

Analisis Perbandingan Performa Kinerja Metode *Huffman Coding* dan Metode *Elias Omega Coding* pada Kompresi File Teks

Atifa Nuzulul Kuraini^{1*}, Arnita², Elmanani Simamora³, Said Iskandar Al Idrus⁴, Insan Taufik⁵

^{1*,2,3,4,5}Program Studi Ilmu Komputer Universitas Negeri Medan

*Email: atifakuraini@gmail.com

Abstract

This study analyzes the performance comparison between the Huffman Coding and Elias Omega Coding methods in the context of text file compression. The research includes testing various types of text files to evaluate the compression ratio, processing speed, and memory usage of both methods. The goal of this study is to simulate the compression and decompression of text using the Huffman Coding and Elias Omega Coding methods and present a comparison of their performance effectiveness. The results show that the Huffman Coding algorithm performs better in compressing and decompressing corpus files, with an average compression ratio of 1.625, a compression ratio of 0.607, space saving of 39.3%, compression time of 499.547 ms, and decompression time of 9,384.501 ms. On the other hand, the Elias Omega Coding algorithm has an average compression ratio of 0.917, a compression ratio of 1.099, a space saving of -9.9%, compression time of 2,937.606 ms, and decompression time of 10,539.144 ms.

Keywords: *Huffman coding, elias omega Coding, teks compression, compression ratio, performance*

Abstrak

Penelitian ini menganalisis perbandingan performa kinerja antara metode *Huffman Coding* dan *Elias Omega Coding* dalam konteks kompresi file teks. Studi ini mencakup pengujian terhadap berbagai jenis file teks untuk mengevaluasi rasio kompresi, kecepatan proses, dan penggunaan memori dari kedua metode. Tujuan Penelitian ini melakukan simulasi pada kompresi dan dekompresi teks dengan metode *Huffman Coding* dan *Elias Omega Coding* serta menunjukkan hasil perbandingan efektifitas performa mana yang lebih baik. Hasil penelitian menunjukkan bahwa algoritma *huffman code* lebih baik dalam melakukan kompresi dan dekompresi terhadap *file corpus* dengan rata-rata *ratio of compression* sebesar 1,625, *compression ratio* sebesar 0,607, *space saving* sebesar 39,3%, waktu kompresi sebesar 499,547 ms, dan waktu dekompresi sebesar 9.384,501 ms. Sedangkan, algoritma *Elias Omega Code* memiliki rata-rata *ratio of compression* sebesar 0,917, *compression ratio* sebesar 1,099, *space saving* sebesar -9,9%, waktu kompresi sebesar 2.937,606 ms, dan waktu dekompresi sebesar 10.539,144 ms.

Kata kunci: *Huffman coding, elias omega coding, kompresi teks, rasio kompresi, performa kinerja*

1. Pendahuluan

Perkembangan teknologi dan informasi yang begitu pesat saat ini memungkinkan pengiriman data dilakukan dengan sangat cepat. Namun, kecepatan transfer informasi sangat bergantung pada ukuran data yang dikirim. Semakin besar kapasitas informasi, semakin lama proses pengirimannya, dan ini juga akan berdampak pada kebutuhan ruang penyimpanan yang lebih besar. Ukuran berkas yang besar bisa menyebabkan kegagalan saat proses transfer karena kapasitas penyimpanan pada media mungkin tidak mencukupi [1]. Untuk mengatasi permasalahan tersebut, salah satu solusi yang dapat diterapkan adalah kompresi berkas teks sebelum dilakukan pengiriman. Kompresi ini bertujuan untuk mengurangi ukuran berkas dengan cara menghilangkan data redundan tanpa mengurangi informasi penting yang terkandung di dalamnya [2]. Dengan demikian, kompresi dapat menghemat ruang penyimpanan serta mempercepat proses transfer data.

Menurut [3] Algoritma kompresi menyimpan data dalam bentuk paling ringkas berdasarkan kemampuan metode tersebut. Namun, metode tersebut harus menghasilkan sejumlah kecil informasi yang menjelaskan isi dari apa yang disimpan. Jika *file* sudah dipadatkan dengan baik, sehingga metode tersebut tidak dapat mengompres *file* tersebut lebih lanjut, maka hasil kompresi tidak akan dibuat lebih kecil, dan ukuran *file* tersebut akan sedikit lebih besar karena sedikitnya jumlah informasi tambahan yang dibuat oleh metode tersebut. Dengan prinsip dasar dari kompresi adalah dengan cara mengurangi duplikasi data yang terdapat di dalam berkas dokumen tersebut. Sehingga ruang penyimpanan yang dibutuhkan untuk merepresentasikan berkas dokumen menjadi lebih sedikit dibandingkan representasi awal atau yang tidak terkompresi [4].

