E-ISSN : 2407 – 3911
P-ISNN : 2686 - 0333

KOMPRESI FILE MENGGUNAKAN KONVERSI BINER HEXADECIMAL DAN ALGORITMA HUFFMAN ENCODING

Karlo Geofandy [1], Erlando Aubrey N [2], Halim Agung [3]
Program Studi Teknik Informatika[1,2,3]
Universitas Bunda Mulia
Jl. Lodan Raya No 2 Ancol, Jakarta
karlogeofandy@gmail.com [1], erlando.aubrey.nathaniel@gmail.com [2], hagung@bundamulia.ac.id [3]

Abstrak

Seiring dengan perkembangannya, teknologi data dalam bentuk *file* terus berkembang dan mengalami peningkatan ukuran dari waktu ke waktu. Perkembangan teknologi yang semakin maju dengan penambahan jumlah pengguna komputer yang semakin banyak menyebabkan ledakan jumlah data serta tingkat perpindahan data dari satu perangkat ke perangkat lain. Data - data tersebut umumnya dikompresi terlebih dahulu agar proses pertukaran data tidak memakan waktu yang terlalu lama. Metode yang akan digunakan pada penelitian ini adalah algoritma *Huffman Encoding* dan konversi Biner *Hexadecimal*. algoritma *Huffman Encoding* adalah salah satu algoritma kompresi dengan cara melakukan pengkodean dalam bentuk bit untuk mewakili data karakter, sedangkan konversi Biner *Hexadecimal* adalah teknik untuk mengubah rangkaian bit yang akan dikonversi menjadi bilangan *hexadecimal*. Hasil dari penelitian ini adalah algoritma *Huffman Encoding* dan konversi Biner *Hexadecimal* dapat diimplementasikan untuk kompresi *file*. Simpulan dari aplikasi ini yaitu metode *Huffman Encoding* dan konversi Biner *Hexadecimal* dapat melakukan mengkompresi dengan baik pada 26 macam ekstensi *file* akan tetapi, metode ini tidak lebih baik dibandingkan program kompresi WinRAR dan WinZIP. Ini telah dibuktikan dengan pengujian keterbatasan dan peforma program terhadap 50 jenis *file* yang berbeda.

Kata kunci:

Kompresi, *file*, *Huffman encoding*, *hexadecimal*, biner, *server*.

Abstract

Along with its development, data technology in the form of files continues to grow and increase in size over time. The development of increasingly advanced technology by increasing the number of computer users which is increasingly causing an explosion in the amount of data and the rate of data transfer from one device to another. The data is generally compressed first so that the data exchange process does not take too long. The method that will be used in this research is the Huffman Encoding algorithm and Binary Hexadecimal conversion. Huffman Encoding algorithm is a compression algorithm by coding in the form of bits to represent character data, while Hexadecimal Binary conversion is a technique to change the sequence of bits to be converted into hexadecimal numbers. The results of this study are the Huffman Encoding algorithm and Hexadecimal Binary conversion can be implemented for file compression. The conclusion of this application is the Huffman Encoding method and Binary Hexadecimal conversion can compress well on 26 file extensions, however, this method is no better than the WinRAR and WinZIP compression programs. This has been proven by testing the limitations and performance of programs against 50 different file types.

Keywords :

Toll Road, Compression, file, Huffman encoding, hexadecimal, binary, server.

I. PENDAHULUAN

Kompresi adalah kegiatan yang dilakukan untuk mengurangi ukurannya menjadi lebih kecil dari sebelumnya. Kompresi dibuat karena kurangnya kapasitas penyimpanan yang memadai. Kompresi *file* juga diperlukan untuk mempercepat aktivitas pengiriman data antar jaringan komputer. Kompresi memiliki aturan yang berbeda antara

kecepatan dan kepadatan (Marlina, Siahaan, Kurniawan, & Sulistianingsih, 2017).

Seiring dengan perkembangannya, teknologi data dari *file-file* terus berkembang dan mengalami peningkatan ukuran menjadi semakin besar dari waktu ke waktu. Terdapat berbagai tipe data yang dapat dimuat yakni teks, gambar, audio dan video.

Perkembangan teknologi yang semakin maju dan penambahan jumlah pengguna komputer yang semakin banyak menyebabkan ledakan data serta perpindahan data dari satu perangkat ke perangkat lain. Data-data tersebut umumnya dikompresi terlebih dahulu agar proses pertukaran tidak memakan waktu yang terlalu lama.

Pada penelitian ini, penulis hendak memadukan algoritma *Huffman Encoding* dengan teknik *Biner to Hexadecimal* dengan harapan dapat memaksimalkan hasil kompresi.

