REPOSITOR, Vol. 2, No. 4, April 2020, Pp. 463-474
ISSN : 2714-7975
E-ISSN : 2716-1382

Perbandingan Kinerja Komputasi Hadoop dan Spark untuk Memprediksi Cuaca
(Studi Kasus: Storm Event Database)

Rendiyono Wahyu Saputro[*1], Aminuddin[2], Yuda Munarko[3]
[1,2.3]Teknik Informatika/Universitas Muhammadiyah Malang
rendi.7936@gmail.com[*1], aminuddin@umm.ac.id[2], yuda.munarko@umm.ac.id[3]

Abstrak

Perkembangan teknologi telah mengakibatkan pertumbuhan data yang semakin cepat dan besar setiap waktunya. Hal tersebut disebabkan oleh banyaknya sumber data seperti mesin pencari, RFID, catatan transaksi digital, arsip video dan foto, user generated content, internet of things, penelitian ilmiah di berbagai bidang seperti genomika, meteorologi, astronomi, fisika, dll. Selain itu, data - data tersebut memiliki karakteristik yang unik antara satu dengan lainnya, hal ini yang menyebabkan tidak dapat diproses oleh teknologi basis data konvensional. Oleh karena itu, dikembangkan beragam framework komputasi terdistribusi seperti Apache Hadoop dan Apache Spark yang memungkinkan untuk memproses data secara terdistribusi dengan menggunakan gugus komputer. Adanya ragam framework komputasi terdistribusi, sehingga diperlukan sebuah pengujian untuk mengetahui kinerja komputasi keduanya. Pengujian dilakukan dengan memproses dataset dengan beragam ukuran dan dalam gugus komputer dengan jumlah node yang berbeda. Dari semua hasil pengujian, Apache Hadoop memerlukan waktu yang lebih sedikit dibandingkan dengan Apache Spark. Hal tersebut terjadi karena nilai throughput dan throughput/node Apache Hadoop lebih tinggi daripada Apache Spark.

Kata Kunci: *Hadoop, Spark, H2o, Deep Learning, Storm Event Database*

Abstract

Technological developments have resulted in rapid and growing data growth every time. This is due to the large number of data sources such as search engines, RFID, digital transaction records, video and photo archives, user generated content, internet of things, scientific research in areas such as genomics, meteorology, astronomy, physics, In addition, these data have unique characteristics of each other, this is the cause cannot be processed by conventional database technology. Therefore, developed various distributed computing frameworks such as Apache Hadoop and Apache Spark that enable to process data in a distributed by using computer cluster. The existence of various frameworks of distributed computing, so required a test to determine the performance of both computing. Testing is done by processing datasets of various sizes and in clusters of computers with different number of nodes. Of all the test results, Apache Hadoop takes less time than the Apache Spark. This happens because the value of throuhgput and throughput / node Apache Hadoop is higher than Apache Spark.

Keywords: *Hadoop, Spark, H2o, Deep Learning, Storm Event Database*

1. Pendahuluan

Perkembangan teknologi telah memudahkan seseorang dalam memperoleh dan menyebarkan informasi baik berupa video, audio, gambar maupun teks. Hal ini juga didukung oleh munculnya beragam jejaring sosial yang merangsang masyarakat dunia untuk selalu bertukar informasi terkini.

Kebiasaan untuk selalu bertukar informasi tersebut membuat data bertambah dengan cepat dan dan menghasilkan ukuran yang besar. Selain jejaring sosial, ada beberapa sumber yang menyebabkan hal itu terjadi antara lain mesin pencari, RFID, catatan transaksi digital, arsip video dan foto, *user generated content*, *internet of things*, penelitian ilmiah di berbagai bidang seperti genomika, meteorologi, astronomi, fisika, dll.

Sumber - sumber tersebut akan menghasilkan data secara terus - menerus dan tidak akan berhenti sepanjang perkembangan teknologi semakin cepat. Data - data yang dihasilkan hanya akan menjadi sampah informasi karena tidak semua data memiliki informasi yang bermanfaat.

Selain itu, jika hal ini terus dibiarkan maka akan memakan space yang besar dan tidak ada manfaat yang diperoleh. Fenomena tersebut dikenal dengan istilah *big data*.

Big data memiliki definisi yang berbeda di kalangan para peneliti karena tidak adanya aturan baku mengenai karakteristik *big data*. Secara umum, *big data* adalah data yang berukuran sangat besar, bertambah dengan cepat dan memiliki ragam data sehingga tidak dapat diproses secara efektif dengan menggunakan teknologi konvensional dalam waktu singkat [1].

Selain itu, ada yang menyebutkan big data dengan istilah 9 V yaitu *Veracity*, *Variety*, *Velocity*, *Volume*, *Validity*, *Variability*, *Volatility*, *Visualization* dan *Value* [2].

Definisi 9 V menunjukan bahwa *big data* memiliki karakteristik yang kompleks sehingga perlu adanya metode tertentu dalam memproses *big data* agar dapat dianalisis untuk memperoleh informasi yang bermanfaat.