Metode Huffman menggunakan prinsip pengkodean yang mirip dengan kode Morse, yaitu tiap karakter (simbol) dikodekan hanya dengan rangkaian bit yang pendek dan karakter

yang jarang muncul dikodekan dengan rangkaian bit yang lebih panjang.

Penelitian terdahulu mengungkap kelebihan pada metode *Huffman Coding* adalah pembuatan pohon yang efektif sehingga kompresi teks dapat menjadi lebih minimum (berkerja berdasarkan prinsip kerja algoritma *greedy*) [5].

Metode *Elias Omega* diperkenalkan pada tahun 1975 oleh Peter Elias. Metode ini berbeda dengan metode *Elias Delta Code* dan *Elias Gamma Code*, metode *Elias Omega Code* menggunakan dirinya sendiri secara rekursif untuk mengkodekan awalan (*prefix*) *M* [6]. Menurut sumber *Wikipedia*, kelebihan pada metode *Elias Omega* adalah metode ini dapat berjalan secara rekursif mengkodekan awalan tersebut, biasanya disebut dengan *recursive Elias codes*.

2. Tinjauan Pustaka

2.1. Kompresi Data

Kompresi data adalah teknik memampatkan data dimana ukuran data akan lebih kecil dari ukuran aslinya. Dengan kompresi data, ukuran berkas dapat dikurangi. Ini akan berguna saat memproses, menyimpan, atau mentransfer berkas besar, yang membutuhkan banyak sumber daya [7].

Banyaknya informasi secara digital dan ukuran yang dibutuhkan untuk menampilkannya semakin meningkat, kompresi data masih diperlukan [8]. Data digital dipresentasikan dengan angka yang diwakili oleh *byte* data. Jumlah *byte* yang dibutuhkan untuk merepresentasikan data bisa sangat besar sehingga kapasitas ruang penyimpanan yang digunakan semakin besar. Proses transmisi data besar akan memakan waktu yang lama.

Agar dapat diterjemahkan oleh komputer, sebuah *file* harus mengikuti unit standar penyimpanan yaitu *byte*. *Byte* terdiri dari sekumpulan 8 bit untuk merepresentasikan satu karakter. Proses *encoding* memungkinkan jumlah bit yang dihasilkan tidak kelipatan 8, maka dari itu perlu dilakukan penambahan bit

yaitu dengan *padding* bit dan *flag* bit agar hasil kompresi dituliskan ke dalam *file* [9].

2.2. File Teks

File teks adalah *file* yang hanya berisi data teks yaitu sekumpulan karakter berupa huruf, angka dan tanda baca. Set karakter yang umumnya digunakan dalam sistem komputer modern yaitu ASCII. Set karakter ASCII (*American Standard Code for Information Interchange*) terdiri dari 256 karakter termasuk huruf beraksen dan simbol khusus, dimana tiap karakter direpresentasikan dalam 8 bit [10]. Jenis *file* teks dalam penelitian ini yaitu teks biasa atau disebut *plain text*. *Plain text* merupakan jenis yang hanya berisi sekumpulan karakter berupa huruf, angka dan tanda baca tanpa adanya format khusus seperti tebal, miring, ukuran huruf dan sebagainya. *Plain text* umumnya berekstensi *.txt, namun sebenarnya banyak ekstensi lain untuk jenis file ini seperti *.html, *.xml, *.lsp, *.c dan sebagainya.