II. KAJIAN LITERATUR

II.1 Sistem Bilangan

Sistem bilangan secara sederhana didefinisikan sebagai cara mudah berhitung. Bilangan atau sistem perhitungan awal yang digunakan manusia dikembangkan untuk membantu menentukan jumlah harta yang seseorang miliki. Sejak manusia menemukan perlunya menghitung benda-benda, mereka mencari cara termudah untuk berhitung, sekitar tahun 1672 Gottfried Wilhelm von Leibnez menyempurnakan mesin yang dapat melakukan semua operasi dasar (Mutiara & Handayani, 2015). Mesin ini yang kemudian digunakan pada sistem komputer.

Lain halnya dengan sistem komputer, sistem komputer merupakan sebuah sistem terintegrasi dari beberapa elemen perangkat lunak dan perangkat keras yang dioperasikan oleh pengguna. Pengintegrasian elemen-elemen tersebut tak lepas dari suatu konsep representasi informasi untuk diimplementasikan dan dilibatkan dalam sistem komputer. Representasi informasi dalam sistem komputer berperan sangat penting dalam melakukan pengolahan data dan informasi (Mutiara & Handayani, 2015). Sistem komputer ini dapat diklasifikasikan sebagai berikut:

1. Bilangan *Decimal*: Bilangan desimal merupakan bilangan yang direpresentasikan untuk pengguna, dimana pengguna melakukan segala komputasi berdasarkan bilangan desimal. Bilangan desimal berbasis sepuluh menggunakan sepuluh simbol bilangan yaitu: "0", "1", "2", "3", "4", "5", “6", "7", "8" dan "9” (Mutiara & Handayani, 2015).
2. Bilangan Biner: Bilangan biner merupakan bilangan dasar sistem komputer berbasis dua. Oleh karena itu bilangan biner menggunakan 2 simbol bilangan yaitu, "0" dan"1" (Mutiara & Handayani, 2015)
3. Bilangan *Octal*: Bilangan oktal merupakan bilangan yang digunakan komputer untuk representasi eksternal yang bertujuan membatasi panjang *string* dan mempermudah pengguna membaca representasi bilangan biner melalui bilangan oktal (Mutiara & Handayani, 2015)
4. Bilangan *Hexadecimal*: Seperti halnya bilangan oktal, bilangan *hexadecimal* merupakan bilangan yang digunakan komputer untuk representasi eksternal yang bertujuan membatasi panjang *string*. Bilangan *hexadecimal* mempermudah pengguna membaca representasi bilangan biner melalui bilangan *hexadecimal*. Bilangan heksadesimal memiliki basis enam belas dengan simbol bilangan yang digunakan terdiri dari "0”, "1", "2", "3", “4", "5", “6", "7", "8", "9", "A", "B", "C", "D", "E", dan "F” (Mutiara & Handayani, 2015)

II.2 Teknik Kompresi

Sampai saat ini setidaknya ada 2 teknik dalam mengkompresi *file*, yaitu:

a) *Lossy compression*: Dalam teknik kompresi *Lossy*, *lossy* mengurangi bit dengan mengenali informasi yang tidak diperlukan dan dengan menghilangkan bit tersebut. Dalam metode ini beberapa kerugian informasi dapat diterima. Menghilangkan informasi yang tidak penting dari sumber data dapat menghemat area penyimpanan (Singh, Kumar, Chouhan, & Shrivastava, 2016)

b) *Lossless compression*: Dalam teknik kompresi *lossless* dengan kompresi dari data yang ketika didekompresi, akan menjadi replika yang sama dengan data aktual. Dalam hal ini, ketika data biner seperti dokumen, aplikasi, gambar dll. Kompresi Ini harus direproduksi persis ketika didekompresi lagi. Sebaliknya, gambar dan musik juga tidak perlu dihasilkan tepat. Kemiripan gambar sebenarnya adalah cukup untuk yang paling objektif, sejauh kesalahan atau masalah antara gambar aktual dan gambar terkompresi dapat dihindari atau lumayan (Singh, Kumar, Chouhan, & Shrivastava, 2016)

II.3 Teknik Kompresi

Algoritma merupakan suatu metode khusus yang tepat dan terdiri dari serangkaian langkah yang terstruktur dan dituliskan secara sistematis yang akan dikerjakan untuk menyelesaikan suatu masalah dengan bantuan komputer. Algoritma merupakan suatu prosedur untuk melakukan satu tugas spesifik berupa gagasan dibalik suatu program.