Adapun manfaat dari analisis *big data* telah dirasakan oleh banyak pihak di berbagai bidang. Contohnya seperti organisasi kesehatan dapat memprediksi penyebaran penyakit agar dapat mencegah penyebarannya semakin luas, pengambilan informasi daratan bumi oleh lembaga meteorologi dan geofisik mengenai dapat digunakan untuk prakiraan cuaca atau pengawasan terhadap cuaca buruk serta masih banyak lagi [3].

Munculnya beragam *framework big data* disebabkan oleh meningkatnya kebutuhan dalam menganalisis *big data*. Beberapa *framework* sengaja dikembangkan dengan tujuan khusus seperti Spark [4] dan Hadoop untuk memproses *big data*, Mahout untuk menyediakan algoritma machine learning dan data mining, Sqoop untuk transfer data dari basis data relasional ke Hadoop, dll [5].

Hadoop dan Spark merupakan *framework* yang sangat penting karena peran keduanya sangat krusial dalam analisis *big data* yaitu memproses *big data* secara terdistribusi, sehingga seringkali disebut dengan *framework* komputasi terdistribusi. Karena memiliki kemampuan yang sama, tentu perlu adanya pengujian untuk menentukan *framework* dengan kinerja komputasi terbaik. Hal ini dilakukan dengan harapan agar pihak yang menggunakan *framework* tersebut dapat menentukan pilihan sesuai dengan kebutuhan. Beberapa penelitian telah melakukan pengujian diantaranya:

Penelitian yang dilakukan oleh Satish Gopalani dan Rohan Arora, 2015 dengan judul "Comparing Apache Spark and Map Reduce with Performance Analysis using K-Means". Penelitian tersebut dilakukan untuk mengetahui kinerja komputasi Spark dan Hadoop dengan menggunakan algoritma K-Means. Dalam menentukan *framework* dengan kinerja komputasi terbaik, kriteria yang digunakan adalah *running time* [6].

Berikutnya, penelitian dengan judul "Processing time of TFIDF and Naive Bayes on Spark 2.0, Hadoop 2.6 and Hadoop 2.7: Which Tool Is More Efficient?" yang dilakukan oleh Erin Gilheany, 2016. Penelitian tersebut membandingkan Hadoop dan Spark dalam memproses data dengan menggunakan algoritma TF-IDF dan Naive Bayes. Kriteria pengujian yang digunakan dalam penelitian adalah *running time* [7].

Penelitan selanjutnya yang dilakukan oleh Lu Liu, 2015 dengan judul "Performance Comparison by Running Benchmarks on Hadoop, Spark, and Hamr". Penelitian ini dilakukan untuk menguji kinerja Hadoop, Spark dan Hamr dengan menjalankan beberapa beban kerja yang telah disediakan oleh Hibench, sebuah alat benchmark yang dikembangkan oleh peneliti Intel Corporation. Beberapa beban kerja ada yang memproses data dengan menggunakan algoritma Naive Bayes dan K-Means. kriteria yang digunakan dalam penelitian ini yaitu *running time*, *CPU usage*, *memory usage*, *throughput*, dan *throughput/node* [8].

Berdasarkan penelitian diatas, dapat ditarik kesimpulan bahwa hanya ada dua algoritma yang sudah digunakan dalam pengujian kinerja komputasi Hadoop dan Spark. Algoritma K-Means dan Naive Bayes sengaja dipilih dalam penelitian tersebut karena keduanya didukung secara *native* baik oleh Hadoop maupun Spark. Padahal, banyak sekali algoritma *machine learning* atau *data mining* yang bisa digunakan dalam pengujian, salah satunya adalah Deep Learning.

Deep Learning merupakan algoritma yang sangat populer saat ini. Hal tersebut disebabkan karena dapat meningkatkan kemampuan dalam *speech recognition*, *visual object recognition*, *object detection*, *natural languange processing*, dll. Algoritma ini juga telah membuka jalan kepada peneliti untuk mewujudkan sebuah *artificial intelligence* yang dapat membantu manusia dalam berbagai hal [9].

Berdasarkan hal tersebut, Deep Learning dipilih sebagai algoritma yang digunakan dalam penelitian ini untuk menguji kinerja komputasi Hadoop dan Spark.

Dalam melakukan pengujian, perlu adanya sebuah data yang dapat digunakan untuk merepresentasikan karakteristik *big data*. Beberapa penelitian telah menggunakan data cuaca seperti Storm Event Database [10] dan Global Surface Summary Of The Day [11]. Data - data tersebut memiliki ukuran yang besar dan bertambah dengan cepat sehingga sangat cocok dengan karakteristik *big data*.

Dalam penelitian ini dilakukan suatu analisa kinerja komputasi Hadoop dan Spark dengan menggunakan data cuaca yaitu Storm Event Database dan Global Surface Summary Of The Day. Pengujian dilakukan dengan melakukan pemrosesan data cuaca menggunakan Hadoop dan Spark yang diimplementasikan dalam sebuah gugus komputer dengan skenario penambahan jumlah komputer dan ukuran dataset yang diproses.