2.3. Metode Huffman Coding

David Huffman awalnya mengembangkan metode pengkodean karakter menggunakan pohon biner sederhana. Namun, kemudian ia menyadari bahwa dengan menerapkan algoritma *greedy*, ia dapat membangun kode awalan yang lebih optimal. Dalam konteks ini, algoritma *greedy* digunakan dalam proses pemilihan dua pohon dengan frekuensi terendah untuk digabungkan dalam pohon Huffman. Pendekatan *greedy* ini bertujuan untuk meminimalkan total biaya yang diperlukan dalam proses penggabungan, di mana biaya dihitung berdasarkan frekuensi akar dari dua pohon yang digabungkan. Dengan demikian, total biaya pembentukan pohon Huffman merupakan akumulasi dari biaya-biaya yang terjadi selama proses penggabungan pohon-pohon tersebut. Oleh karena itu, *metode huffman* dapat dianggap sebagai salah satu contoh metode kompresi yang memanfaatkan algoritma *greedy*.

2.4. Elias Omega Coding

Elias Omega (ω) *Code* merupakan salah satu dari tiga algoritma kompresi data yang ditemukan Peter Elias pada 1975. *Elias Omega Code* adalah kode universal yang mengodekan bilangan bulat positif secara rekursif sehingga terkadang disebut sebagai kode Elias rekursif (Rosmaida & Hondro, 2022). Algoritma *Elias Omega Code* adalah yang terpendek diantara algoritma Elias lain dalam hal panjang *codeword* serta memiliki struktur rekursif [11].

2.5. Parameter Kinerja Kompresi Dta

Untuk mengukur seberapa baik pada algoritma kompresi adalah dengan melihat rasio jumlah bit sebelum dikompresi dan setelah dikompresi [12].

Kinerja kompresi data dapat diukur berdasarkan parameter berikut:

a. Ratio of Compression (RC)

Ratio of Compression (RC) adalah rasio yang menggambarkan perbandingan antara ukuran bit data sebelum kompresi dengan ukuran bit data setelah proses kompresi. Rumus berikut untuk menghitung rasio ini secara matematis.

$$R_c = \frac{\text{Ukuran Bit Data Sebelum Dikompresi}}{\text{Ukuran Bit Data Setelah Dikompresi}}$$

Misalnya, jika nilai *Ratio of Compression* (RC) sebesar 2.89, ini berarti ukuran data sebelum dikompresi adalah 2.89 kali lipat ukuran data setelah dikompresi.

b. Compression Ratio (CR)

Compression Ratio (CR) adalah perbandingan dalam bentuk persentase antara ukuran bit data setelah kompresi dengan ukuran bit data sebelum kompresi. Secara matematis, rumus untuk menghitung *Compression Ratio* (CR) adalah sebagai berikut [13].

$$C_R = \frac{\text{Ukuran Bit Data Setelah Dikompresi}}{\text{Ukuran Bit Data Sebelum Dikompresi}} \times 100\%$$

Misalnya, jika nilai *Compression Ratio* (CR) sebesar 54%, berarti ukuran data setelah dikompresi adalah 54% dari data sebelum dikompresi.

c. *Space Saving* (SS)

Space Saving (SS) adalah persentase selisih antara ukuran data sebelum kompresi dengan ukuran data setelah kompresi. Secara matematis, rumus untuk menghitung *Space Saving* (SS) dapat dituliskan sebagai berikut [14].

$$SS = 1 - C_R$$

Misalnya, jika nilai *Space Saving* (SS) sebesar 49.5%, ini berarti persentase penghematan ruang untuk data terkompresi adalah 49.5%.

d. *Time Proses* (Waktu Kompresi dan Dekompresi)

Time Process merujuk pada waktu yang diperlukan oleh sistem untuk melakukan proses kompresi dan dekompresi. Semakin kecil waktu yang dibutuhkan untuk kedua proses tersebut, maka semakin efisien kinerja algoritma yang digunakan.

3. Metoda Penelitian

3.1. Pengumpulan Data

Data yang digunakan dalam pembuatan sistem aplikasi kompresi dengan metode *Huffman Coding* dan metode *Elias Omega Coding* adalah file dari *The Canterbury Corpus*. Berikut file dari *The Canterbury Corpus*:

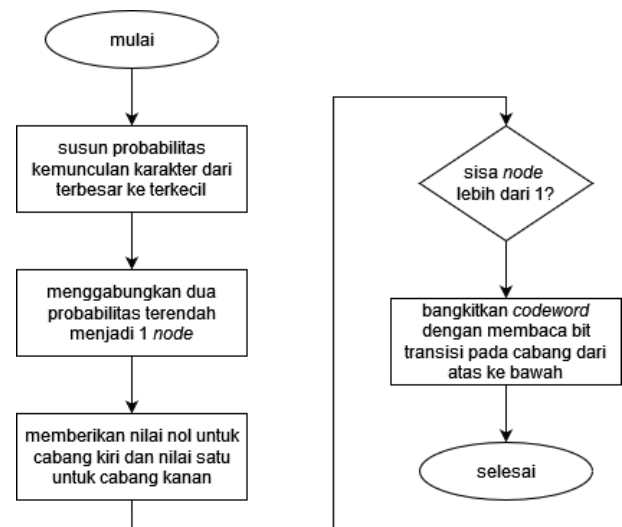
Tabel 2. Tabel File dari *The Canterbury Corpus*

File	Kategori	Ukuran (Byte)
alice29.txt	English text	152089
asyoulik.txt	Shakespeare	125179
cp.html	HTML source	24603
fields.c	C source	11150
grammar.lsp	LISP source	3721
lcet10.txt	Technical writing	426754
plrabn12.txt	Poetry	481861
xargs.l	GNU manual page	4227

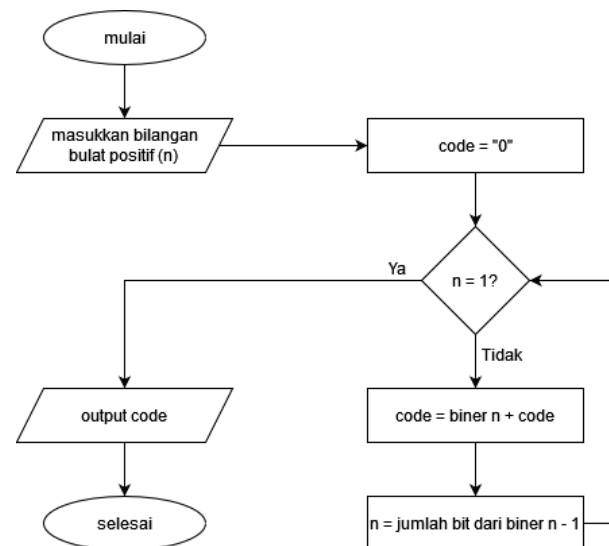
3.2. Analisis Data

Dalam tahap ini, sistem melakukan proses penghitungan dengan menerapkan metode *huffman coding* dan metode *Elias Omega Coding* pada file corpus yang dimasukkan dan menghasilkan *Ratio of Compression* (RC), *Compression Ratio* (CR), *Space Saving* (SS),

dan *Process Time* dalam satuan milisekon (waktu kompresi dan dekompresi).



Gambar 1. Flowchart Metode Huffman Coding



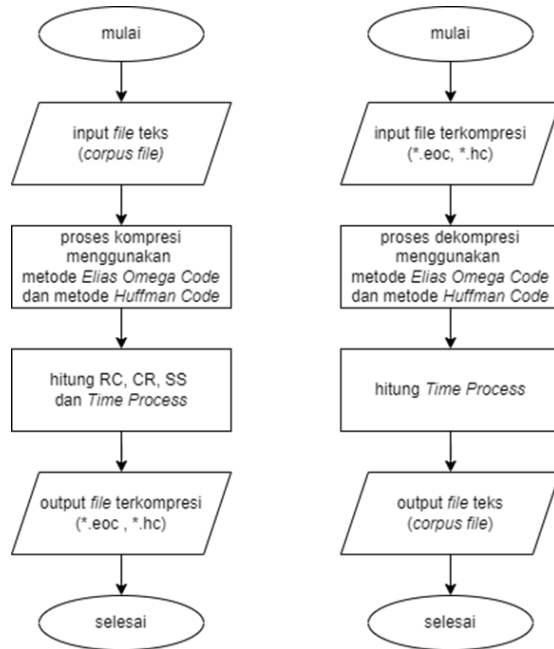
Gambar 2. Flowchart Metode Elias Omega Coding

3.3. Perancangan Sistem

Pada penelitian ini dirancang sebuah aplikasi berbasis desktop untuk memudahkan proses pengguna. Selain itu, bahasa *C Sharp* (C#) diterapkan dalam penelitian ini untuk membuat tampilan *user interface* agar mudah dimengerti pengguna. Tahap perancangan sistem merupakan proses untuk merancang dan menyusun berbagai komponen perangkat lunak dan perangkat keras secara terintegrasi, dengan

tujuan untuk menghasilkan sebuah sistem yang lebih efektif dan efisien [15].

