Jurnal Teknik Informatika dan Teknologi Informasi
Volume 6, Nomor 2, Agustus 2026
e-ISSN: 2827-9387, p-ISSN: 2827-9379, Hal. 498-516
DOI: https://doi.org/10.55606/jutiti.v6i2.7642
Tersedia: https://journalshub.org/index.php/jutiti

Analisis Performa *Hadoop Multi-Node Cluster* pada Skenario Pemrosesan *Small Files*

I Putu Ngurah Rio Aditya Pratama[1*], Ngakan Kutha Krisnawijaya[2]
[1,2]Teknologi Informasi, Universitas Pendidikan Nasional, Indonesia
[*]*Penulis Korespondensi: Riyoaditya444@gmail.com*

Abstract. *The rapid growth of Big Data demands computing systems that are capable of processing data efficiently and in a distributed manner. Apache Hadoop, as a distributed computing framework, is widely used to address the limitations of centralized systems; however, it still faces challenges in handling small files, which can lead to increased file management overhead and imbalanced resource utilization. This study aims to analyze the performance of a Hadoop Multi-Node Cluster in small file processing scenarios, particularly in terms of workload distribution and CPU and memory resource utilization across nodes. The research method involves implementing a Hadoop Multi-Node Cluster in a local environment with a configuration of one master node and five worker nodes, followed by concurrent small file processing tests using the MapReduce mechanism. Performance evaluation is conducted by monitoring resource utilization through Hadoop ResourceManager, as well as Prometheus and Grafana monitoring systems to obtain real-time performance visualizations. The results of this study are expected to provide insights into the effectiveness of resource distribution in Hadoop Multi-Node Clusters for handling small files and to serve as a reference for the development and evaluation of distributed computing systems in academic and research environments.*

Keywords: *Big Data; Hadoop; Multi-Node Cluster; Resource Distribution; Small Files.*

Abstrak. Pertumbuhan Big Data yang sangat pesat menuntut sistem komputasi yang mampu mengolah data secara efisien dan terdistribusi. *Apache Hadoop* sebagai *framework* komputasi terdistribusi banyak digunakan untuk mengatasi keterbatasan sistem terpusat, namun masih menghadapi tantangan dalam menangani *small files* yang dapat menyebabkan meningkatnya *overhead* pengelolaan berkas serta ketidakseimbangan pemanfaatan sumber daya. Penelitian ini bertujuan menganalisis performa *Hadoop Multi-Node Cluster* pada skenario pengolahan *small files*, khususnya dari aspek distribusi beban kerja serta pemanfaatan sumber daya CPU dan memori pada setiap *node*. Metode penelitian dilakukan dengan mengimplementasikan *Hadoop Multi-Node Cluster* pada lingkungan lokal dengan konfigurasi satu *master node* dan lima *worker node*, kemudian melakukan pengujian pengolahan *small files* secara bersamaan menggunakan mekanisme *MapReduce*. Evaluasi performa dilakukan melalui pemantauan pemanfaatan sumber daya menggunakan *Hadoop ResourceManager*, serta sistem pemantauan *Prometheus* dan *Grafana* untuk memperoleh visualisasi performa secara *real-time*. Hasil penelitian ini diharapkan dapat memberikan gambaran mengenai efektivitas distribusi sumber daya pada *Hadoop Multi-Node Cluster* dalam menangani *small files*, serta menjadi referensi bagi pengembangan dan evaluasi sistem komputasi terdistribusi di lingkungan akademik maupun penelitian

Kata Kunci: Big Data; Distribusi Sumber Daya; *Hadoop*; Klaster *Multi-Node*; *Small Files.*

1. LATAR BELAKANG

Perkembangan Big Data ditandai dengan meningkatnya volume, kecepatan, dan keragaman data telah menghasilkan kumpulan data berskala besar yang bersumber dari aktivitas digital seperti media sosial, transaksi daring, perangkat *Internet of Things*, dan sensor industri. Data terus berkembang dengan cepat dan menciptakan tantangan dalam pengolahan dan pemanfaatannya (Buhl et al., 2013). Untuk mengolah Big Data, sistem komputasi harus mampu menyimpan, mengelola, dan memproses data secara efisien dan tepat waktu agar dapat menghasilkan informasi yang bernilai bagi organisasi dan juga pengguna (Setyowati et al., 2021). Namun, pengelolaan data besar dengan berbagai format ini memerlukan infrastruktur yang lebih canggih dan terkelola dengan baik.

Naskah Masuk: 03 Mei 2026; Revisi: 10 Juni 2026; Diterima: 21 Juli 2026; Tersedia: 29 Juli 2026

Sistem komputasi tradisional yang mengandalkan model terpusat tidak lagi cukup untuk menangani beban kerja dalam skala besar (Helwan University et al., 2021). Pemrosesan data yang bergantung pada satu mesin utama cenderung menyebabkan *bottleneck*. *Bottleneck* merupakan penumpukan beban yang menghambat kinerja sistem secara keseluruhan. Ketergantungan pada satu titik pemrosesan juga meningkatkan risiko kegagalan tunggal (*single point of failure*), di mana gangguan pada satu komponen atau mesin dapat menghentikan seluruh proses (Aggarwal et al., 2022). Dengan meningkatnya volume data yang perlu diproses, sistem terpusat menjadi kurang efisien dan sulit berkembang (*scalable),* sehingga tidak lagi memadai untuk memenuhi tuntutan pengolahan Big Data yang semakin kompleks.

Komputasi terdistribusi atau *distributed computing* menawarkan solusi dengan memanfaatkan beberapa komputer atau *node* yang bekerja secara paralel dalam satu jaringan (Helwan University et al., 2021). Setiap *node* bertanggung jawab untuk bagian tertentu dari penyimpanan dan pemrosesan data yang memungkinkan pembagian beban kerja yang lebih merata dan tidak hanya meningkatkan kecepatan pemrosesan data, tetapi juga memperkuat keandalan sistem karena kegagalan pada satu *node* tidak langsung mempengaruhi seluruh sistem (Zagan & Danubianu, 2021). Permasalahan ini dapat diselesaikan dengan menggunakan *Apache Hadoop*, yang memungkinkan pemrosesan data besar secara paralel dan efisien dalam *cluster* komputer, serta mengatasi kendala yang ada pada sistem terpusat (Ferdiansyah & Nasution, 2023).

Namun, pemrosesan Big Data menggunakan *Hadoop* masih menghadapi permasalahan seperti *small files*. Banyaknya file berukuran kecil yang tersebar dalam jumlah besar memerlukan penanganan khusus karena setiap file harus dikelola secara terpisah oleh sistem penyimpanan. Kondisi ini menimbulkan beban tambahan yang cukup besar sehingga membebani sistem penyimpanan dan jaringan serta mengurangi kelancaran proses pemrosesan (El-Sayed et al., 2019). Ketika *Hadoop* menangani *small files*, sumber daya sistem yang digunakan untuk mengelola file-file tersebut menjadi signifikan, meskipun ukuran data yang diproses relatif kecil.