Gugus komputer akan diimplementasikan dengan maksimal komputer berjumlah 4. Adapun kriteria dalam pengujian adalah *running time*, *max CPU usage, average CPU usage*, *max memory usage*, *average memory usage*, *throughput*, dan *throughput/node.*

2. Metode Penelitian

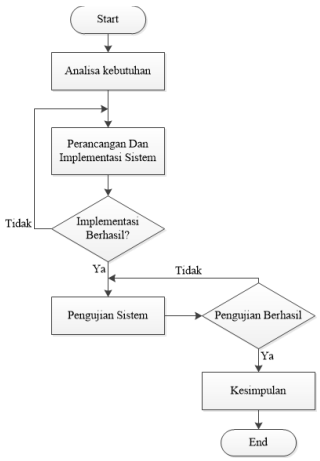
Tahap kerja penelitian ini akan diuraikan pada kerangka kerja, yang mana kerangka kerja tersebut akan digunakan sebagai acuan atau langkah dalam membangun sistem "*Perbandingan Kinerja Komputasi Hadoop dan Spark untuk memprediksi Cuaca (Studi Kasus: Storm Event Database)*" yang akan di implementasikan dengan Iaas. Pada awalnya yang akan dilakukan yaitu melakukan analisis terhadap kebutuhan membangun sistem ini. Selanjutnya akan dilakukan dengan perancangan dan pengujian kinerja komputasi dari masing - masing framework komputasi terdistribusi serta dibandingkan untuk mengetahui hasilnya.

Dalam metodologi penelitian ini dapat diketahui bahwa permasalahan terhadap big data membuat perkembangan framework komputasi terdistribusi semakin meningkat. Munculnya beragam framework, tentu menghasilkan dilema dalam menentukan framework yang tepat karena setiap framework dikembangkan dengan tujuan tertentu.

Pemilihan Apache Hadoop dan Apache Spark dalam pengujian ini didasari bahwa kedua framework tersebut sangat popular. Selain itu, dengan kode sumber yang terbuka memungkinkan untuk pengembangan framework yang cepat karena dikembangkan secara bersama - sama oleh orang - orang di berbagai belahan dunia. Hanya saja, saat ini masih belum ada penelitian mengenai kinerja komputasi keduanya dengan menggunakan algoritma Deep Learning. Algoritma Deep Learning juga sangat popular saat ini karena membantu para peneliti untuk memahami kinerja kecerdasan buatan semakin baik.

Dengan adanya masalah tersebut, maka metodologi yang dapat digunakan untuk mengetahui kinerja komputasi yaitu dengan mengimplementasikan gugus komputer untuk memproses dataset dengan berbagai ukuran dan dalam jumlah komputer yang berbeda.

Pada bab ini, metode penelitian yang akan digunakan pada tugas akhir ini adalah studi pustaka, yang mana dengan melakukan penelitian dengan cara mencari sumber literature, baik bersumber dari jurnal-jurnal, buku-buku, maupun artikel-artikel yang berhubungan dengan masalah serta perancangan dan implementasi pada bahan yang dibahas sesuai dengan yang akan diteliti, seperti pada penjelasan pada Gambar 1.

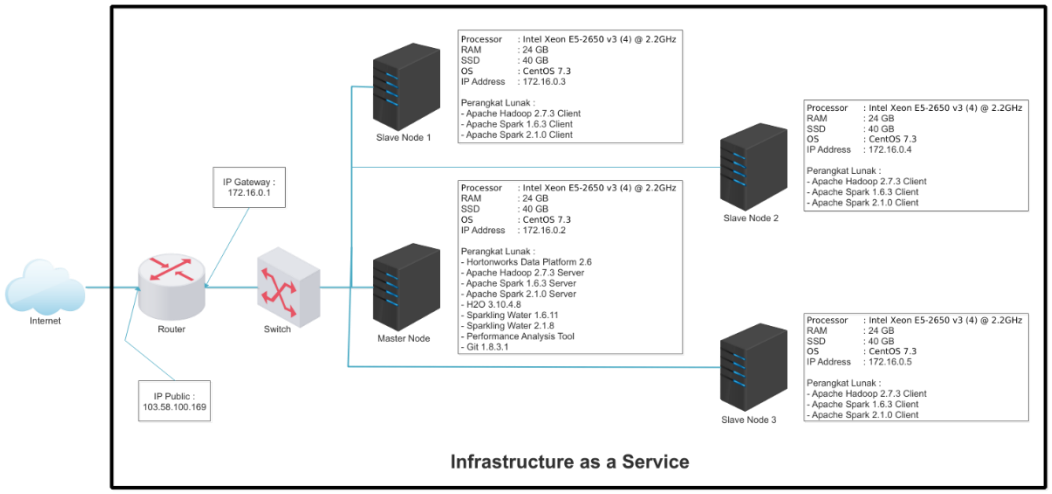


Gambar 1. Flowchart Tahapan Pengerjaan Sistem

3. Analisa dan Perancangan Sistem

3.1 Analisa Sistem

Gugus komputer akan diimplementasikan dengan menggunakan IaaS yaitu BiznetGioCloud. IaaS menyediakan berbagai sumber daya sehingga sangat mudah untuk merancang arsitektur sistem yang diinginkan. Gambar 2 berikut ini adalah arsitektur sistem yang akan digunakan dalam penelitian.