Sebuah algoritma kompresi dapat dievaluasi dalam sejumlah cara yang berbeda. Yang diukur adalah kompleksitas relatif dari algoritma kompresi, memori yang diperlukan untuk melaksanakan algoritma, seberapa cepat algoritma melakukan pada mesin yang diberikan, jumlah kompresi, dan bagaimana erat rekonstruksi menyerupai aslinya, berikut beberapa algoritma *lossless compression*:

1. *Elias Delta Code* ditemukan oleh Peter Elias. Kode ini juga menerapkan metode *Gamma code*, terutama di kepala. Teknik-tekniknya adalah sebagai berikut:
   a. Cari peringkat tertinggi dari biner, misalnya desimal 11 jika ditetapkan ke 1011 di mana peringkat tertinggi adalah 3. Jadi N' = 3.
   b. Gunakan *Gamma Coding* untuk menyandikan angka N di mana N = N' + 1. Jadi untuk kasus desimal 11 maka kita harus membuat *Gamma Coding* dari 4 yaitu 00100.
   c. Tambahkan sisa biner N' ke hasil # 2. Jadi jawabannya adalah 00100011.

   Prinsipnya adalah kebalikan dari langkah satu hingga tiga di atas (Hariyanto & Siahaan, 2018).
2. *Huffman Encoding*: Algoritma Huffman adalah algoritma yang dikembangkan David A. Huffman pada 1952. Algoritma Huffman menggunakan prinsip pengkodean yang mirip dengan kode morse, yaitu tiap karakter (*symbol*) dikodekan hanya dengan rangkaian beberapa bit, dimana karakter yang sering muncul dikodekan dengan rangkaian bit yang pendek dan karakter yang jarang muncul dikodekan dengan rangkaian bit yang lebih panjang, karena prosesnya yang menggunakan kode ini, membuat algoritma Huffman sebagai algoritma keluarga dengan variable *codeword length* (Prayoga & Suryaningrum, 2018). Cara kerjanya:
   a. Menghitung banyaknya jenis karakter dan jumlah dari masing-masing karakter yang terdapat dalam sebuah *file*.
   b. Menyusun setiap jenis karakter dengan urutan jenis karakter yang jumlahnya paling sedikit ke yang jumlahnya paling banyak.
   c. Membuat pohon biner berdasarkan urutan karakter dari yang jumlahnya terkecil ke yang terbesar, dan memberi kode untuk tiap karakter.
   d. Mengganti data yang ada dengan kode bit berdasarkan pohon biner.
   e. Menyimpan jumlah bit untuk kode bit yang terbesar, jenis karakter yang diurutkan dari frekuensi keluarnya terbesar ke terkecil beserta data yang sudah berubah menjadi kode bit sebagai data hasil kompresi.

III. METODOLOGI PENELITIAN

III.1 Perancangan Sistem

Pada pembangunan sistem akan digunakan model pengembangan *waterfall*, dikarenakan model pengembangan *waterfall* melaksanakan proses pembangunan sistemnya secara bertahap dan tidak ada proses dikerjakan secara bersamaan. Berikut adalah tahapan-tahapan yang dilakukan pada perancangan sistem dengan model *waterfall*:

a) Analisis Kebutuhan
   i. Analisis Kebutuhan Fungsional
      1. Sistem dapat membaca *binary file* dan mengkonversi menjadi bilangan *hexadecimal*.
      2. Sistem dapat melakukan kompresi data terhadap bilangan *hexadecimal* yang telah dikonversi dengan menggunakan algoritma *Huffman Encoding*.
      3. Sistem dapat mengembalikan data yang telah dikompresi
   ii. Analisis Kebutuhan Non Fungsional

      Kebutuhan perangkat keras (*Hardware*) yang diperlukan untuk mengimplementasikan sistem minimal memiliki spesifikasi sebagai berikut:
      1. *Processor* min. intel core i5 inside 7200U
      2. Memori : 8GB
      3. Sistem Operasi: Microsoft Windows
b) Desain Sistem

Pada tahapan desain sistem, dirancang bagaimana tampilan antarmuka dari sistem atau aplikasi agar mudah digunakan oleh *user*. Tidak hanya desain antar muka yang dirancang, diagram - diagram yang menggambarkan bagaimana sistem bekerja dan berinteraksi dengan *user* juga dirancang pada tahapan desain sistem berupa *use case diagram* dan *activity diagram*

c) Penulisan Kode Program

Bahasa pemrograman yang digunakan dalam merancang/membangun aplikasi adalah bahasa pemrograman Javascript yang berjalan pada *server* Node.js. Aplikasi yang digunakan untuk menulis kode program adalah Visual Studio Code. Untuk menjalankan aplikasi menggunakan *browser*.