Hadoop Distributed File System (HDFS) dirancang untuk mengelola file berukuran besar melalui mekanisme pembagian block dengan ukuran sekitar 128 MB pada setiap *block*. Ketika sistem menangani ribuan file kecil, HDFS tetap harus mengelola setiap file tersebut secara terpisah. Semakin banyak jumlah *small files* yang disimpan, semakin besar pula beban kerja yang harus ditangani oleh sistem. Dampaknya tidak hanya dirasakan pada sistem penyimpanan, tetapi juga dapat mempengaruhi pembagian tugas pada *Yet Another Resource Negotiator* (YARN). Banyaknya file berukuran kecil dapat menghasilkan proses yang lebih

banyak sehingga distribusi beban kerja antar *worker node* menjadi kurang merata. Kondisi ini berpotensi meningkatkan latensi awal pemrosesan dikarenakan sistem harus mengalokasikan lebih banyak proses sebelum job berjalan secara stabil.

Pengelolaan *small files* yang tidak optimal dapat menyebabkan *bottleneck* dalam distribusi data antar *node* dalam kluster *Hadoop* (Aggarwal et al., 2022). Setiap *node* harus mengalokasikan waktu dan sumber daya yang signifikan untuk memproses file-file kecil tersebut, yang memperlambat seluruh proses pemrosesan data. Beberapa solusi yang dapat diterapkan untuk mengatasi hal ini antara lain menggabungkan *small files* menjadi file yang lebih besar atau menggunakan format penyimpanan yang lebih terstruktur, seperti HDFS (*Hadoop Distributed File System*), untuk mengurangi *overhead* dan memperbaiki pengelolaan data besar.

Penelitian ini bertujuan untuk mengimplementasikan *Apache Hadoop* sebagai *framework* untuk komputasi terdistribusi, yang memungkinkan pemrosesan secara paralel dalam *cluster komputer*. Penelitian ini difokuskan untuk menganalisis distribusi sumber daya antara master *node* dan *worker node* dalam *Hadoop Multi-Node Cluster* yang diterapkan di lingkungan lokal. Fokus utama penelitian adalah menguji bagaimana *Hadoop* membagi beban kerja antar *node*, serta mengoptimalkan penggunaan CPU dan memori dalam pengolahan data besar. Dengan demikian, hasil penelitian diharapkan dapat memberikan kontribusi dalam pengembangan sistem komputasi terdistribusi yang lebih baik dan dapat dijadikan referensi untuk penelitian lanjutan di bidang pengelolaan sumber daya komputasi.

2. METODE PENELITIAN

Penelitian ini menggunakan metode eksperimental untuk menganalisis performa Hadoop *Multi-Node Cluster* pada skenario pemrosesan *small files*. Implementasi sistem dilakukan menggunakan *Apache Hadoop* pada lingkungan lokal dengan konfigurasi satu *master node* dan lima *worker node*. Pengujian dilakukan melalui beberapa skenario untuk mengevaluasi distribusi beban kerja, pemanfaatan sumber daya, serta kemampuan sistem dalam mempertahankan proses pengolahan data ketika terjadi gangguan pada *worker node*. Seluruh proses pengujian dipantau menggunakan *Hadoop ResourceManager, Prometheus*, dan *Grafana* sehingga kondisi setiap *node* dapat diamati secara *real-time* (Kretzer et al., 2025). Tahapan penelitian meliputi perancangan arsitektur sistem, penyusunan data uji, penentuan parameter pengujian, pemantauan performa sistem, dan pelaksanaan skenario pengujian.

Arsitektur *Hadoop Multi-Node Cluster*

Penelitian ini menggunakan arsitektur *Hadoop Multi-Node Cluster* yang terdiri atas satu *master node* dan lima *worker node*. Arsitektur tersebut dipilih karena mampu mendistribusikan proses penyimpanan dan pengolahan data ke beberapa *node* sehingga beban kerja tidak terpusat pada satu komputer (Bakni & Assayad, 2025). Melalui pendekatan ini, setiap *node* memiliki peran yang berbeda namun saling terintegrasi dalam menjalankan proses *MapReduce* sehingga pemanfaatan sumber daya dapat berlangsung secara lebih terdistribusi.

Master *node* berfungsi sebagai pusat pengelolaan seluruh cluster dengan menjalankan layanan *NameNode, ResourceManager*, dan *JobHistory Server. NameNode* bertanggung jawab mengelola informasi penyimpanan *file* pada *Hadoop Distributed File System* (HDFS), termasuk informasi lokasi penyimpanan blok data serta struktur direktori yang terdapat pada sistem penyimpanan *Hadoop* (Putra et al., 2021). *Resource Manager* bertugas mengelola alokasi sumber daya dan melakukan penjadwalan tugas kepada setiap *worker node* selama proses MapReduce berlangsung sehingga distribusi pekerjaan dapat dilakukan sesuai kapasitas sumber daya yang tersedia (Saadoon et al., 2021). Selain itu, *JobHistory* Server digunakan untuk menyimpan riwayat eksekusi *job* sehingga hasil pemrosesan dapat dianalisis setelah seluruh proses selesai.

Setiap *worker node* menjalankan layanan *DataNode* dan *NodeManager*. *DataNode* bertugas menyimpan blok data yang didistribusikan oleh HDFS sehingga proses penyimpanan dapat dilakukan secara terdistribusikan pada seluruh *node* (Gupta et al., 2025). Sementara itu, *NodeManag*er bertanggung jawab menjalankan *container* serta mengeksekusi tugas *MapReduce* yang diberikan oleh *ResourceManager* (Zhang et al., 2024). Pembagian fungsi tersebut memungkinkan proses pengolahan data dilakukan secara paralel sehingga distribusi beban kerja antar *worker node* dapat diamati selama proses pengujian (Hong et al., 2016).

Seluruh *node* dihubungkan dalam satu jaringan lokal sehingga komunikasi data antar *node* dapat berlangsung secara langsung. Ketika *job MapReduce* dijalankan, *master node* akan mendistribusikan pekerjaan kepada setiap *worker node* berdasarkan mekanisme penjadwalan yang dijalankan oleh YARN (Saadoon et al., 2021). Setelah proses pemetaan (*Map*) dan penggabungan data (*Reduce*) selesai dilakukan oleh *worker node*, hasil pemrosesan dikembalikan kepada *master node* sebagai keluaran akhir proses *MapReduce* (Hong et al., 2016).

Selama proses pengolahan data berlangsung, aktivitas setiap *node* dipantau menggunakan *Hadoop ResourceManager*, *Prometheus*, dan *Grafana*. *Hadoop ResourceManager* digunakan untuk mengamati status *job*, penggunaan *container*, serta kondisi

setiap *node* di dalam *cluster* (Kretzer et al., 2025). *Prometheus* berfungsi mengumpulkan metrik penggunaan CPU, RAM, disk, aktivitas jaringan, dan *load average* dari setiap *node*, sedangkan *Grafana* digunakan untuk memvisualisasikan seluruh metrik tersebut dalam bentuk *dashboard* secara *real-time* sehingga perubahan performa sistem dapat diamati dengan lebih mudah (Shafiyah et al., 2022).

Tabel 1. Spesifikasi Perangkat Keras.