Gambar 2. Topologi Sistem

Gugus komputer dibangun dengan menggunakan 4 cluster node. Cluster node tersebut terdiri dari 1 master node dan 3 slave node, masing - masing memiliki spesifikasi Processor Intel Xeon E5-2650 v3 (4) @ 2.2GHz, RAM 24 GB, SSD 80 GB untuk master node serta SSD 40 GB untuk slave node dan menggunakan sistem operasi CentOS 7.3. IP Address untuk setiap cluster node akan diatur sebelum digunakan dalam IaaS.

Master node berperan dalam mengatur gugus komputer secara keseluruhan dan memberikan perintah kepada slave node. Sedangkan slave node menjalankan semua perintah yang diberikan pengguna dan master node, baik dilakukan secara independen atau bekerja sama dengan slave node lainnya. Master node dan slave node akan dipasang dengan perangkat lunak yang berbeda. Informasi tersebut dapat dilihat pada Gambar 2.

Cluster node tersebut tersambung dengan sebuah Switch, sehingga memungkinkan untuk saling berkomunikasi antara satu dengan lainnya. Switch tersambung dengan sebuah Router melalui IP Gateway. IP Gateway digunakan oleh cluster node untuk mendapatkan akses internet melalui Router.

Hal tersebut dapat terjadi karena Router memiliki IP Public, sebuah IP Address khusus yang tersambung dengan internet secara langsung. IP Gateway dan IP Public juga secara otomatis diatur oleh IaaS ketika meluncurkan sebuah cluster node.

3.2 Analisa Kebutuhan Sistem

Pada tahap ini akan diuraikan kebutuhan apa saja yang akan diperlukan, mulai dari perangkat keras maupun perangkat lunak yang akan digunakan pada masing-masing *server*.

3.3 Kebutuhan Perangkat Keras

Tabel 1 berikut ini adalah rincian dan spesifikasi perangkat keras yang akan diperlukan untuk membangun sistem ini.

Tabel 1. Kebutuhan Perangkat Keras

Nama	Spesifikasi Kebutuhan		
	Processor	RAM	SSD
Master Node	Intel Xeon E5-2650 v3 (4) @ 2.299GHz	24 GB	80 GB
Slave Node 1	Intel Xeon E5-2650 v3 (4) @ 2.299GHz	24 GB	40 GB
Slave Node 2	Intel Xeon E5-2650 v3 (4) @ 2.299GHz	24 GB	40 GB
Slave Node 3	Intel Xeon E5-2650 v3 (4) @ 2.299GHz	24 GB	40 GB

3.4 Kebutuhan Perangkat Lunak

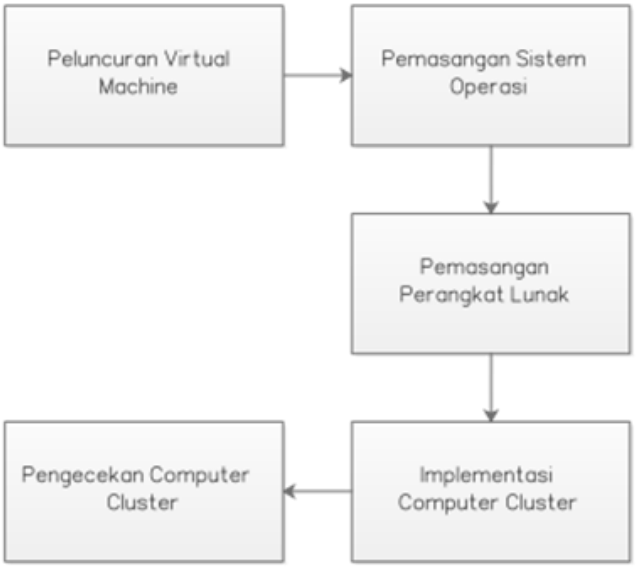
Perangkat lunak yang diperlukan untuk membangun sistem ini seperti pada Tabel 2.

Tabel 2. Spesifikasi Kebutuhan Perangkat Lunak

Perangkat Lunak	Deskripsi
CentOS 7.3	Sistem operasi yang dikembangkan dari kode sumber Red Hat Enterprise Linux.
Hortonworks Data Platform 2.6	Salah satu Hadoop Distribution yang memudahkan dalam memasang berbagai framework khusus untuk menangani data berukuran besar.
Apache Hadoop 2.7.3	Sebuah framework komputasi terdistribusi yang digunakan untuk memproses data berukuran besar secara terdistribusi.
Apache Spark 1.6.3	Sebuah alternatif dari Apache Hadoop, juga dapat melakukan pemrosesan data secara terdistribusi.
Apache Spark 2.1.0	Versi terbaru dari Apache Spark yang dirilis pada tanggal 28 Desember 2016.
H2O 3.10.4.8	Perangkat lunak yang dapat memproses data dengan menggunakan berbagai algoritma machine learning, salah satunya Deep Learning.
Sparkling Water 1.6.11	H2O yang dikembangkan secara khusus agar dapat terintegrasi dengan Apache Spark 1.6.8.
Sparkling Water 2.1.8	H2O yang dikembangkan khusus untuk Apache Spark 2.1.0. Pengembangan Sparkling Water yang berbeda disebabkan oleh perubahan Apache Spark dalam hal teknis.
Performance Analysis Tool	Kumpulan program yang digunakan untuk mengumpulkan informasi CPU, memori, dan jaringan pada sebuah komputer atau gugus komputer.
Git 1.8.3.1	Git dapat melacak perubahan dalam berkas, dan biasa digunakan untuk pengembangan perangkat lunak.

