pada parameter average completion time. Total nilai pada parameter job throughput dengan algoritma Fair menghasilkan nilai sebesar 29,44 job/menit dan HFSP sebesar 39,48 job/menit dengan demikian algoritma HFSP lebih unggul sebesar 10,04 job/menit pada parameter job throughput. Total nilai pada parameter Task Fail Rate dengan algoritma Fair menghasilkan nilai sebesar 4,78% dan HFSP sebesar 8,5% dengan demikian algoritma Fair lebih unggul sebesar 3,72% pada parameter task fail rate. Dalam melakukan kecepatan baca tulis pada cluster (I/O stats) algoritma HFSP unggul sebesar 6.44 mb/s dibandingkan algoritma Fair. Gambar 10 menunjukkan hasil pengujian skenario 6.

Dari percobaan yang telah dilaksanakan, perbandingan performa 2 algoritma penjadwalan yang diimplementasikan diatas hadoop, akan dibandingkan kedalam 3 skenario yang telah ditentukan sebelumnya, berikut adalah hasil pengujiannya :

Tabel 4 Rangkuman Hasil Skenario Pengujian

Skenario	*Average Completion Time*	*Job Throughput*	*Task Fail Rate*
Skenario 1	HFSP	HFSP	HFSP
Skenario 2	HFSP	HFSP	FAIR
Skenario 3	HFSP	HFSP	FAIR
Skenario 4	HFSP	HFSP	FAIR
Skenario 5	HFSP	HFSP	FAIR
Skenario 6	HFSP	HFSP	FAIR

Tabel 4 berisikan rangkuman hasil eksperimen job scheduling dari seluruh skenario yang menunjukkan algoritma yang lebih baik dalam setiap skenario dan jumlah job antara Fair Scheduling dan HFSPS sesuai dengan parameter yang digunakan.

Berdasarkan tabel 4 HFSPS unggul untuk parameter *average completion time* dan *job throughput*. Pada parameter *task fail rate,* algoritma Fair mengungguli algoritma HFSP. Hal ini dapat terjadi karena pengaloksian prioritas berdasarkan ukuran job membuat Hadoop Fair Sojourn Protocol Scheduler unggul dibandingkan *Fair Scheduler*, algoritma HFSP memiliki sifat mengerjakan job berdasarkan ukuran terkecil sehingga cenderung mempercepat eksekusi tiap job agar menghasilkan waktu eksekusi job menjadi cepat, hanya saja waktu eksekusi yang cepat mengorbankan task fail menjadi lebih tinggi dari biasanya hal ini disebabkan oleh waktu pengerjaan setiap task bisa mengalami timeout dan Hadoop menganggap task tersebut terjadi fail dan akan langsung mengeksekusi task selanjutnya.

Semakin tinggi nilai yang dihasilkan oleh sebuah algoritma pada parameter *job throughput*, maka algoritma tersebut bekerja dengan optimal, karena dapat meminimalkan waktu pengerjaan dari tiap *job*. Semakin rendah nilai yang dihasilkan oleh sebuah algoritma pada parameter *average completion time*, maka algoritma tersebut bekerja dengan optimal, karena dapat memaksimalkan waktu dari *job throughput.* Semakin rendah nilai yang dihasilkan oleh sebuah algoritma pada parameter *task fail rate*, maka algoritma tersebut bekerja dengan optimal, karena dapat meminimalkan waktu dari pengerjaan tiap *job.*

Pada skenario 1 (*job bersifat homogen*) algoritma HFSP mengungguli, algoritma *Fair Scheduler* pada keseluruhan parameter, hal ini dikarenakan algoritma HFSP memiliki sifat mengerjakan job berdasarkan ukuran terkecil sehingga cenderung mempercepat eksekusi tiap job agar menghasilkan waktu eksekusi job menjadi cepat pada seluruh parameter yang diujikan.

Pada skenario 2,3 dan 4 (*job bersifat homogen*) algoritma HFSP mengungguli, algoritma *Fair Scheduler* pada parameter *average completion time,* dan *job throughput* hal ini dikarenakan algoritma HFSP memiliki sifat mengerjakan job berdasarkan ukuran terkecil sehingga cenderung mempercepat eksekusi tiap job agar menghasilkan waktu eksekusi job menjadi cepat, sementara pada skenario 2 algoritma *Fair Scheduler* menungguli algoritma HFSP pada parameter *task fail rate* hal ini terjadi karena pada algoritma fair membagi kesulurahan resource sama rata sehingga menyebabkan minimnya kegagalan dalam pengerjaan job.

Pada skenario 5 dan 6 (*job bersifat heterogen*) algoritma HFSP mengungguli, algoritma *Fair Scheduler* pada parameter *average completion time,* dan *job throughput* hal ini dikarenakan algoritma HFSP memiliki sifat mengerjakan job berdasarkan ukuran terkecil sehingga cenderung mempercepat eksekusi tiap job agar menghasilkan waktu eksekusi job menjadi cepat, sementara pada skenario 2 algoritma *Fair Scheduler* menungguli algoritma HFSP pada parameter *task fail rate* hal ini terjadi karena pada algoritma fair membagi kesulurahan resource sama rata sehingga menyebabkan minimnya kegagalan dalam pengerjaan job.