Komponen	***Master Node***	***Worker Node***
Jumlah *Node*	1	5
Proessor	Intel® Core™ i5-14400	Intel® Core™ i5-14400
VCPU	4	2
RAM	12 Gb	12 Gb
Storage	30 Gb	25 Gb

Tabel 1 menunjukkan spesifikasi perangkat keras yang digunakan pada penelitian ini. Seluruh skenario pengujian menggunakan spesifikasi perangkat yang sama sehingga perubahan performa sistem dipengaruhi oleh variasi jumlah *worker node*, jumlah *small files*, maupun kondisi gangguan pada *worker node*. Penggunaan spesifikasi yang seragam juga bertujuan menjaga konsistensi hasil pengujian sehingga setiap perubahan parameter yang diamati merepresentasikan pengaruh skenario pengujian yang diterapkan.

Data Uji

Data uji yang digunakan pada penelitian ini berupa file teks (*.txt*) dengan ukuran masing-masing sebesar 21,8 KB. File berukuran kecil dipilih karena mampu merepresentasikan karakteristik *small files* pada *Hadoop* yang menjadi fokus penelitian. Penggunaan ukuran file yang seragam bertujuan menghilangkan pengaruh variasi ukuran data sehingga analisis difokuskan pada pengaruh jumlah file terhadap performa *Hadoop Multi-Node Cluster*(Giri & Sharma, 2022).

Jumlah file divariasikan menjadi 100 file, 500 file, 1.000 file, 2.000 file, dan 10.000 file. Variasi tersebut digunakan untuk mengevaluasi perubahan performa *Hadoop* ketika jumlah *small files* yang diproses semakin meningkat. Penambahan jumlah file diharapkan dapat memperlihatkan perubahan penggunaan sumber daya, aktivitas komunikasi antar *node*, serta waktu penyelesaian proses *MapReduce* (Perwiratama, 2024).

Tabel 2. Data Uji.

Jumlah File	**Total Ukuran Data**
1 File	21,8 KB
100 File	2,18 MB

500 File	10,9 MB
1000 File	21,8 MB
2000 File	43,6 MB
10000 File	218 MB

Berdasarkan Tabel 2, seluruh data uji menggunakan ukuran file yang sama sehingga variasi pengujian hanya dipengaruhi oleh jumlah *small files* yang diproses. Dengan pendekatan tersebut, perubahan performa *Hadoop* yang diamati berasal dari peningkatan jumlah informasi penyimpanan file pada HDFS, distribusi pekerjaan pada YARN, serta aktivitas komunikasi antar *node* selama proses *MapReduce* berlangsung.

Parameter Pengujian

Evaluasi performa *Hadoop Multi-Node Cluster* dilakukan menggunakan beberapa parameter yang mewakili kondisi komputasi, penggunaan sumber daya, aktivitas jaringan, serta waktu penyelesaian proses. Parameter tersebut diamati selama proses *MapReduce* berlangsung untuk memperoleh gambaran mengenai perubahan performa sistem pada setiap skenario pengujian (Singh et al., 2024).

Penggunaan beberapa parameter secara bersamaan bertujuan memberikan evaluasi yang lebih menyeluruh. Selain mengamati waktu penyelesaian proses, penelitian ini juga mengevaluasi bagaimana *Hadoop* memanfaatkan CPU, memori, media penyimpanan, serta aktivitas komunikasi antar *node* ketika jumlah *worker node* maupun jumlah *small files* mengalami perubahan (Mandala Putra et al., 2021).

Tabel 3. Parameter Pengujian.

No	Parameter	Keterangan
1	CPU	Mengukur tingkat penggunaan prosesor pada *master node* dan *worker node* selama proses *MapReduce* berlangsung.
2	RAM	Mengukur penggunaan memori pada setiap *node* selama proses pengolahan data.
3	Disk	Mengamati aktivitas dan penggunaan media penyimpanan selama proses pengolahan data.
4	*Load Average*	Mengukur rata-rata beban sistem berdasarkan jumlah proses yang sedang berjalan atau menunggu untuk dieksekusi oleh CPU dan disk I/O.
5	*Network Rx*	Menunjukkan jumlah atau kecepatan data yang diterima (masuk) oleh suatu *node* dari jaringan. Parameter ini digunakan untuk mengamati aktivitas penerimaan data selama proses komunikasi antar-*node*, terutama pada fase *Shuffle and Sort*.
6	*Network Tx*	Menunjukkan jumlah atau kecepatan data yang dikirim (keluar) oleh suatu *node* ke jaringan. Parameter ini digunakan untuk mengamati aktivitas pengiriman data antar-*node* selama proses *Shuffle and Sort*.
7	*Latency*	Mengukur waktu respons sistem. Semakin kecil nilai *latency*, semakin cepat sistem memberikan respons.
8	Waktu Eksekusi	Mengukur durasi yang dibutuhkan *cluster* untuk menyelesaikan proses atau *job*.

Parameter pada Tabel 3 digunakan untuk mengevaluasi performa *Hadoop Multi-Node Cluster* dari aspek komputasi, penggunaan sumber daya, komunikasi jaringan, serta waktu penyelesaian proses. Seluruh parameter diamati selama proses *MapReduce* berlangsung sehingga perubahan performa sistem pada setiap skenario pengujian dapat dianalisis secara menyeluruh (Sudirman et al., 2024).

Perangkat Pemantauan

Pemantauan performa Hadoop *Multi-Node Cluster* dilakukan menggunakan *Hadoop ResourceManager, Prometheus*, dan *Grafana. Hadoop ResourceManager* digunakan untuk memantau status *job*, penggunaan *container*, serta aktivitas setiap *node* selama proses *MapReduce* berlangsung (Kretzer et al., 2025). *Prometheus* berfungsi mengumpulkan metrik penggunaan CPU, RAM, disk, *load average*, dan aktivitas jaringan dari setiap *node* dalam *cluster*. Selanjutnya, *Grafana* digunakan untuk memvisualisasikan seluruh metrik tersebut dalam bentuk *dashboard* sehingga perubahan performa sistem dapat diamati secara *real-time* (Julia et al., 2024).

Data hasil pemantauan digunakan sebagai dasar untuk menganalisis distribusi beban kerja, pemanfaatan sumber daya, aktivitas komunikasi antar *node*, serta waktu penyelesaian proses pada setiap skenario pengujian. Dengan demikian, perubahan performa *Hadoop Multi-Node Cluster* dapat dievaluasi secara lebih terukur (Bakni & Assayad, 2024).

Skenario Pengujian

Pengujian dilakukan melalui tiga skenario yang dirancang untuk mengevaluasi performa *Hadoop Multi-Node Cluster* dari aspek distribusi beban kerja, pengaruh jumlah *small files*, serta kemampuan sistem dalam mempertahankan proses pengolahan data ketika terjadi gangguan pada *worker node*. Ketiga skenario tersebut menggunakan parameter pengujian yang sama sehingga hasil yang diperoleh dapat dibandingkan secara konsisten.