3.5 Perancangan Sistem

Gambar 3 berikut ini adalah blok diagram perancangan sistem Perbandingan Kinerja Komputasi Hadoop dan Spark untuk Memprediksi Cuaca (Studi Kasus: Storm Event Database) yang akan diimplementasikan.



Gambar 3. Blok Diagram Perancangan Sistem Cluster

4. Hasil Penelitian dan Pembahasan

4.1 Implementasi Sistem

Berdasarkan blok diagram pada Gambar 3, berikut penjelasan mengenai masing - masing tahapan.

Tahap awal dalam implementasi sistem adalah meluncurkan Virtual Machine dengan menggunakan IaaS. IaaS menyediakan berbagai jenis Virtual Machine agar pengguna dapat memilih sesuai dengan kebutuhannya. Virtual Machine tersebut dapat dihentikan atau dihapus kapanpun saat tidak lagi diperlukan.

Tahap ke-2 adalah memasang sistem operasi pada Virtual Machine. Sebelum Virtual Machine diluncurkan, ada pilihan untuk menentukan sistem operasi yang akan digunakan. Proses pemasangan sistem operasi akan berlangsung secara otomatis.

Tahap ke-3 yaitu memasang perangkat lunak pada Virtual Machine. Virtual Machine yang diluncurkan akan membentuk jaringan komputer. Dengan memasang perangkat lunak khusus, jaringan komputer tersebut dapat digunakan untuk memproses masalah komputasi secara bersama - sama dan biasa disebut dengan computer cluster. Computer cluster akan dibangun menggunakan arsitektur master-slave, dan keduanya dipasang dengan perangkat lunak yang berbeda. Penjelasan mengenai perangkat lunak tersebut telah dijelaskan pada Tabel 3.2.

Tahap ke-4, implementasi computer cluster. Computer cluster dapat dibangun dengan bantuan perangkat lunak khusus yang dipasang pada setiap master node dan slave node. Pada proses tersebut, ada beberapa konfigurasi yang perlu dilakukan seperti pengaturan IP Address, Hostname, SSH Key, dll.

Tahap ke-5, pengecekan computer cluster. Setelah melakukan beberapa konfigurasi, computer cluster perlu dicek ulang untuk memastikan dapat berjalan dengan baik dan mencegah terjadinya kegagalan pada tahap pengujian sistem.

4.2 Hasil Pengujian Framework Komputasi Terdistribusi beserta Perbandingan

1. Hasil Pengujian *Running Time*

Pada hasil pengujian Tabel 3, penulis membandingkan lama waktu yang diperlukan oleh masing - masing framework komputasi terdistribusi untuk memproses data dengan berbagai ukuran dalam gugus komputer dengan node yang berbeda.

Tabel 3. Tabel Pengujian Running Time

Framework	Dataset	Running Time (s)		
		2 Node	3 Node	4 Node
Apache Hadoop 2.7.3	SED 250 MB	71	70	66
Apache Spark 1.6.3		93	86	85
Apache Spark 2.1.0		103	98	86
Apache Hadoop 2.7.3	SED 500 MB	95	92	85
Apache Spark 1.6.3		122	106	102
Apache Spark 2.1.0		137	115	104
Apache Hadoop 2.7.3	SED 1 GB	179	136	112
Apache Spark 1.6.3		184	149	134
Apache Spark 2.1.0		198	173	137
Apache Hadoop 2.7.3	GSOD 2 GB	204	173	166
Apache Spark 1.6.3		225	210	188
Apache Spark 2.1.0		244	217	196
Apache Hadoop 2.7.3	GSOD 4 GB	281	244	220
Apache Spark 1.6.3		334	272	266
Apache Spark 2.1.0		356	297	269
Apache Hadoop 2.7.3	GSOD 6 GB	370	310	279
Apache Spark 1.6.3		431	362	288
Apache Spark 2.1.0		438	371	317

2. Hasil Pengujian Throughput

Pada hasil pengujian Tabel 4, penulis membandingkan ukuran data yang diproses per detik oleh masing - masing framework komputasi terdistribusi untuk memproses data dengan berbagai ukuran dalam gugus komputer dengan node yang berbeda.