Baik dalam algoritma Fair dan HFSP alokasi dari *value maximum mapper* dan *reducer* sangat diperlukan, semakin banyak jumlah *maximum mapper* dan *reducer* akan mempersingkat waktu dari parameter *average completion time* dari tiap *job* yang akan dikerjakan (*dengan catatan value mapper diset sesuai dengan batas maximum dari masing-masing algoritma*).

Dari keseluruhan skenario dapat disimpulkan bahwa semakin rendahnya nilai average completion time dan task fail rate akan mempengaruhi tingginya nilai job throughput dan dari keseluruhan pengujian, jenis job yang memiliki nilai average completion time tertinggi adalah job wordcount, nilai task fail rate terendah adalah job grep, dan nilai job throughput tertinggi adalah job grep.

4. KESIMPULAN

Hasil pengujian diperoleh bahwa HFSPS lebih baik untuk menangani jenis jos yang homogen dilihat dari parameter *average completion time* dan *job throughput.* Sementara Fair Scheduller masih menunjukkan performa yang dominan ketika digunakan untuk mengeksekusi job yang sifatnya heterogen dilihat dari keseluruhan parameter uji. Dari keseluruhan pengujian, jenis job yang memiliki

nilai average completion time tertinggi adalah job wordcount, nilai task fail rate terendah adalah job grep, dan nilai job throughput tertinggi adalah job grep. Adapun saran pengembangan dari penelitian ini yaitu dengan memodifikasi scheduler yang lain dimana dapat diimplementasi, menambah resource data, menambah jumlah server serta memperbanyak jumlah variasi *jobs* yang dijalankan sistem Hadoop, dengan harapan akan dapat lebih terlihat karakteristik dari Hadoop secara lebih jelas.

DAFTAR PUSTAKA

[1] M. Karanasou, A. Ampla, C. Doulkeridis, and M. Halkidi, "Scalable and Real-Time Sentiment Analysis of Twitter Data," *in IEEE International Conference on Data Mining Workshops, ICDMW*, 2016.

[2] Dijcks, J. (2012). Oracle: Big data for the enterprise. Oracle White Paper, (June), 16. Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Oracle+:+Big+Data+for+the+Enterprise#0

[3] T. White, Hadoop: The definitive guide, 4th ed. Sebastopol, CA, USA: O'reilly , 2015.

[4] M. Pastorelli, D. Carra, M. Dellamico, and P. Michiardi, "HFSP: Bringing Size-Based Scheduling to Hadoop," *IEEE Trans. Cloud Comput.*, 2017.

[5] T. R. Pamungkas, F. A. Yulianto, and S. Prabowo, "Analisis Penggabungan Delay Scheduling dan Fair share Scheduling Algorithm Dengan beberapa karakteristik Job pada Hadoop Analysis Combination Delay Scheduling and Fair share Scheduling Algorithm with Job Characteristics On Hadoop," *e-Proceeding Eng. Telkom Univ.*, Vol. 3, No. 1, pp. 994–1002, 2013.

[6] S. Prabowo, "Implementasi Algoritma Penjadwalan untuk pengelolaan Big Data dengan Hadoop," *Indones. J. Computing*., Vol 2, No 2, pp. 119-126, 2017.

[7] R. . Pratomo, M. Abdurohman, and S. Prabowo, "Analisis Perbandingan Performa Classification and Optimization based Scheduler for Heterogenous Hadoop Systems (COSHH) dan Fair Scheduler pada Job Scheduling Berdasarkan Karakteristik Job pada Hadoop," *e-Proceeding Eng. Telkom Univ.*, 2017.

[8] M. K. Danthala, "Tweet Analysis : Twitter Data processing Using Apache Hadoop," *Int. J. Core Eng. Manag.*, vol. 1, no. 11, pp. 94–102, 2015.

[9] M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy, S. Shenker, and I. Stoica, "Delay scheduling: A simple technique for achieving locality and fairness in cluster scheduling," *in EuroSys'10 - Proceedings of the EuroSys* Conference, 2010.

[10] R. B. Madhumala and Y. B. Ravichandra, "A MapReduce Framework for an Effective Scheduler Based on the Job Size in Hadoop," *International Journal of Science and Research (IJSR).*, vol 4, issue 7, pp.261-266., 2015.

[11] Y. Xia, L. Wang, Q. Zhao, and G. Zhang, "Research on Job Scheduling Algorithm in Hadoop," *Journal of Computational Information Systems.*, vol. 7,issue.16, pp.5769-5775, 2011.

[12] K. D. Priharyani, G. B. Satrya, A. Herutomo, and M. Eng, "Analisis Penggunaan Algoritma Delay Scheduling terhadap Karakteristik Job Scheduling pada Hadoop," *e-Proceeding Eng. Telkom Univ.*, vol. 2, no. 1, pp. 3–10, 2013.

[13] U. Chaudhary and H. Singh, "Mapreduce Performance Evaluation through Benchmarking and Stress Testing On Multi-Node Hadoop Cluster," *Int. J. Comput. Eng. Res.*, vol. 04, no. 5, pp. 2250–3005, 2014.

[14] F. Faghri, S. Bazarbayev, M. Overholt, R. Farivar, R. H. Campbell, and W. H. Sanders, "Failure scenario as a service (FSaaS) for Hadoop clusters," *Proceedings of the Workshop on Secure and Dependable Middleware for Cloud Monitoring and Management (SDMCMM)*, 2012. https://doi.org/10.1145/2405186.2405191