Skenario Variasi Jumlah *Worker Node*

Skenario pertama dilakukan untuk menganalisis pengaruh jumlah *worker node* terhadap performa *Hadoop Multi-Node Cluster*. Pada skenario ini jumlah data dibuat tetap sebanyak 500 *small files*, sedangkan jumlah *worker node* divariasikan mulai dari satu hingga lima *node*. Pengujian ini bertujuan mengevaluasi perubahan distribusi beban kerja, pemanfaatan sumber daya, serta waktu penyelesaian proses *MapReduce* ketika jumlah *worker node* ditingkatkan (Kalia & Gupta, 2021).

Tabel 4. Skenario Variasi Jumlah *Worker Node*

Skenario	*Master Node*	*Worker Node*	Jumlah File
1	1	1	500
2	1	2	500
3	1	3	500
4	1	4	500
5	1	5	500

Skenario Variasi Jumlah *Small Files*

Skenario kedua dilakukan untuk menganalisis pengaruh peningkatan jumlah *small files* terhadap performa *Hadoop Multi-Node Cluster*. Jumlah *worker node* dibuat tetap sebanyak lima *node*, sedangkan jumlah file divariasikan menjadi 100, 500, 1.000, 2.000, dan 10.000 file. Skenario ini digunakan untuk mengevaluasi perubahan penggunaan sumber daya, aktivitas komunikasi antar *node*, serta waktu penyelesaian proses ketika jumlah *small files* yang diproses semakin meningkat (Aggarwal et al., 2022).

Tabel 5. Skenario Variasi Jumlah *Small Files,*

Skenario	*Master Node*	*Worker Node*	Jumlah File
1	1	1	100
2	1	2	500
3	1	3	1000
4	1	4	2000
5	1	5	10000

Skenario *Fault Tolerance*

Skenario ketiga dilakukan untuk mengevaluasi kemampuan *fault tolerance Hadoop Multi-Node Cluster*. Pengujian dilakukan menggunakan 10.000 *small files* dengan konfigurasi lima *worker node*. Selama proses *MapReduce* berlangsung dilakukan dua kondisi pengujian, yaitu mematikan satu *worker node* dan mematikan dua *worker node*. Skenario ini bertujuan menganalisis kemampuan *Hadoop* dalam mendeteksi kegagalan *node*, melakukan *task reallocation*, serta mempertahankan proses pengolahan data melalui mekanisme replikasi HDFS (Gaykar et al., 2022).

Tabel 6. Skenario *Fault Tolerance*.

Skenario	*Master Node*	*Worker Node*	Gangguan	Jumlah File
1	1	5	1 *Worker Node* dimatikan	10.000
2	1	5	2 *Worker Node* di matikan	10.000

3. HASIL DAN PEMBAHASAN

Pengaruh Variasi Jumlah *Worker Node* Terhadap Performa *Hadoop Multi-Node Clustera*

Pengujian Pada Skenario Pertama dilakukan untuk menganalisis pengaruh jumlah *Worker Node* terhadap performa *Hadoop Multi-Node Cluster* dalam memproses 500 *small files*. Pada skenario ini *Master Node* sebanyak satu, sedangkan jumlah *Worker Node* akan divariasikan mulai dari satu hingga lima *node*. Parameter yang akan diamati meliputi penggunaan CPU, RAM, DISK, Load Average, *Network Receive(Rx*), *Network Transmit(Tx), latency,* dan waktu ekesuksi (*Execution Time*). Ringkasan hasil pengujian akan di tunjukan pada Tabel 7 dan Tabel 8.

Tabel 7. Hasil Pengujian *Master Node* pada variasi Jumlah *Worker Node*.

Jumlah *Worker*	CPU (%)	RAM (GB)	DISK (GB)	Load Average	Rx (Mbps)	Tx (Mbps)	*Latency* (s)	Waktu Eksekusi(s)
1	23,07	3,74	24,11	0,82	0,06	0,07	0,32	11
2	49,47	3,63	24,55	2,12	0,13	0,11	0,16	10
3	29,83	3,63	23,94	0,57	0,16	0,18	0,31	13
4	20,57	3,56	23,94	0,87	0,20	0.,4	0,26	13
5	30,54	3,88	24,00	2,96	0,24	0,20	0,24	12

Tabel 8. Rata-rata Penggunaan Sumber Daya *Worker Node* pada variasi Jumlah *Worker Node*.

Jumlah *Worker*	PU Rata-rata (%)	RAM Rata-rata (GB)	Disk Rata-rata (GB)	Load Average	Rx (Mbps)	Tx (Mbps)	Latency
1	14,69	2,49	15,37	0,38	0,08	0,05	0,06
2	17,38	1,41	15,27	0,31	0,20	0,18	0,15
3	20,22	1,53	15,02	0,67	0,33	0,32	0,05
4	27,63	1,41	15,10	1,16	1,39	1,37	0,07
5	19,29	1,55	14,89	0,53	0,31	0,16	0,07

Berdasarkan Tabel 7 dan Tabel 8, variasi jumlah *Worker Node* memberikan pengaruh terhadap performa *Hadoop Multi-Node Cluster* dalam memproses 500 *small files*. Konfigurasi dua *Worker Node* menghasilkan waktu eksekusi tercepat, yaitu 10 detik, sedangkan konfigurasi satu *Worker Node* membutuhkan waktu 11 detik. Ketika jumlah *Worker Node* ditingkatkan menjadi tiga dan empat *node*, waktu eksekusi meningkat menjadi 13 detik, kemudian menurun menjadi 12 detik pada konfigurasi lima *Worker Node*. Hasil tersebut menunjukkan bahwa penambahan jumlah *Worker Node* tidak selalu diikuti oleh peningkatan performa. Meskipun proses komputasi dapat didistribusikan ke lebih banyak *Worker Node*, penambahan jumlah *node* juga meningkatkan aktivitas koordinasi antar *node* pada proses penjadwalan tugas oleh

YARN, distribusi blok data HDFS, serta komunikasi selama fase *Shuffle and Sort*. Akibatnya, muncul *overhead* koordinasi yang menyebabkan waktu penyelesaian proses tidak selalu semakin cepat meskipun jumlah *Worker Node* bertambah.

Selain memengaruhi waktu eksekusi, penambahan jumlah *Worker Node* juga berdampak terhadap penggunaan sumber daya pada *Master Node* maupun *Worker Node*. Penggunaan CPU pada *Master Node* mencapai nilai tertinggi sebesar 49,47% pada konfigurasi dua *Worker Node*, sedangkan rata-rata penggunaan CPU pada *Worker Node* juga mengalami peningkatan dibandingkan konfigurasi satu *Worker Node*. Kondisi tersebut menunjukkan bahwa proses komputasi telah berhasil didistribusikan ke *Worker Node*, Sementara *Master Node* tetap berperan dalam mengelola informasi file pada HDFS, melakukan penjadwalan tugas melalui *ResourceManager*, serta mengoordinasikan komunikasi antar *node*. Penggunaan RAM pada *Master Node* berada pada kisaran 3,56–3,88 GB, sedangkan rata-rata penggunaan RAM pada *Worker Node* relatif stabil pada setiap konfigurasi. Hal ini menunjukkan bahwa kapasitas memori masih mampu mendukung proses pengolahan data tanpa mengalami kekurangan sumber daya meskipun jumlah *Worker Node* bertambah.