Tabel 4. Tabel Pengujian Throughput

Framework	Dataset	Throughput (b/s)		
		2 Node	3 Node	4 Node
Apache Hadoop 2.7.3	SED 250 MB	3692167,549	3744912,8	3971877,212
Apache Spark 1.6.3		2818751,57	3048184,837	3084045,835
Apache Spark 2.1.0		2545086,369	2674937,714	3048184,837
Apache Hadoop 2.7.3	SED 500 MB	5518893,968	5698857,902	6168175,612
Apache Spark 1.6.3		4297499,402	4946178,557	5140146,343
Apache Spark 2.1.0		3826970,27	4559086,322	5041297,375
Apache Hadoop 2.7.3	SED 1 GB	5999114,609	7895893,493	9587870,67
Apache Spark 1.6.3		5836095,19	7206990,034	8013742,649
Apache Spark 2.1.0		5423441,995	6207176,387	7838259,234
Apache Hadoop 2.7.3	GSOD 2 GB	10526880,47	12413200,09	12936648,28
Apache Spark 1.6.3		9544371,622	10226112,45	11422785,19
Apache Spark 2.1.0		8801162,357	9896237,857	10956549,06
Apache Hadoop 2.7.3	GSOD 4 GB	15284580,54	17602324,32	19522577,88
Apache Spark 1.6.3		12859183,03	15790320,34	16146492,98
Apache Spark 2.1.0		12064514,42	14461168,8	15966420,57
Apache Hadoop 2.7.3	GSOD 6 GB	17412031,82	20782102,49	23091224,99
Apache Spark 1.6.3		14947683,93	17796828,1	22369624,21
Apache Spark 2.1.0		14708794	17365099,12	20323191,71

3. Hasil Pengujian Throughput/node

Pada hasil pengujian Tabel 5, penulis membandingkan ukuran data yang diproses per detik per node oleh masing - masing framework komputasi terdistribusi untuk memproses data dengan berbagai ukuran dalam gugus komputer dengan node yang berbeda.

Tabel 5. Tabel Pengujian Throughput/Node

Framework	Dataset	Throughput/node (b/s)		
		2 Node	3 Node	4 Node
Apache Hadoop 2.7.3	SED 250 MB	1846083,775	1248304,267	992969,303
Apache Spark 1.6.3		1409375,785	1016061,612	771011,4588
Apache Spark 2.1.0		1272543,184	891645,9048	762046,2093
Apache Hadoop 2.7.3	SED 500 MB	2759446,984	1899619,301	1542043,903
Apache Spark 1.6.3		2148749,701	1648726,186	1285036,586
Apache Spark 2.1.0		1913485,135	1519695,441	1260324,344
Apache Hadoop 2.7.3	SED 1 GB	2999557,304	2631964,498	2396967,667
Apache Spark 1.6.3		2918047,595	2402330,011	2003435,662
Apache Spark 2.1.0		2711720,997	2069058,796	1959564,808
Apache Hadoop 2.7.3	GSOD 2 GB	5263440,233	4137733,362	3234162,071
Apache Spark 1.6.3		4772185,811	3408704,151	2855696,297

Apache Spark 2.1.0		4400581,178	3298745,952	2739137,264
Apache Hadoop 2.7.3		7642290,272	5867441,439	4880644,469
Apache Spark 1.6.3	GSOD 4 GB	6429591,516	5263440,114	4036623,245
Apache Spark 2.1.0		6032257,209	4820389,599	3991605,142
Apache Hadoop 2.7.3		8706015,908	6927367,497	5772806,247
Apache Spark 1.6.3	GSOD 6 GB	7473841,963	5932276,033	5592406,052
Apache Spark 2.1.0		7354397	5788366,372	5080797,927

4. Hasil Pengujian Average Memory Usage

Pada hasil pengujian Tabel 6 penulis membandingkan nilai rata - rata penggunaan memori oleh masing - masing framework komputasi terdistribusi untuk memproses data dengan berbagai ukuran dalam gugus komputer dengan node yang berbeda.

Tabel 6. Tabel Pengujian Average Memory Usage

Framework	Dataset	Average Memory Usage (%)		
		2 Node	3 Node	4 Node
Apache Hadoop 2.7.3		27,389	32,263	24,521
Apache Spark 1.6.3	SED 250 MB	44,262	27,409	18,625
Apache Spark 2.1.0		30,871	24,537	28,922
Apache Hadoop 2.7.3		30,441	34,529	26,049
Apache Spark 1.6.3	SED 500 MB	48,091	27,993	20,586
Apache Spark 2.1.0		37,069	26,984	30,225
Apache Hadoop 2.7.3		42,520	40,876	28,879
Apache Spark 1.6.3	SED 1 GB	45,614	30,720	25,291
Apache Spark 2.1.0		50,568	34,477	32,444
Apache Hadoop 2.7.3		60,840	40,562	31,796
Apache Spark 1.6.3	GSOD 2 GB	60,984	37,812	27,799
Apache Spark 2.1.0		44,617	32,580	15,215
Apache Hadoop 2.7.3		78,278	50,743	42,943
Apache Spark 1.6.3	GSOD 4 GB	75,565	44,914	39,855
Apache Spark 2.1.0		64,990	47,018	40,652
Apache Hadoop 2.7.3		73,032	66,317	55,824
Apache Spark 1.6.3	GSOD 6 GB	71,518	55,461	44,819
Apache Spark 2.1.0		70,988	55,670	53,774

5. Hasil Pengujian Max Memory Usage

Pada hasil pengujian Tabel 7, penulis membandingkan nilai maksimum penggunaan memori oleh masing - masing framework komputasi terdistribusi untuk memproses data dengan berbagai ukuran dalam gugus komputer dengan node yang berbeda.