d) Kompresi

Kompresi dilakukan dengan mengombinasikan algoritma *Huffman Encoding* yang digunakan untuk melakukan kompresi data dengan cara tiap karakter (simbol) dikodekan hanya dengan rangkaian beberapa bit, dimana karakter yang sering muncul dikodekan dengan rangkaian bit yang pendek dan karakter yang jarang muncul dikodekan dengan rangkaian bit yang lebih panjang dan teknik *Biner to Hexadecimal* untuk mengubah rangkaian bit yang akan dikonversi menjadi bilangan *hexadecimal*, dimana *hexadecimal* memiliki panjang bit yang lebih sedikit

e) Dekompresi

File yang telah berhasil dikompresi akan disimpan menjadi *file* lain dengan ekstensi yang baru, untuk menggunakannya *file* harus terlebih dahulu didekompresi dengan menggunakan algoritma *Huffman Encoding* dan mengembalikan data *hexadecimal* menjadi *binary* kembali

II.1 PERANCANGAN PROSES

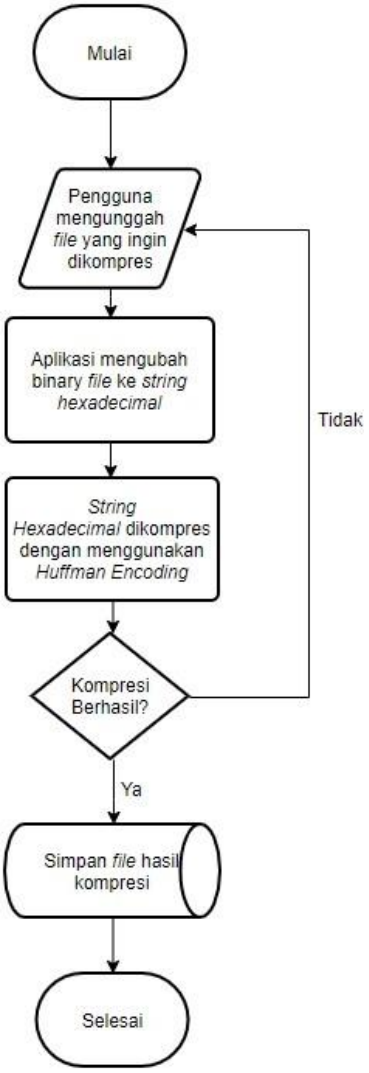
a) Flowchart Diagram

Flowchart Diagram pada gambar 1.a menjelaskan alur proses saat melakukan pengkompresian. Berikut merupakan penjelasannya:

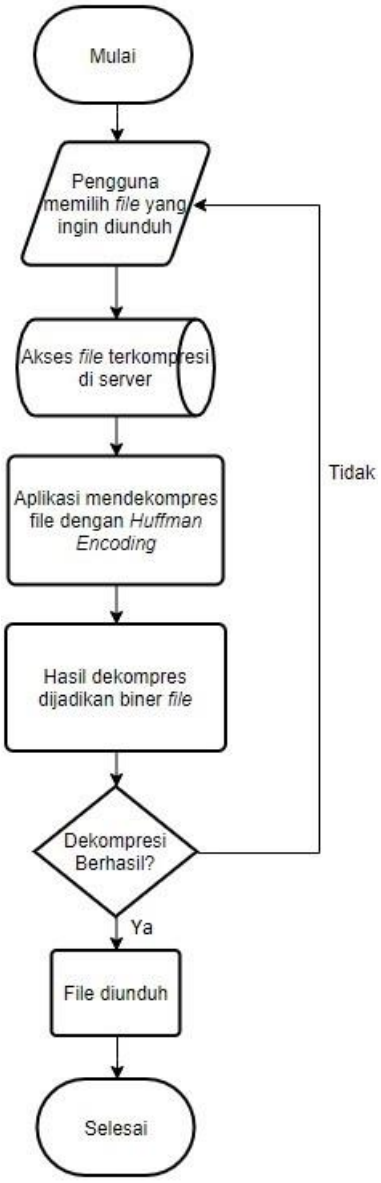
1. User mengunggah file yang telah dipilih,
2. Program membaca data binary dari file dan diubah menjadi data hexadecimal.
3. Data hexadecimal tersebut akan dikompresi menggunakan Huffman Encoding
4. Data disimpan pada server.

Flowchart Diagram pada gambar 1.b menjelaskan alur proses saat melakukan pengdekompresian. Berikut merupakan penjelasannya:

1. User memilih file yang akan didekompres,
2. File akan didekompres menggunakan Huffman Encoding.
3. Data dikembalikan menjadi binary file
4. file siap untuk diunduh.



Gambar 1 *Flowchart* alur sistem kompresi

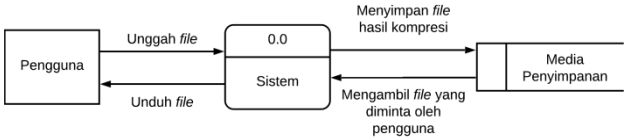


Gambar 2 *Flowchart* alur sistem dekompresi

a) Data Flow Diagram

Pada gambar 3 menjelaskan Data Flow Diagram level 0, Berikut merupakan penjelasannya:

1. User mengunggah file yang akan dikompres ke sistem.
2. Program menyimpan hasil kompresi.
3. User mengirimkan permintaan untuk mengunduh file
4. Program melakukan dekompresi file yang diminta user.
5. File dikirimkan kembali ke pengguna.