Penggunaan disk pada *Master Node* maupun rata-rata *Worker Node* juga tidak mengalami perubahan yang signifikan selama proses pengujian. Kondisi tersebut menunjukkan bahwa penambahan jumlah *Worker Node* tidak menyebabkan peningkatan penggunaan kapasitas penyimpanan secara berarti karena data yang diproses tetap berjumlah 500 *small files*. Aktivitas penyimpanan lebih banyak terjadi pada proses pembacaan dan penulisan blok data di HDFS dibandingkan penambahan kapasitas penyimpanan baru. Sementara itu, nilai *load average* mengalami fluktuasi pada setiap konfigurasi yang menunjukkan adanya perubahan jumlah proses yang dijalankan sistem sesuai dengan beban kerja yang diterima. *Aktivitas Network Receive (Rx)* dan *Network Transmit (Tx)* meningkat seiring bertambahnya jumlah *Worker Node*, yang mengindikasikan semakin intensifnya proses pertukaran data antar *node* selama pemrosesan berlangsung. Meskipun demikian, nilai *latency* tetap berada pada rentang yang relatif stabil sehingga komunikasi antar *node* masih berjalan dengan baik tanpa mengalami hambatan.

Pengaruh Variasi Jumlah *Small File* Terhadap Performa *Hadoop Multi Node Cluster*

Pengujian pada skenario kedua dilakukan untuk menganalisis pengaruh variasi jumlah *small files* terhadap performa *Hadoop Multi-Node Cluster*. Pada skenario ini jumlah *worker node* dibuat tetap sebanyak lima *node*, sedangkan jumlah *small files* divariasikan mulai dari 100, 500, 1.000, 2.000, hingga 10.000 file. Parameter yang diamati meliputi penggunaan CPU, RAM, disk, *load average*, *Network Receive* (Rx), *Network Transmit* (Tx), *latency*, dan waktu

eksekusi (*execution time*). Ringkasan hasil pengujian disajikan pada Tabel 9 dan Tabel 10

Tabel 9. Hasil Pengujian *Master Node* pada Variasi Jumlah *Small Files*.

Jumlah File	CPU (%)	RAM (GB)	DISK (GB)	Load Average	Rx (Mbps)	Tx (Mbps)	Latency (s)	Waktu Eksekusi(s)
100	28,04	4,03	21,17	0,69	0,23	0,10	0,29	12
500	27,37	4,03	21,17	0,75	0,23	0,19	0,24	11
1000	28,79	4,20	21,36	0,91	0,25	0,31	0,33	12
2000	27,11	4,34	21,37	0,80	0,28	0,54	0,31	13
10000	36,67	4,15	21,27	0,60	0,60	2,35	0,32	22

Tabel 10. Rata-rata Penggunaan Sumber Daya *Worker Node* pada Variasi Jumlah *Small Files.*

mlah File	PU Rata-rata (%)	RAM Rata-rata (GB)	Disk Rata-rata (GB)	Load Average	Rx (Mbps)	Tx (Mbps)	Latency
100	18,34	1,71	12,34	0,42	0,11	0,13	0,07
500	16,53	1,78	12,89	0,45	0,14	0,15	0,06
1000	29,26	1,79	12,33	0,42	0,18	0,17	0,07
2000	18,36	1,86	12,33	0,59	0,25	0,20	0,07
10000	26,51	1,68	12,36	0,63	0,89	0,93	0,06

Berdasarkan Tabel 9 dan Tabel 10, peningkatan jumlah *small files* memberikan pengaruh terhadap performa *Hadoop Multi-Node Cluster*, terutama pada waktu eksekusi proses *MapReduce.* Pengolahan 100 file membutuhkan waktu 12 detik, kemudian menurun menjadi 11 detik pada 500 file. Selanjutnya, waktu eksekusi meningkat menjadi 12 detik pada 1.000 file, 13 detik pada 2.000 file, dan mencapai 22 detik pada 10.000 file. Hasil tersebut menunjukkan bahwa semakin banyak jumlah *small files* yang diproses, semakin besar beban yang harus ditangani oleh *Hadoop* sehingga waktu penyelesaian proses cenderung meningkat. Kondisi ini terjadi karena setiap *small file* harus dikelola sebagai informasi file tersendiri di HDFS dan diproses melalui mekanisme *MapReduce*. Bertambahnya jumlah file menyebabkan semakin banyak informasi file yang harus dikelola oleh Name*Node*, meningkatnya proses penjadwalan tugas oleh YARN, serta bertambahnya aktivitas komunikasi selama distribusi dan penggabungan hasil pemrosesan.

Selain memengaruhi waktu eksekusi, peningkatan jumlah *small files* juga berdampak pada penggunaan sumber daya sistem. Penggunaan CPU *Master Node* meningkat secara bertahap dan mencapai nilai tertinggi sebesar 36,67% pada pengolahan 10.000 file, sedangkan rata-rata penggunaan CPU *Worker Node* juga meningkat karena semakin banyak tugas Map yang harus diproses secara bersamaan. Penggunaan RAM pada *Master Node* berada pada

kisaran 4,03–4,34 GB, sedangkan penggunaan RAM *Worker Node* relatif stabil selama seluruh proses pengujian. Hal ini menunjukkan bahwa kapasitas memori masih mampu mendukung proses pengolahan data meskipun jumlah file yang diproses meningkat hingga 10.000 file.

Penggunaan disk pada *Master Node* maupun rata-rata *Worker Node* juga tidak mengalami perubahan yang signifikan selama seluruh pengujian. Kondisi tersebut menunjukkan bahwa peningkatan jumlah *small files* lebih banyak memengaruhi aktivitas baca dan tulis data pada HDFS dibandingkan penggunaan kapasitas penyimpanan baru. Di sisi lain, *aktivitas Network Receive (Rx) dan Network Transmit (Tx)* meningkat secara bertahap, terutama pada pengolahan 10.000 file, yang menunjukkan semakin intensifnya pertukaran data antar *node* selama tahapan *Shuffle and Sort*. Nilai *load average* menunjukkan peningkatan jumlah proses yang harus ditangani sistem, sedangkan nilai *latency* tetap berada pada rentang 0,24–0,33 detik, sehingga komunikasi antar *node* masih berlangsung dengan baik meskipun jumlah file yang diproses semakin besar.

Pengujian *Fault Tolerance Hadoop Multi-Node Cluster*

Pengujian pada skenario ketiga dilakukan untuk menganalisis kemampuan *fault tolerance Hadoop Multi-Node Cluster* ketika terjadi kegagalan pada *worker node* selama proses *MapReduce* berlangsung. Pada skenario ini proses pengolahan data dilakukan menggunakan 10.000 *small files* dengan konfigurasi lima *worker node*. Pengujian dilakukan dalam dua kondisi, yaitu ketika satu *worker node* dimatikan dan ketika dua *worker node* dimatikan saat proses pengolahan data sedang berlangsung. Parameter yang diamati meliputi penggunaan CPU, RAM, disk, *load average*, *Network Receive* (Rx), *Network Transmit* (Tx), *latency*, dan waktu eksekusi (*execution time*). Ringkasan hasil pengujian disajikan pada Tabel 11 dan Tabel 12.