Saat membuat suatu aplikasi, pengembang akan membuat diagram yang menggambarkan cara kerja aplikasi tersebut. Penulis menggunakan *flowchart* dengan menggunakan sistem aplikasi yang akan dibuat.



Gambar 3. Flowchart Sistem

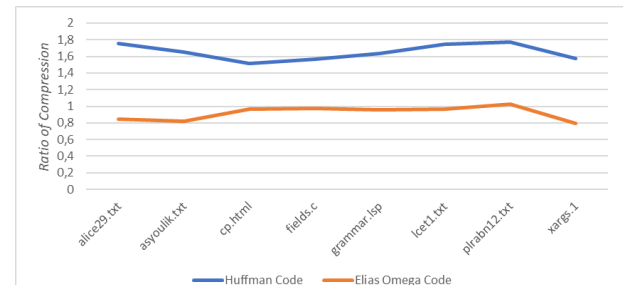
4. Hasil Penelitian

Sistem pada penelitian ini diuji dengan spesifikasi AMD Ryzen 7 5700U with Radeon Graphics 1.80 GHz, RAM 8 GB, tipe sistem 64-bit operating system dan x64-based processor, sistem operasi Windows 11 Home Single Language 23H2.

Hasil Penelitian sistem merupakan hasil kompresi dan dekompresi yang dilakukan terhadap *file corpus*. Simulasi dilakukan terhadap 8 *file corpus* berupa *The Canterbury Corpus*, yaitu *alice29.txt*, *asyoulik.txt*, *cp.html*, *fields.c*, *grammar.lsp*, *lcet1.txt*, *plrabn12.txt*, dan *xargs.l*. Simulasi kompresi dan dekompresi terhadap *file corpus* menggunakan algoritma *huffman code* dan algoritma *Elias Omega Code*.

Tabel 3. Perbandingan terhadap *Ratio of Compression*

Nama File	Ukuran Sebelum Kompresi (Byte)	Ukuran Sesudah Kompresi (Byte)		<i>Ratio of Compression</i>	
		HC	EOC	HC	EOC
alice29.txt	148.481	84.571	174.716	1,756	0,850
asyoulik.txt	125.179	75.808	153.484	1,651	0,816
cp.html	24.603	16.259	25.582	1,513	0,962
fields.c	11.150	7.119	11.481	1,566	0,971
grammar.lsp	3.721	2.276	3.879	1,635	0,959
lcet1.txt	426.754	243.881	442.663	1,750	0,964
plrabn12.txt	471.162	266.201	460.839	1,770	1,022
xargs.l	4.227	2.678	5.335	1,578	0,792
Rata-rata				1,625	0,917

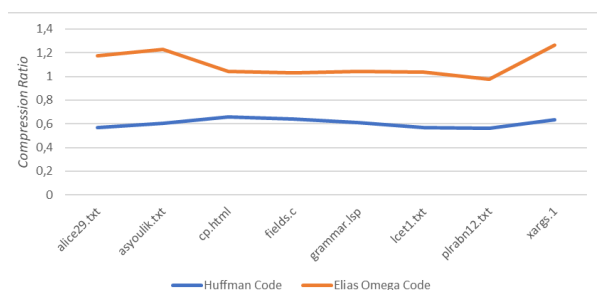


Gambar 4. Grafik Perbandingan terhadap *Ratio of Compression*

Berdasarkan pada grafik *Ratio of Compression* pada Gambar 4. dapat diketahui bahwa *Ratio of Compression* pada algoritma *huffman code* lebih besar dibandingkan algoritma *Elias Omega Code*. Hal ini dapat menunjukkan bahwa algoritma *huffman code* melakukan kompresi file *corpus* lebih baik dibandingkan algoritma *Elias Omega Code*.