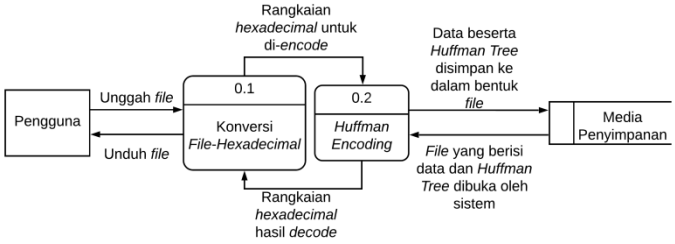
Gambar 3 *Data Flow Diagram level* 0

Data flow diagram level 1 pada gambar 4 menjelaskan proses dalam sistem yang dibagi menjadi 2 proses, yaitu konversi biner *hexadecimal* dan *Huffman Encoding*. Berikut merupakan penjelasan proses kompresi:

1. Untuk mengkompresi *File* yang diunggah, data *binary* pada *file* dikonversi menjadi *hexadecimal*.
2. *File* dikompresi menggunakan *Huffman Encoding*.
3. Sistem menyimpan *key* dan data ke media penyimpanan.

Proses dekompresi dijelaskan sebagai berikut:

1. Untuk mendekompresi *File* yang sudah tersimpan, dilakukan *Huffman Decoding* sesuai dengan *key* dan data yang sudah disimpan.
2. Hasil dekompresi dibangun kembali *file*nya dari *hexadecimal*.
3. Hasil *Huffman Decoding* kemudian dikirimkan ke *user*.



Gambar 4 *Data Flow Diagram level* 1

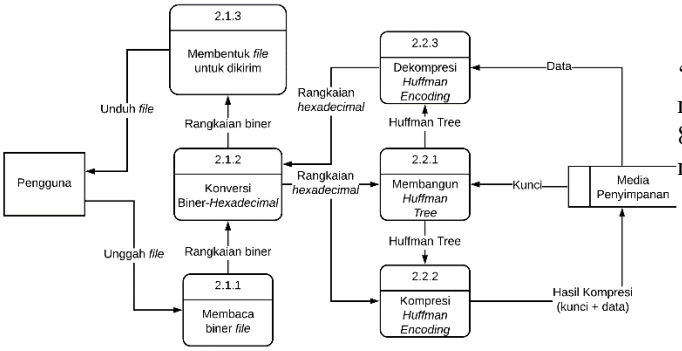
Pada gambar 5, *Data Flow Diagram level* 2 digambarkan lebih ringkas. Berikut penjelasan dari saat mengkompresi:

1. Biner *file* dibaca,
2. Setiap 4 bit dari *file*, diubah ke dalam bentuk *hexadecimal*.
3. Dilakukan pembuatan *Huffman Tree* untuk dijadikan *key* dari *Huffman Encoding* yang akan dilakukan dan *Huffman Decoding* nantinya,

4. *Key* dan data disimpan menjadi 1 *file* ke dalam media penyimpanan.

Untuk proses dekompresi, dijelaskan sebagai berikut:

1. *File* yang sudah tersimpan di media penyimpanan, dipecah menjadi kunci dan data.
2. Data diproses menggunakan *Huffman Decoding* yang menghasilkan serangkaian *hexadecimal*.
3. Setelah itu serangkaian tersebut dibuat menjadi *file* yang kemudian dikirimkan ke pengguna yang memintanya.



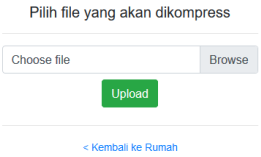
Gambar 5 *Data Flow Diagram level* 2

III.2 Tampilan Antarmuka

Ketika *user* menjalankan program, program akan menampilkan halaman awal seperti yang terlihat pada Gambar 6. Pada tampilan halaman awal, *user* akan memilih antara melakukan *compress files* atau *decompress files.*

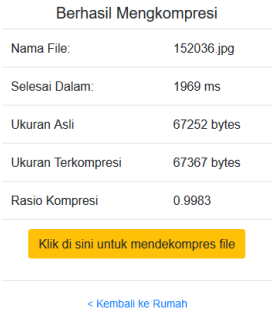
Gambar 6 Antarmuka Halaman Awal

Pada halaman *compress*, *user* akan memilih *file* yang akan dikompres seperti yang terlihat pada Gambar 7. Pada tampilan, tombol '*Browse'* digunakan untuk memilih *file* yang akan dikompres, setelah *file* telah dipilih. Tekan tombol '*Upload'* untuk memulai kompresi.