Tabel 11. Hasil Pengujian *Fault Tolerance* dengan Satu *Worker Node* Dimatikan

Node	**CPU (%)**	**RAM (GB)**	**DISK (GB)**	***Load Average***	**Rx (Mbps)**	**Tx (Mbps)**	***Latency***	**Waktu Eksekusi**
Master	19,94	4,29	21,57	0,56	0,35	1,16	0.30	10m 06s
Worker 1	2,71	0,59	13,90	0,00	0,07	0,05	0.05	
Worker 2	19,95	1,66	12,11	0,18	1,90	1,65	0.10	
Worker 3	18,63	2,04	11,95	0,12	1,39	1,57	0.10	
Worker 4	11,72	1,78	11,91	0,22	0,11	1,62	0.04	

Worker 5	21,89	1,79	11,81	0,45	2,99	1,15	0,10	

Tabel 12. Hasil *Pengujian Fault Tolerance* dengan Dua *Worker Node* Dimatikan.

Node	**CPU (%)**	**RAM (GB)**	**DISK (GB)**	***Load Average***	**Rx (Mbps)**	**Tx (Mbps)**	***Latency***	**Waktu Eksekusi**
Master	16,22	4,15	21,46	0,77	0,23	0,24	0.28	
Worker 1	2,06	0,59	13,87	0,00	0,01	0,03	0.05	
Worker 2	10,19	1,66	12,11	0,04	0,10	1,63	0.10	
Worker 3	9,72	2,01	11,95	0,34	0,15	1,31	0.10	10m 10s
Worker 4	21,11	1,80	11,92	0,41	3,01	1,15	0.12	
Worker 5	3,03	0,58	11,81	0,00	0,01	0,03	0,05	

Berdasarkan Tabel 11 dan Tabel 12, *Hadoop Multi-Node Cluster* tetap mampu menyelesaikan proses *MapReduce* meskipun terjadi kegagalan pada satu maupun dua *Worker Node* selama proses pengolahan data berlangsung. Pada kondisi satu *Worker Node* dimatikan, waktu eksekusi yang dibutuhkan untuk menyelesaikan pengolahan 10.000 *small files* adalah 10 menit 06 detik, sedangkan pada kondisi dua *Worker Node* dimatikan waktu eksekusi tercatat 10 menit 10 detik. Waktu eksekusi tersebut jauh lebih lama dibandingkan skenario sebelumnya karena *Hadoop* tidak langsung mendeteksi bahwa *Worker Node* mengalami kegagalan. *ResourceManage*r terlebih dahulu menunggu hingga batas waktu *heartbeat timeout* untuk memastikan bahwa *node* benar-benar tidak lagi aktif. Setelah *node* dinyatakan gagal, *ResourceManager* akan melakukan r*escheduling* dengan mendistribusikan kembali tugas yang belum selesai ke *Worker Node* lain yang masih aktif. Proses deteksi kegagalan, penjadwalan ulang tugas, dan pemulihan inilah yang menyebabkan waktu eksekusi meningkat hingga sekitar sepuluh menit. Meskipun demikian, seluruh proses *MapReduce* tetap berhasil diselesaikan karena HDFS mempertahankan ketersediaan blok data melalui mekanisme replikasi sehingga tidak terjadi kehilangan data selama proses berlangsung.

Selain memengaruhi waktu eksekusi, kegagalan *Worker Node* juga memberikan perubahan terhadap penggunaan sumber daya sistem. Berdasarkan Tabel 8, penggunaan CPU *Master Node* meningkat menjadi 19,94% dengan penggunaan RAM sebesar 4,29 GB dan disk 21,57 GB ketika satu *Worker Node* dimatikan. Pada kondisi ini terlihat bahwa *Worker* 2,

Worker 3, dan *Worker* 5 mengalami peningkatan penggunaan CPU menjadi 19,95%, 18,63%, dan 21,89%, yang menunjukkan bahwa ketiga *node* tersebut menerima distribusi ulang pekerjaan dari *node* yang gagal. *Aktivitas Network Receive (Rx)* dan *Network Transmit (Tx)* pada *Worker Node* tersebut juga meningkat, menandakan terjadinya pertukaran data yang lebih intensif selama proses *rescheduling*. Pada kondisi dua *Worker Node* dimatikan sebagaimana ditunjukkan pada Tabel 9, penggunaan CPU *Master Node* menjadi 16,22%, sedangkan peningkatan beban kerja paling tinggi terjadi pada *Worker* 4 dengan penggunaan CPU sebesar 21,11%. Penggunaan RAM dan disk pada *Master Node* maupun *Worker Node* tetap relatif stabil selama pengujian, sedangkan nilai load average dan latency masih berada pada rentang yang normal sehingga komunikasi antar *node* tetap berlangsung dengan baik selama proses pemulihan. Hal tersebut menunjukkan bahwa Hadoop mampu mengalihkan beban kerja ke *Worker Node* yang masih aktif tanpa memberikan perubahan yang signifikan terhadap penggunaan memori maupun kapasitas penyimpanan.

Pembahasan

Analisis Pengaruh Jumlah Worker Node terhadap Kinerja Hadoop Multi-Node Cluster

Hasil pengujian menunjukkan bahwa variasi jumlah *Worker Node* memberikan pengaruh terhadap performa *Hadoop Multi-Node Cluster* dalam mengelola 500 *small files* menggunakan MapReduce. Konfigurasi dua *Worker Node* menghasilkan waktu eksekusi paling rendah, yaitu 10 detik, sedangkan penambahan jumlah *Worker Node* menjadi tiga, empat, dan lima *node* tidak selalu menurunkan waktu eksekusi. Kondisi tersebut menunjukkan bahwa penambahan jumlah *Worker Node* tidak secara langsung meningkatkan performa sistem. Meskipun proses komputasi dapat didistribusikan ke lebih banyak *node*, setiap penambahan *Worker Node* juga meningkatkan proses koordinasi antar *node*, penjadwalan tugas oleh YARN, distribusi blok data pada HDFS, serta komunikasi selama tahapan *Shuffle and Sort.* Akibatnya, sistem menghasilkan *overhead* yang memengaruhi waktu penyelesaian proses *MapReduce.*

Selain memengaruhi waktu eksekusi, variasi jumlah *Worker Node* juga memengaruhi penggunaan sumber daya sistem. Penggunaan CPU pada *Master Node* maupun *Worker Node* menunjukkan bahwa proses komputasi berhasil didistribusikan ke beberapa *node* selama proses *MapReduce* berlangsung. Sementara itu, penggunaan RAM dan disk relatif stabil pada setiap konfigurasi sehingga peningkatan jumlah *Worker Node* tidak memberikan perubahan yang signifikan terhadap penggunaan memori maupun kapasitas penyimpanan. Aktivitas Network Receive (Rx) dan *Network Transmit (Tx)* meningkat seiring bertambahnya jumlah *Worker Node*, yang menunjukkan bahwa semakin banyak *node* yang terlibat maka semakin besar pula

aktivitas pertukaran data antar *node* selama proses pengolahan berlangsung. Hasil tersebut menunjukkan bahwa performa *Hadoop* tidak hanya dipengaruhi oleh jumlah *Worker Node*, tetapi juga oleh besarnya *overhead* komunikasi yang terjadi selama proses komputasi terdistribusi.