Tabel 4. Perbandingan terhadap *Compression Ratio*

Nama File	Ukuran Sebelum Kompresi (Byte)	Ukuran Sesudah Kompresi (Byte)		<i>Compression Ratio</i>	
		HC	EOC	HC	EOC
alice29.txt	148.481	84.571	174.716	0,570	0,850
asyoulik.txt	125.179	75.808	153.484	0,606	0,816
cp.html	24.603	16.259	25.582	0,661	0,962
fields.c	11.150	7.119	11.481	0,638	0,971
grammar.lsp	3.721	2.276	3.879	0,612	0,959
lcet1.txt	426.754	243.881	442.663	0,571	0,964
plrabn12.txt	471.162	266.201	460.839	0,565	1,022
xargs.l	4.227	2.678	5.335	0,634	0,792
Rata-rata				0,607	1,099

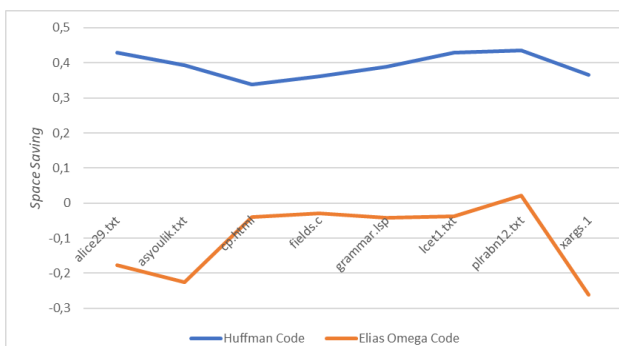


Gambar 5. Grafik Perbandingan terhadap *Compression Ratio*

Berdasarkan grafik *Compression Ratio* pada Gambar 5, diketahui bahwa *Compression Ratio* pada algoritma *huffman code* lebih kecil dibandingkan algoritma *Elias Omega Code*. Hal ini dapat menunjukkan bahwa algoritma *huffman code* melakukan kompresi file *corpus* lebih baik dibandingkan algoritma *Elias Omega Code*.

Tabel 5. Tabel Perbandingan terhadap *Space Saving*

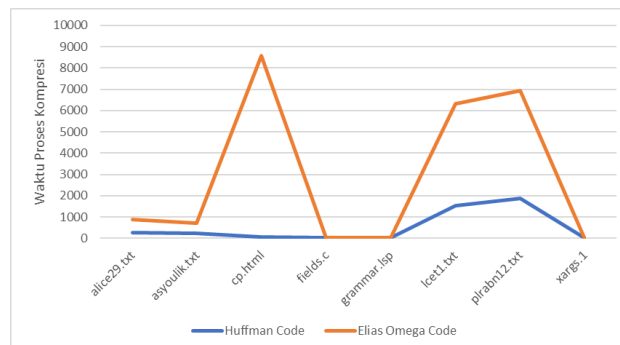
Nama File	Ukuran Sebelum Kompresi (Byte)	Ukuran Sesudah Kompresi (Byte)		Space Saving (%)	
		HC	EOC	HC	EOC
alice29.txt	148.481	84.571	174.716	43	-17,7
asyoulik.txt	125.179	75.808	153.484	39,4	-22,6
cp.html	24.603	16.259	25.582	33,9	-4
fields.c	11.150	7.119	11.481	36,2	-3
grammar.lsp	3.721	2.276	3.879	38,8	-4,2
lcet1.txt	426.754	243.881	442.663	42,9	-3,7
plrabn12.txt	471.162	266.201	460.839	43,5	2,2
xargs.l	4.227	2.678	5.335	36,6	-26,2
Rata-rata				39,3	-9,9

Gambar 6. Grafik Perbandingan terhadap *Space Saving*

Berdasarkan pada grafik *Space Saving* pada Gambar 6, diketahui bahwa *Space Saving* pada algoritma *huffman code* lebih besar dibandingkan algoritma *Elias Omega Code*. Hal ini dapat menunjukkan bahwa dalam melakukan kompresi file *corpus*, algoritma *huffman code* mampu menyusutkan ukuran data dan menghemat ruang penyimpanan yang dibutuhkan lebih baik dibandingkan algoritma *Elias Omega Code*.