Gambar 7 Antarmuka Halaman Kompres

File yang berhasil dikompres akan disimpan pada folder '*upload'* dengan ekstensi .hxhf dan program akan menampilkan rincian kompresi seperti terlihat pada Gambar 8. Tombol "Klik di sini untuk mendekompres *file*" akan mengarahkan *user* ke halaman decompress.



Gambar 8 Antarmuka Halaman Rincian Kompres

Pada halaman dekompres, *user* dapat melihat daftar *file* yang telah berhasil dikompres sebelumnya seperti terlihat pada Gambar 9. *User* juga dapat mendekompres *file* itu kembali ke bentuk awal seperti sebelum dikompresi dengan menekan nama *file* yang berwarna biru

Gambar 9 Antarmuka Halaman Dekompres

IV. ANALISIS DAN PERANCANGAN

Tabel 1 Analisa Keterbatasan Program

No	Ukuran *File* (*Byte*)	Ekstensi	Waktu Kompresi (Ms)	Ukuran Kompresi (*Byte*)	Hasil
1	31.222.057	.ai	0	0	GAGAL
2	226.918	.ai	8.646	224.445	BERHASIL
3	98.622.667	.psd	0	0	GAGAL
4	7.976.792	.exe	0	0	GAGAL
5	409.600	.exe	7.913	379.807	BERHASIL
6	524.288	.bin	9.109	419.777	BERHASIL
7	118.128	.css	3.213	105.041	BERHASIL
8	769.568	.dat	21.970	769.706	BERHASIL
9	30.740	.html	781	25.525	BERHASIL
10	4.662	.html	321	4.357	BERHASIL
11	353.118	.ico	6.732	254.475	BERHASIL
12	3.027.933	.jpg	84.823	3.028.079	BERHASIL
13	143.573	.jpg	5.575	143.703	BERHASIL
14	11.226	.jpg	445	11.340	BERHASIL
15	262.152	.json	8.812	202.709	BERHASIL
16	479	.key	23	438	BERHASIL
17	7.972	.xlsx	310	7.252	BERHASIL
18	19.612	.xlsx	765	19.345	BERHASIL
19	2.570.322	.pptx	84.938	2.570.468	BERHASIL
20	2.066.191	.pptx	38.137	2.057.184	BERHASIL
21	1.492.306	.docx	25.002	1.492.452	BERHASIL
22	17.442.472	.docx	390.723	17.442.634	BERHASIL
23	150.697	.docx	2.978	150.827	BERHASIL
24	91.082.298	.mkv	-	-	GAGAL
25	17.426.531	.mkv	353.179	17.426.693	BERHASIL
26	7.469.755	.mp3	135.766	7.469.903	BERHASIL
27	3.256.320	.mp3	56.620	3.256.466	BERHASIL
28	3.000.253	.mp4	53.807	3.000.399	BERHASIL
29	19.195.672	.mp4	463.835	19.195.834	BERHASIL
30	1.100.260	.apk	19.327	1.100.406	BERHASIL
31	50.206.216	.apk	-	-	GAGAL
32	13.648	.otf	295	12.963	BERHASIL
33	6.555.683	.pdf	133.681	6.491.434	BERHASIL
34	105.058	.pdf	1.817	99.523	BERHASIL
35	6.058.918	.pdf	117.713	6.059.064	BERHASIL
36	610.359	.pdf	11.510	610.489	BERHASIL
37	77.444	.pkt	1.331	77.558	BERHASIL
38	6.197.404	.png	122.657	6.197.550	BERHASIL
39	519	.png	29	576	BERHASIL
40	10.578	.png	198	10.692	BERHASIL
41	1.462	.txt	76	1.346	BERHASIL
42	100.494	.txt	2.120	83.920	BERHASIL
43	349.680	.ttf	7.475	286.298	BERHASIL
44	144.156	.ttf	2.864	121.384	BERHASIL
45	24.846.336	.msi	-	-	GAGAL
46	11.336.116	.zip	202.468	11.336.278	BERHASIL
47	77.615	.zip	1.264	77.729	BERHASIL
48	5.939.144	.zip	101.448	5.939.290	BERHASIL
49	52.914.928	.exe	-	-	GAGAL
50	50.240	.docx	910	49.974	BERHASIL