Analisis Pengaruh Jumlah Small Files terhadap Kinerja Hadoop Multi-Node Cluster

Hasil pengujian menunjukkan bahwa jumlah *small files* memberikan pengaruh terhadap performa *Hadoop Multi-Node Cluster*. Pengolahan 500 *small files* menghasilkan waktu eksekusi paling rendah, sedangkan waktu eksekusi meningkat secara bertahap ketika jumlah file bertambah hingga 10.000 *small files*. Kondisi tersebut menunjukkan bahwa semakin banyak *small files* yang diproses, semakin besar beban kerja yang harus ditangani oleh *Hadoop*. Setiap *small file* memiliki informasi pengelolaan file tersendiri pada HDFS sehingga *NameNode* harus mengelola informasi tersebut dalam jumlah yang semakin besar. Selain itu, YARN juga harus melakukan penjadwalan lebih banyak tugas *MapReduce* sehingga meningkatkan aktivitas pemrosesan selama pengolahan data berlangsung.

Peningkatan jumlah *small files* juga memengaruhi penggunaan sumber daya sistem. Penggunaan CPU pada *Master Node* maupun *Worker Node* meningkat ketika jumlah file yang diproses semakin banyak, sedangkan penggunaan RAM dan disk relatif stabil pada seluruh skenario pengujian. *Aktivitas Network Receive (Rx)* dan *Network Transmit (Tx)* meningkat terutama pada pengolahan 10.000 *small files*, yang menunjukkan bahwa pertukaran data antar *node* menjadi lebih intensif selama tahapan *Shuffle and Sort.* Oleh karena itu, peningkatan waktu eksekusi pada pengolahan *small files* lebih dipengaruhi oleh meningkatnya beban pengelolaan informasi file dan komunikasi antar *node* dibandingkan dengan perubahan penggunaan memori maupun kapasitas penyimpanan.

Analisis Kemampuan Fault Tolerance Hadoop Multi-Node Cluster

Hasil pengujian menunjukkan bahwa *Hadoop Multi-Node Cluster* tetap mampu menyelesaikan proses *MapReduce* meskipun terjadi kegagalan pada satu maupun dua *Worker Node*. Pada kedua skenario tersebut, waktu eksekusi meningkat menjadi sekitar 10 menit, yang menunjukkan bahwa *Hadoop* memerlukan waktu tambahan untuk mendeteksi *Worker Node* yang mengalami kegagalan sebelum proses dapat dilanjutkan. Mekanisme *heartbeat* digunakan oleh *ResourceManager* untuk memastikan bahwa *node* benar-benar tidak aktif. Setelah *node* dinyatakan gagal, YARN secara otomatis melakukan *rescheduling* dengan mendistribusikan kembali tugas yang belum selesai kepada *Worker Node* yang masih aktif. Di sisi lain, HDFS mempertahankan ketersediaan data melalui mekanisme replikasi blok sehingga proses *MapReduce* tetap dapat dilanjutkan tanpa kehilangan data. Perubahan

penggunaan CPU pada beberapa *Worker Node* menunjukkan bahwa *node* yang masih aktif menerima tambahan beban kerja setelah terjadi kegagalan *node*. Sementara itu, penggunaan RAM dan disk relatif stabil selama proses pengujian, sedangkan aktivitas *Network Receive (Rx)* dan Network *Transmit (Tx)* meningkat pada *Worker Node* yang menerima hasil *rescheduling*. Nilai load average dan latency juga tetap berada pada kisaran yang stabil sehingga komunikasi antar *node* tetap berlangsung dengan baik selama proses pemulihan. Hasil tersebut menunjukkan bahwa mekanisme *fault tolerance* pada *Hadoop* mampu mempertahankan keberlangsungan proses pengolahan data meskipun sebagian *Worker Node* mengalami kegagalan selama proses berlangsung.

Keterbatasan Penelitian

Penelitian ini masih memiliki beberapa keterbatasan yang perlu diperhatikan. Pengujian dilakukan pada lingkungan *Hadoop Multi-Node Cluster* dengan satu *Master Node* dan maksimal lima *Worker Node*, sehingga hasil penelitian masih terbatas pada konfigurasi *cluster* tersebut. Selain itu, data yang digunakan berupa *small files* dengan jumlah maksimum 10.000 file, sehingga belum menggambarkan performa sistem pada jumlah file yang lebih besar maupun pada variasi ukuran file yang berbeda.

Penelitian ini juga hanya berfokus pada pengukuran penggunaan CPU, RAM, disk*, load average, Network Receive (Rx), Network Transmit (Tx), latency*, dan waktu eksekusi selama proses MapReduce berlangsung. Oleh karena itu, penelitian selanjutnya dapat dilakukan dengan menggunakan konfigurasi *cluster* yang lebih beragam, jumlah *Worker Node* yang lebih banyak, serta variasi jumlah maupun ukuran *small files* yang lebih luas sehingga diperoleh gambaran performa *Hadoop Multi-Node Cluster* pada kondisi beban kerja yang berbeda.

4. KESIMPULAN

Penelitian ini menunjukkan bahwa implementasi *Hadoop Multi-Node Cluster* memberikan pengaruh terhadap performa pengolahan *small files* menggunakan *framework MapReduce*. Hasil pengujian menunjukkan bahwa konfigurasi dua *worker node* menghasilkan waktu eksekusi paling rendah, yaitu 10 detik, sedangkan penambahan jumlah *worker node* tidak selalu meningkatkan performa karena adanya *overhead* koordinasi, komunikasi antar *node*, serta proses *Shuffle and Sort*. Selain itu, peningkatan jumlah *small files* juga memengaruhi waktu eksekusi, di mana pengolahan 500 *small files* menghasilkan waktu eksekusi 11 detik, sedangkan pengolahan 10.000 *small files* membutuhkan waktu 22 detik akibat meningkatnya beban pengelolaan informasi file pada HDFS dan distribusi tugas

MapReduce.

Selain pengujian performa, penelitian ini juga membuktikan bahwa *Hadoop Multi-Node Cluster* memiliki kemampuan *fault tolerance* yang mampu mempertahankan keberlangsungan proses *MapReduce* ketika terjadi kegagalan pada satu maupun dua *worker node*. Melalui mekanisme *heartbeat*, *task rescheduling* pada YARN, dan replikasi data pada HDFS, proses pengolahan data tetap dapat diselesaikan tanpa kehilangan data, meskipun waktu eksekusi meningkat akibat proses deteksi kegagalan dan pendistribusian ulang tugas. Dengan demikian, implementasi *Hadoop Multi-Node Cluster* mampu mendukung pengolahan *small files* menggunakan *MapReduce* serta mempertahankan proses komputasi ketika terjadi kegagalan pada *worker node*.