Tabel 6. Perbandingan terhadap Waktu Kompresi

Nama File	Ukuran Sebelum Kompresi (Byte)	Ukuran Sesudah Kompresi (Byte)		Waktu Kompresi (ms)	
		HC	EOC	HC	EOC
alice29.txt	148.481	84.571	174.716	274,355	871,205
asyoulik.txt	125.179	75.808	153.484	233,281	708,577
cp.html	24.603	16.259	25.582	46,584	8.582,352
fields.c	11.150	7.119	11.481	21,504	33,518
grammar.lsp	3.721	2.276	3.879	11,434	13,984
lcet1.txt	426.754	243.881	442.663	1.530,444	6.330,583
plrabn12.txt	471.162	266.201	460.839	1.867,947	6.944,127
xargs.l	4.227	2.678	5.335	10,826	16,503
Rata-rata				499,547	9.384,501

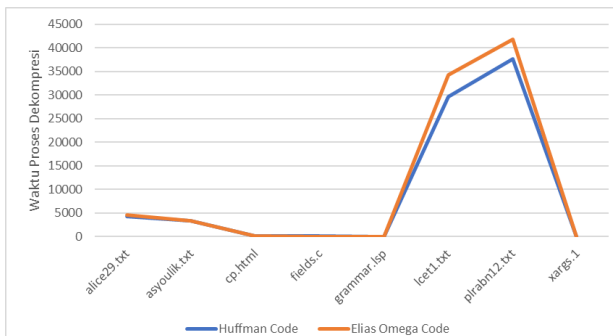


Gambar 7. Grafik Perbandingan terhadap Waktu Proses Kompresi

Berdasarkan pada grafik Waktu Kompresi pada Gambar 7, diketahui bahwa nilai rata-rata waktu kompresi algoritma *huffman code* lebih singkat dibandingkan algoritma *Elias Omega Code*. Hal ini dapat diartikan bahwa algoritma *huffman code* lebih cepat dibandingkan algoritma *Elias Omega Code* dalam proses melakukan kompresi file *corpus*.

Tabel 4.5 Tabel Perbandingan terhadap Waktu Dekompresi

Nama File	Ukuran Sebelum Kompresi (Byte)	Ukuran Sesudah Kompresi (Byte)		Waktu Dekompresi (ms)	
		HC	EOC	HC	EOC
alice29.txt	148.481	84.571	174.716	4.229,455	4.635,257
asyoulik.txt	125.179	75.808	153.484	3.398,515	3.385,857
cp.html	24.603	16.259	25.582	105,338	76,847
fields.c	11.150	7.119	11.481	36,268	20,894
grammar.lsp	3.721	2.276	3.879	19,58	8,182
lcet1.txt	426.754	243.881	442.663	29.635,385	34.274,633
plrabn12.txt	471.162	266.201	460.839	37.629,292	41.900,966
xargs.l	4.227	2.678	5.335	22,174	10,518
Rata-rata				2.937,606	10.539,144



Gambar 8. Grafik Perbandingan terhadap Waktu Proses Dekompresi

Berdasarkan pada grafik Waktu Dekompresi pada Gambar 8, diketahui bahwa nilai rata-rata waktu dekomposisi algoritma *huffman code* lebih singkat dibandingkan algoritma *Elias Omega Code*. Hal ini dapat diartikan bahwa algoritma *huffman code* lebih cepat dibandingkan algoritma *Elias Omega Code* dalam proses melakukan dekomposisi file corpus.

Keunggulan *Elias Omega* lebih terasa dalam konteks pengkodean bilangan bulat besar dengan kompresi ruang yang lebih efisien untuk bilangan numerik atau panjang data tertentu. Keunggulan *huffman code* terletak pada kompresi simbol berdasarkan frekuensi dan distribusi simbol dalam dataset, membuatnya sangat efektif untuk teks atau data berbasis simbol dengan frekuensi yang tidak merata.

5. Kesimpulan

Penelitian ini berhasil membangun aplikasi yang dapat melakukan kompresi dan dekomposisi *file* teks menggunakan algoritma *huffman code* dan algoritma *Elias Omega Code*. Perbandingan kinerja kedua algoritma dalam hasil simulasi terhadap *file corpus* menunjukkan bahwa algoritma *huffman code* lebih baik dalam melakukan kompresi dan dekomposisi terhadap *file corpus* dengan rata-rata *ratio of compression* sebesar 1,625, *compression ratio* sebesar 0,607, *space saving* sebesar 39,3%, waktu kompresi sebesar 499,547 ms, dan waktu dekomposisi sebesar 9.384,501 ms. Sedangkan, algoritma *Elias Omega Code* memiliki rata-rata *ratio of compression* sebesar 0,917, *compression ratio*

sebesar 1,099, *space saving* sebesar -9,9%, waktu kompresi sebesar 2.937,606 ms, dan waktu dekomposisi sebesar 10.539,144 ms.