Tabel 2 Analisa Peforma Program

No	Ukuran *File* (*Byte*)	Ekstensi	Hasil ZIP	Hasil RAR	Hasil Program	Kesimpulan
1	46.200	docx	43.768	43.632	46.315	LEBIH BESAR
2	226.918	.ai	172.974	170.973	224.445	LEBIH BESAR
3	409.600	.exe	212.725	171.341	379.807	LEBIH BESAR
4	524.288	.BIN	240.493	219.983	419.777	LEBIH BESAR
5	118.128	.css	16.920	15.758	105.041	LEBIH BESAR
6	769.568	.dat	769.848	361.276	769.706	LEBIH BESAR
7	30.740	.html	3.640	3.517	25.525	LEBIH BESAR
8	4.662	.html	1.510	1.433	4.357	LEBIH BESAR
9	353.118	.ico	41.160	52.047	254.475	LEBIH BESAR
10	3.027.933	.jpg	2.990.660	2.978.472	3.028.079	LEBIH BESAR
11	143.573	.jpg	142.261	142.521	143.703	LEBIH BESAR
12	11.226	.jpg	10.712	10.642	11.340	LEBIH BESAR
13	262.152	.json	41.263	35.808	202.709	LEBIH BESAR
14	479	.key	479	435	438	LEBIH BESAR
15	7.972	.xlsx	5.665	5.602	7.252	LEBIH BESAR
16	19.612	.xlsx	14.947	14.808	19.345	LEBIH BESAR
17	2.570.322	.pptx	2.402.444	2.391.147	2.570.468	LEBIH BESAR
18	2.066.191	.pptx	1.682.387	1.649.032	2.057.184	LEBIH BESAR
19	1.492.306	.docx	1.452.944	1.453.130	1.492.452	LEBIH BESAR
20	17.442.472	.docx	16.913.122	16.461.875	17.442.634	LEBIH BESAR
21	150.697	.docx	145.434	145.404	150.827	LEBIH BESAR
22	17.426.531	.mkv	17.375.319	17.363.324	17.426.693	LEBIH BESAR
23	7.469.755	.mp3	7.451.667	7.464.444	7.469.903	LEBIH BESAR
24	3.256.320	.mp3	3.225.424	3.224.395	3.256.466	LEBIH BESAR
25	3.000.253	.mp4	2.962.204	2.958.646	3.000.399	LEBIH BESAR
26	19.195.672	.mp4	19.065.616	19.078.921	19.195.834	LEBIH BESAR

No	Ukuran *File* (*Byte*)	Ekstensi	Hasil ZIP	Hasil RAR	Hasil Program	Kesimpulan
27	1.100.260	.apk	1.090.806	1.092.059	1.100.406	LEBIH BESAR
28	13.648	.otf	10.043	9.770	12.963	LEBIH BESAR
29	6.555.683	.pdf	5.059.007	4.875.768	6.491.434	LEBIH BESAR
30	105.058	.pdf	60.579	60.490	99.523	LEBIH BESAR
31	6.058.918	.pdf	5.459.131	5.433.677	6.059.064	LEBIH BESAR
32	610.359	.pdf	599.450	599.605	610.489	LEBIH BESAR
33	77.444	.pkt	77.625	77.519	77.558	LEBIH BESAR
34	6.197.404	.png	6.198.491	6.197.579	6.197.550	LEBIH KECIL
35	519	.png	630	587	576	LEBIH KECIL
36	10.578	.png	10.739	10.648	10.692	LEBIH BESAR
37	1.462	.txt	893	840	1.346	LEBIH BESAR
38	100.494	.txt	26.095	25.619	83.920	LEBIH BESAR
39	349.680	.ttf	153.794	101.598	286.298	LEBIH BESAR
40	144.156	.ttf	66.199	63.319	121.384	LEBIH BESAR
41	11.336.116	.zip	11.334.111	11.336.374	11.336.278	LEBIH BESAR
42	77.615	.zip	77.796	77.690	77.729	LEBIH BESAR
43	5.939.144	.zip	5.929.047	5.939.361	5.939.290	LEBIH BESAR
44	50.240	.docx	44.815	44.716	49.974	LEBIH BESAR
45	109.505	.png	109.244	109.105	109.635	LEBIH BESAR
46	1.537	.txt	472	397	1.340	LEBIH BESAR
47	4.976.165	.pdf	4.576.439	4.550.148	4.976.311	LEBIH BESAR
48	46.200	.docx	43.768	43.632	46.315	LEBIH BESAR
49	1.404	.txt	364	288	736	LEBIH BESAR
50	3.359.542	.txt	1.221.450	1.014.672	2.879.426	LEBIH BESAR

V. KESIMPULAN

Memuat beberapa kesimpulan penting dari hasil penelitian dan mencantumkan saran untuk perbaikan penelitian berikutnya.

Referensi dituliskan menggunakan style APA fifth edition, yaitu Nama penulis, tahun, judul, sumber [jurnal, prosiding, bagian buku, artikel dari internet dan lain-lain.