DAFTAR REFERENSI

Aggarwal, R., Verma, J., & Siwach, M. (2022a). Small files' problem in Hadoop: A systematic literature review. *Journal of King Saud University - Computer and Information Sciences*, *34*(10), 8658–8674. https://doi.org/10.1016/j.jksuci.2021.09.007

Aggarwal, R., Verma, J., & Siwach, M. (2022b). Small files' problem in Hadoop: A systematic literature review. *Journal of King Saud University - Computer and Information Sciences*, *34*(10), 8658–8674. https://doi.org/10.1016/j.jksuci.2021.09.007

Bakni, N.-E., & Assayad, I. (2024). Analysis of the MapReduce Performance in Hadoop. *Revue d'Intelligence Artificielle*, *38*(6), 1391–1397. https://doi.org/10.18280/ria.380601

Bakni, N.-E., & Assayad, I. (2025). Smart Data Placement Strategy in Heterogeneous Hadoop. *HighTech and Innovation Journal*, *6*(1), 42–53. https://doi.org/10.28991/HIJ-2025-06-01-03

Buhl, H. U., Röglinger, M., Moser, F., & Heidemann, J. (2013). Big Data: A Fashionable Topic with(out) Sustainable Relevance for Research and Practice? *Business & Information Systems Engineering*, *5*(2), 65–69. https://doi.org/10.1007/s12599-013-0249-5

El-Sayed, T., Badawy, M., & El-Sayed, A. (2019). Impact of Small Files on Hadoop Performance: Literature Survey and Open Points. *Menoufia Journal of Electronic Engineering Research*, *28*(1), 109–120. https://doi.org/10.21608/mjeer.2019.62728

Gaykar, R. S., Khanaa, V., & Joshi, S. D. (2022). Faulty *Node* Detection in HDFS Using Machine Learning Techniques. *Revue d'Intelligence Artificielle*, *36*(4), 553–560. https://doi.org/10.18280/ria.360406

Giri, P. R., & Sharma, G. (2022). Apache Hadoop Architecture, Applications, and Hadoop Distributed File System. *Semiconductor Science and Information Devices*, *4*(1), 14–20. https://doi.org/10.30564/ssid.v4i1.4619

Gupta, A., Santhiya, P., Thiyagarajan, C., Gupta, A., Gupta, M., & Dwivedi, R. Kr. (2025). The Cutting-Edge Hadoop Distributed File System: Un-leashing Optimal Performance. *ICST Transactions on Scalable Information Systems*, *12*(5). https://doi.org/10.4108/eetsis.9027

Helwan University, Sawiris, B., El-Gaber, S., Helwan University, Abdel-Fattah, M., & Helwan University. (2021). Centralization of Big Data Using Distributed Computing Approach in IoT. *International Journal of Intelligent Engineering and Systems*, *14*(4), 393–409. https://doi.org/10.22266/ijies2021.0831.35

Hong, Z., Xiao-ming, W., Jie, C., Yan-hong, M., Yi-rong, G., & Min, W. (2016). An Optimized Model for MapReduce Based on Hadoop. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, *14*(4), 1552. https://doi.org/10.12928/telkomnika.v14i4.3606

Julia, Mutahari, M. I., Renaldi, & Saepullah. (2024). Analisis Kinerja Basis Data Terdistribusi dalam Linkungan Cloud Computing. *Karimah Tauhid*, *3*(2), 1771–1782. https://doi.org/10.30997/karimahtauhid.v3i2.11907

Kalia, K., & Gupta, N. (2021). Analysis of hadoop MapReduce scheduling in heterogeneous environment. *Ain Shams Engineering Journal*, *12*(1), 1101–1110. https://doi.org/10.1016/j.asej.2020.06.009

Kretzer, A. R., Barreto Vavassori Benitti, F., & Siqueira, F. (2025). Challenges and Opportunities in Big Data Analytics for Industry 4.0: A Systematic Evaluation of Current Architectures. *IEEE Access*, *13*, 183419–183447. https://doi.org/10.1109/ACCESS.2025.3624558

Mandala Putra, H., Akbar, T., Ahmadi, A., & Iman Darmawan, M. (2021). Analisa Performa Klastering Data Besar pada Hadoop. *Infotek: Jurnal Informatika dan Teknologi*, *4*(2), 174–183. https://doi.org/10.29408/jit.v4i2.3565

Perwiratama, R. (2024). Desain Infrastruktur: Kubernetes dan Hadoop sebagai Penyimpanan Data Terdistribusi. *KONSTELASI: Konvergensi Teknologi dan Sistem Informasi*, *4*(2). https://doi.org/10.24002/konstelasi.v4i2.10446

Rahul Dev Singh, Vikram Kumar Gupta, & Priya Anjali Patel. (2024). Optimization Of Big Data Processing Using Distributed Computing In Cloud Environments. *International Journal of Computer Technology and Science*, *1*(2), 01–07. https://doi.org/10.62951/ijcts.v1i2.58

Saadoon, M., Hamid, S. H. A., Sofian, H., Altarturi, H., Nasuha, N., Azizul, Z. H., Sani, A. A., & Asemi, A. (2021). Experimental Analysis in Hadoop MapReduce: A Closer Look at Fault Detection and Recovery Techniques. *Sensors*, *21*(11), 3799. https://doi.org/10.3390/s21113799

Setyowati, L., & Nasir Ahmad, D. (2021). Pemanfaatan Big Data Dalam Era Teknologi 5.0. *ABDINE: Jurnal Pengabdian Masyarakat*, *1*(2), 117–122. https://doi.org/10.52072/abdine.v1i2.205

Shafiyah, S., Ahsan, A. S., & Asmara, R. (2022). Big Data Infrastructure Design Optimizes Using Hadoop Technologies Based on Application Performance Analysis. *SISTEMASI*, *11*(1), 55. https://doi.org/10.32520/stmsi.v11i1.1510

Sudirman, A., Irawan, & Saharuna, Z. (2024). Pengembangan Sistem Big Data: Rancang Bangun Infrastruktur dengan Framework Hadoop. *Journal of Informatics and Computer Engineering Research*, *1*(1), 25–32. https://doi.org/10.31963/jicer.v1i1.4919

Veri Ferdiansyah & Muhammad Irwan Padli Nasution. (2023). Penerapan Teknologi Big Data Dalam Pengembangan Database Pendidikan. *Jurnal Riset Manajemen*, *1*(3), 22–29. https://doi.org/10.54066/jurma.v1i3.591

Zagan, E., & Danubianu, M. (2021). HADOOP: A Comparative Study between Single-*Node* and Multi-*Node* Cluster. *International Journal of Advanced Computer Science and Applications*, *12*(2). https://doi.org/10.14569/IJACSA.2021.0120207

Zhang, D., Dai, Z.-Y., Sun, X.-P., Wu, X.-T., Li, H., Tang, L., & He, J.-H. (2024). A distributed data processing scheme based on Hadoop for synchrotron radiation experiments. *Journal of Synchrotron Radiation*, *31*(3), 635–645. https://doi.org/10.1107/S1600577524002637