Tabel 7. Tabel Pengujian Max Memory Usage

Framework	Dataset	Max Memory Usage (%)		
		2 Node	3 Node	4 Node
Apache Hadoop 2.7.3		31,470	36,053	28,020
Apache Spark 1.6.3	SED 250 MB	49,920	41,100	23,247
Apache Spark 2.1.0		37,945	30,517	34,047

Apache Hadoop 2.7.3		36,890	40,270	30,230
Apache Spark 1.6.3	SED 500 MB	56,210	68,470	26,277
Apache Spark 2.1.0		48,475	36,020	36,850
Apache Hadoop 2.7.3		58,955	63,930	35,667
Apache Spark 1.6.3	SED 1 GB	62,680	78,700	34,140
Apache Spark 2.1.0		66,355	74,490	43,100
Apache Hadoop 2.7.3		87,580	55,410	34,242
Apache Spark 1.6.3	GSOD 2 GB	67,220	85,420	31,727
Apache Spark 2.1.0		50,315	46,555	35,835
Apache Hadoop 2.7.3		98,000	68,700	48,200
Apache Spark 1.6.3	GSOD 4 GB	98,920	64,535	82,060
Apache Spark 2.1.0		98,980	63,145	47,420
Apache Hadoop 2.7.3		98,170	87,155	72,330
Apache Spark 1.6.3	GSOD 6 GB	98,820	74,240	52,625
Apache Spark 2.1.0		98,500	74,875	98,910

6. Hasil Pengujian Average CPU Usage

Pada hasil pengujian Tabel 8, penulis membandingkan nilai rata - rata penggunaan CPU oleh masing-masing framework komputasi terdistribusi untuk memproses data dengan berbagai ukuran dalam gugus komputer dengan node yang berbeda.

Tabel 8. Tabel Pengujian Average CPU Usage

Framework	Dataset	Average CPU Usage (%)		
		2 Node	3 Node	4 Node
Apache Hadoop 2.7.3		50,572	37,842	33,086
Apache Spark 1.6.3	SED 250 MB	49,117	37,761	27,713
Apache Spark 2.1.0		45,113	34,925	27,696
Apache Hadoop 2.7.3		52,847	39,014	34,980
Apache Spark 1.6.3	SED 500 MB	45,904	37,152	27,309
Apache Spark 2.1.0		47,245	36,971	28,551
Apache Hadoop 2.7.3		43,328	36,332	34,348
Apache Spark 1.6.3	SED 1 GB	47,000	38,069	27,817
Apache Spark 2.1.0		45,575	35,254	28,295
Apache Hadoop 2.7.3		71,669	67,626	64,246
Apache Spark 1.6.3	GSOD 2 GB	64,865	55,295	50,814
Apache Spark 2.1.0		65,078	56,313	46,647
Apache Hadoop 2.7.3		75,419	66,252	59,792
Apache Spark 1.6.3	GSOD 4 GB	66,845	59,711	54,068
Apache Spark 2.1.0		67,280	59,221	51,011
Apache Hadoop 2.7.3		74,827	67,213	65,120
Apache Spark 1.6.3	GSOD 6 GB	70,504	59,029	52,429
Apache Spark 2.1.0		70,917	58,095	52,603

7. Hasil Pengujian Max CPU Usage

Pada hasil pengujian ini, penulis membandingkan nilai maksimum penggunaan CPU oleh masing-masing framework komputasi terdistribusi untuk memproses data dengan berbagai ukuran dalam gugus komputer dengan node yang berbeda.

Tabel 9. Tabel Pengujian Max CPU Usage

Framework	Dataset	Max CPU Usage (%)		
		2 Node	3 Node	4 Node
Apache Hadoop 2.7.3	SED 250 MB	91,641	95,848	91,105
Apache Spark 1.6.3		91,902	88,156	81,952
Apache Spark 2.1.0		91,188	84,150	80,646
Apache Hadoop 2.7.3	SED 500 MB	98,599	87,626	99,341
Apache Spark 1.6.3		96,667	92,701	70,069
Apache Spark 2.1.0		97,381	80,714	94,477
Apache Hadoop 2.7.3	SED 1 GB	97,323	97,271	96,693
Apache Spark 1.6.3		92,725	94,170	98,150
Apache Spark 2.1.0		97,541	98,906	93,462
Apache Hadoop 2.7.3	GSOD 2 GB	99,851	99,454	99,751
Apache Spark 1.6.3		97,665	99,364	99,503
Apache Spark 2.1.0		99,452	99,617	99,477
Apache Hadoop 2.7.3	GSOD 4 GB	99,826	99,585	99,525
Apache Spark 1.6.3		98,224	99,435	99,590
Apache Spark 2.1.0		99,575	99,481	99,414
Apache Hadoop 2.7.3	GSOD 6 GB	99,799	99,503	99,614
Apache Spark 1.6.3		99,825	99,503	99,652
Apache Spark 2.1.0		99,752	99,700	99,527

5. Kesimpulan

Dari seluruh percobaan dan pengujian pada sistem yang telah dibangun oleh penulis, dapat disimpulkan sebagai berikut:

1. Apache Hadoop 2.7.3 menggunakan waktu yang lebih sedikit dibandingkan dengan Apache Spark 1.6.3 dan Apache Spark 2.1.0 dalam memproses dataset dengan berbagai ukuran dan dalam mode implementasi gugus komputer yang berbeda.
2. Hal tersebut disebabkan oleh nilai throughput dan throughput per node yang selalu di atas Apache Spark 1.6.3 dan Apache Spark 2.1.0. (ukuran data yang diproses per detik)
3. Pada nilai maximum cpu usage, average cpu usage, maximum memory usage dan average memory usage memiliki pola yang cukup unik karena tidak memiliki pola yang beraturan saat dilakukan pengujian. Saat memproses dataset Storm Event Database, nilai - nilai dari masing - masing framework kadang lebih tinggi atau lebih rendah dari yang lain. Berbeda dengan saat memproses Global Surface Summary of The Day, nilai - nilai tersebut tercatat dengan selisih sangat kecil sehingga menghasilkan nilai yang hampir sama. Penulis menyimpulkan bahwa struktur dataset yang berbeda dapat mempengaruhi penggunaan sumber daya komputer dalam memproses dataset tersebut
4. Pada penelitian sebelumnya, telah dijelaskan bahwa Apache Spark 2.1.0 menggunakan waktu yang lebih sedikit dalam memproses data dibandingkan dengan Apache Hadoop 2.7.3 dan Apache Spark 1.6.3. Berdasarkan pengujian yang telah penulis lakukan, hasilnya cukup berbeda. Hal tersebut dapat dimaklumi karena penulis menggunakan H2O sebagai software tambahan untuk menyediakan algoritma Deep Learning. Algoritma Deep Learning masih belum didukung secara native oleh Apache Hadoop dan Apache Spark. Dengan adanya hasil ini, diharapkan adanya pengembangan sebuah library algoritma Deep Learning agar dapat digunakan oleh Apache Hadoop dan Apache Spark.

Setelah melakukan penelitian ini, ada beberapa saran penulis untuk mengembangkan apa yang telah diujikan antara lain:

1. Munculnya framework baru seperti Apache Flink yang diklaim lebih cepat dari Apache Spark dapat dijadikan bahan pengujian untuk melakukan penelitian kinerja komputasi lebih lanjut.
2. Apache Spark diketahui memiliki abstraksi data yang unik yaitu dataframe, RDD, dan dataset. Sejauh yang penulis tahu, belum ada penelitian yang menjabarkan ke-3 abstraksi data tersebut dan menjelaskan tentang kelebihan dan kekurangan dari masing - masing.
3. Selain H2O, adanya banyak sekali library algoritma Deep Learning seperti TensorFlow, Caffe, MXNET, DeepLearning4J, dll. Adanya fakta ini tentu akan membuka peluang untuk penelitian dalam menguji kinerja komputasi dari masing - masing library.
4. Saat ini, library algoritma Deep Learning berlomba - lomba agar dapat digunakan dengan GPU. Tentu saja sangat diharapkan jika ada penelitian selanjutnya yang membahas tentang kinerja komputasi algoritma Deep Learning dengan menggunakan GPU.

Referensi

[1] R. Han, L. Xiaoyi, and X. Jiangtao, “On Big Data Benchmarking,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 8807, 2014, pp. 3–18.
[2] S. S. Owais and N. S. Hussein, “Extract Five Categories CPIVW from the 9 V’s Characteristics of the Big Data,” Int. J. Adv. Comput. Sci. Appl., vol. 7, no. 3, pp. 254–258, 2016.
[3] R. Kune, P. Konugurthi, A. Agarwal, C. R. Rao, and R. Buyya, “The Anatomy of Big Data Computing,” CoRR, vol. abs/1509.0, 2015.
[4] A. G. Shoro and T. R. Soomro, “Big Data Analysis: Apache Spark Perspective,” Glob. J. Comput. Sci. Technol., vol. 15, no. 1, 2015.
[5] V. B. Bobade, “Survey Paper on Big Data and Hadoop,” Int. Res. J. Eng. Technol., vol. 3, no. 1, pp. 861–863, 2016.
[6] S. Gopalani and R. Arora, “Comparing Apache Spark and Map Reduce with Performance Analysis using K-Means,” Int. J. Comput. Appl., vol. 113, no. 1, pp. 8–11, Mar. 2015.
[7] E. Gilheany, “Processing time of TFIDF and Naive Bayes on Spark 2.0, Hadoop 2.6 and Hadoop 2.7: Which Tool Is More Efficient?,” Dublin, National College of Ireland, 2016.
[8] L. Liu, “Performance Comparison by Running Benchmarks on Hadoop, Spark, and Hamr,” University of Delaware, 2015.
[9] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, May 2015.
[10] C.. Shabariram, K. E. Kannammal, and T. Manojpraphakar, “Rainfall analysis and rainstorm prediction using MapReduce Framework,” 2016 Int. Conf. Comput. Commun. Informatics, vol. 5, no. 11, pp. 1–4, Jan. 2016.
[11] Doreswamy; and I. Gad, “Big Data Techniques: Hadoop and Map Reduce for Weather Forecasting,” Int. J. Latest Trends Eng. Technol. Spec., pp. 194–199, 2016.

ISSN: 2714-7975; E-ISSN: 2716-1382