Referensi ditampilkan sesuai urutan pencantuman pada artikel, bukan berdasarkan alphabet penulis.

REFERENSI

Afyenni, R. (2014). Perancangan Data Flow Diagram Untuk Sistem Informasi Sekolah (Studi Kasus Pada SMA Pembangunan Laboratorium UNP). *Jurnal TEKNOIF*, *2*(1), 35-39.

Ardiyanto, D., & Purwoto, B. H. (2014). Kompresi Citra dengan Menggunakan Metode Delta Modulation. *Jurnal Emittor*, *14*(1), 1-12.

Chulkamdi, M. T., Pramono, S. H., & Yudaningtyas, E. (2015, Juni). Kompresi Teks Menggunakan Algoritma Huffman dan Md5 pada Instant Messaging Smartphone Android. *Jurnal EECCIS*, *9*(1), 103-108.

Dennis, A., Wixom, B. H., & Tegarden, D. (2015). *Systems Analysis and Design* (Vol. 5). New York: Wiley.

Hariyanto, E., & Siahaan, A. P. (2018). Design of Adaptive Compression Algorithm Elias Delta Code and Huffman. *International Journal For Innovative Research In Multidisciplinary Field*, *4*(10), 78-87.

Hidayat, T., Zakaria, M. H., & Pee, A. C. (2018, December). Comparison of Lossless Compression Schemes for WAV Audio Data 16-Bit between Huffman and Coding Arithmetic. *International Journal of Simulation Systems, Schience & Technology*, *19*(6), 36.1-36.7.

Jambek, A. B., & Khairi, N. A. (2014). Performance Comparison of Huffman and Lempel-Ziv Welch Data Compression for Wireless Sensor Node Application. *American Journal of Applied Sciences*, *11*(1), 119-126.

Kaur, S., & Singh, S. (2016). Entropy Coding and Different Coding Techniques. *Journal of Network Communications and Emerging Technologies (JNCET)*, *6*(5), 4-7.

Krasmala, R., Purba, A. B., & Lenggana, U. (2017, Juni). Kompresi Citra dengan MenggabungkanMetode Discrete Cosine Transform (DCT) dan Algoritma Huffman. *Jurnal Online Informatika*, *2*(1-9), 1.

Marlina, L., Siahaan, A. P., Kurniawan, H., & Sulistianingsih, I. (2017). Data Compression Using Elias Delta Code. *International Journal of Recent Trends in Engineering & Research*, *3*(8), 210-217.

Marzuki, I. (2017). Aplikasi Kompresi Untuk Pengiriman Data Menggunakan Metode LZW (Lemple Ziv Welch). *Energy*, *7*(2), 38-43.

Muslihudin, M., & Oktafianto. (2016). *Analisis dan Perancangan Sistem Informasi Menggunakan Model Terstruktur dan UML*. Yogyakarta: Penerbit Andi.

Mutiara, G. A., & Handayani, R. (2015). *Sistem Komputer Representasi Data*. Sleman: Deepublish.

Prayoga, E., & Suryaningrum, K. M. (2018). Implementasi Algoritma Huffman Dan Run Length Encoding Pada Kompresi Berbasis Web. *Jurnal Ilmiah Teknologi Informasi Terapan*, *14*(1), 92-101.

Purwaningsih, F., & Badrul, M. (2017). Penerapan Algoritma Huffman Untuk Aplikasi Pengamanan Sms Berbasis Android. *Jurnal PROSISKO*, *4*(2), 60-66.

Raharja, B. D., & Harsadi, P. (2018). Implementasi Kompresi Citra Digital Dengan Mengatur Kualitas Citra Digital. *Jurnal Ilmiah SINUS (JIS)*, *16*(2), 71-77.

Santoso, & Nurmalina, R. (2017). Perencanaan dan Pengembangan Aplikasi Absensi Mahasiswa Menggunakan Smart Card Guna Pengembangan Kampus Cerdas (Studi Kasus Politeknik Negeri Tanah Laut). *Jurnal Integrasi*, *9*(1), 84-91.

Singh, M., Kumar, S., Chouhan, S. S., & Shrivastava, M. (2016). Various Image Compression Techniques: Lossy and Lossless. *International Journal of Computer Applications*, *142*(6), 23-26.

Sitorus, L. (2015). *Algoritma dan pemrograman.* Yogyakarta: Penerbit Andi.

Yana, G. I., & Hondro, R. K. (2016, Desember). Implementasi Algoritma Huffman dan LZ78 untuk Kompresi Data. *Jurnal Riset Komputer (JURIKOM)*, *3*(6), 42-44.

Yaragunti, H. S., & Reddy, T. (2015, December). Text Based Image Compression Using Hexadecimal Conversion. *Oriental Journal of Computer Science & Technology*, *8*(3), 216-221.