6. Saran

Penelitian selanjutnya dapat dikembangkan dan diimplementasikan algoritma yang digunakan pada penelitian ini tidak terbatas pada kompresi *file* teks tetapi juga dapat melakukan kompresi jenis *file* lain, seperti citra, suara, dan video, serta pengembangan pada sistem dapat dibangun dengan berbasis *web* dan *mobile*

7. Daftar Pustaka

- [1] K. Mahesa and K. Karpen, "Rancang Bangun Aplikasi Kompresi dan Dekompresi pada Citra Digital Menggunakan Metode Huffman," *PROCESSOR*, vol. 12, no. 1, pp. 948–963, Apr. 2017.
- [2] R. Sihotang, "Perancangan Aplikasi Kompresi File Teks Dokumen Menggunakan Kombinasi Algoritma Bijective Burrows-Wheeler Transform (BBWT) dan Algoritma Stout Code," *Inf. dan Teknol. Ilm.*, vol. 9, no. 3, pp. 60–63, 2022.
- [3] K. Levkoff, "What is the reason a zip file is larger than the original - Quora," *Quora*, 2023.
- [4] V. Lusiana, "Teknik Kompresi Citra Digital untuk Penyimpanan File menggunakan Format Data XML," *J. Teknol. Inf. Din.*, vol. 19, no. 2, pp. 112–119, Jun. 2014, [Online]. Available: <http://www.w3.org/>
- [5] A. P. U. Siahaan, "Implementasi Teknik Kompresi Teks Huffman," *J. Inform.*, vol. 10, no. 2, pp. 1251–1261, Jul. 2016.
- [6] S. A. B. Siburian, R. K. Hondro, and B. Purba, "Implementasi Algoritma Elias Omega Coding Untuk Kompresi File Teks Pada Aplikasi Media Pembelajaran Buku Sekolah Digital Untuk Pelajar," vol. 4, no. 1, 2020, doi: 10.30865/komik.v4i1.2677.
- [7] A. Ani, "Data Compression: Do we need it," *Medium*, 2021.
- [8] N. Seregar, "Penerapan Algoritma Lzy Untuk Mengkompresi Record Database MySQL," *J. Ilmu Komputer, Teknol. Dan Inf.*, vol. 1, no. 2, pp. 43–50, Jul. 2023, [Online]. Available: <https://journal.grahamitra.id/index.php/jurikti>
- [9] D. Rachmawati, M. A. Budiman, and M. A. Subada, "Comparison study of Fibonacci code

- algorithm and Even-Rodeh algorithm for data compression,” *J. Phys. Conf. Ser.*, vol. 1321, no. 3, pp. 1–8, Nov. 2019, doi: 10.1088/1742-6596/1321/3/032015.
- [10] N. Dale and J. Lewis, *Computer Science Illuminated*, 7th ed. Burlington: Jones & Bartlett Learning, 2020.
- [11] M. Nangir, “An Efficient Secure Universal Block Source Coding Algorithm for Integers,” *Indian J. Fundam. Appl. Life Sci.*, vol. 5, no. S3, pp. 1430–1439, 2015, [Online]. Available: www.cibtech.org/sp.ed/jls/2015/03/jls.htm
- [12] K. Sayood, *Introduction to Data Compression, Third Edition*, 3rd ed. Elsevier Inc., 2006.
- [13] M. Apriyanto and H. Hutrianto, “Analisa Penerapan Algoritma GoldBach Codes dan Metode Shannon-Fano pada Kompresi File Teks,” *Bina Darma Conf. Comput. Sci.*, vol. 2, no. 5, pp. 207–218, 2020.
- [14] M. A. Budiman and D. Rachmawati, “On using Goldbach G0 codes and Even-Rodeh codes for text compression on using Goldbach G0 codes and Even-Rodeh codes for text compression,” in *IOP Conference Series: Materials Science and Engineering*, Mar. 2017, vol. 180, no. 1. doi: 10.1088/1757-899X/180/1/012062.
- [15] N. Azis, *Analisis Perancangan Sistem Informasi*. Bandung: Penerbit Widina Bhakti Persada Bandung, 2022.