

ANALISIS PERBANDINGAN ALGORITMA LZW DENGAN LEVENSTHEIN PADA KOMPRESI FILE PPT

Monalisa Dosma Sihombing¹, Hery Sunandar², Kurnia Ulfa³

^{1,2,3}Universitas Budi Darma, Medan

¹monalisadosmasihombing19@gmail.com, ²herysun1975@gmail.com,

³kurniaulfa82@gmail.com

ABSTRAK

Di era digital sekarang file ppt atau pptx sangat dibutuhkan saat ini sangat penting dalam dunia kerja untuk presentasi atau mengolah data. Dengan menggunakan pptx saat ini dalam presentasi, kegiatan pemaparan bisa lebih tertata dan terstruktur sesuai dengan apa yang dirancang. Namun terkadang, size file pptx cukup besar dibandingkan dengan data lain yang menyebabkan pemborosan penyimpanan. Oleh karena itu diperlukan pengkompresian. Ketika melakukan kompresi file pptx, penting untuk mempertimbangkan keseimbangan antara ukuran file dan kualitas konten. Semakin tinggi tingkat kompresi, semakin kecil ukuran file, tetapi ada kemungkinan kualitas file pptx menjadi terpengaruh. Kompresi data adalah teknik yang digunakan untuk mengurangi ukuran file atau informasi tanpa mengorbankan kualitas atau signifikansi data. Dalam era di mana data semakin melimpah, kompresi data menjadi sangat penting untuk menghemat ruang penyimpanan dan mempercepat transfer informasi. Dengan menghilangkan redundansi dan memanfaatkan pola yang ada dalam data, kompresi data dapat menciptakan representasi yang lebih efisien dan ringkas. Kendala yang terjadi karena ukuran data *file pptx* cenderung lebih besar dari data lainnya akibat banyaknya penggunaan slide yang memboroskan memori penyimpanan dan memperlambat transfer data. Berdasarkan hal tersebut untuk mengatasi masalah tersebut, diperlukan suatu teknik pengecilan data atau sering disebut sebagai kompresi data. Kompresi bertujuan untuk memperkecil ukuran data menjadi ukuran yang lebih kecil, dan parameter kompresi yang didapatkan adalah berupa *ratio of compression*, *compression ratio* dan *space saving*.

Kata Kunci: kompresi file PPT, *algoritma* lzw, ratio kompresi

ABSTRACT

In today's digital era, ppt or pptx files are crucial for presentations or data processing in the professional world. Using pptx for presentations allows for a more organized and structured presentation according to the design. However, the size of pptx files can be quite large compared to other data, leading to storage wastage. Thus, compression is needed. When compressing pptx files, it is important to balance file size and content quality. Higher compression levels reduce file size but may affect file quality. Data compression is a technique used to reduce file or information size without compromising data quality or significance. In an age of abundant data, data compression is essential for saving storage space and speeding up information transfer. By eliminating redundancy and utilizing existing data patterns, data compression can create a more efficient and compact representation. The challenge arises as pptx file sizes tend to be larger than other data due to the extensive use of slides, which wastes storage and slows down data transfer. To address this issue, a data reduction technique or compression is needed. Compression aims to reduce data size to a smaller size, and the compression parameters obtained include the ratio of compression, compression ratio, and space saving.

Keywords: PPT file compression, lzw algorithm, compression ratio

A. Pendahuluan

Di era digital saat ini, file PPT atau PPTX sangat penting dalam dunia kerja, terutama untuk presentasi dan pengolahan data. Penggunaan PPTX memungkinkan presentasi menjadi lebih tertata dan terstruktur, sesuai dengan desain yang diinginkan (Akbar, 2017). Namun, masalah yang sering muncul adalah ukuran file PPTX yang cukup besar, yang dapat mengakibatkan pemborosan ruang penyimpanan. Oleh karena itu, kompresi file menjadi solusi penting untuk mengurangi ukuran file tanpa mengorbankan kualitas konten (Putra et al., 2022). Penting untuk mempertimbangkan keseimbangan antara ukuran file dan kualitas, karena kompresi yang terlalu tinggi bisa memengaruhi kualitas file.

Kompresi data adalah teknik esensial yang digunakan untuk mengurangi ukuran file atau informasi tanpa mengorbankan kualitas data (Alqori, 2024);(Sipapaga, 2024). Di era informasi yang melimpah ini, kompresi data menjadi sangat penting untuk menghemat ruang penyimpanan dan mempercepat transfer informasi. Dengan menghilangkan redundansi dan

memanfaatkan pola yang ada dalam data, kompresi dapat menciptakan representasi yang lebih efisien dan ringkas, mempermudah pengelolaan data dan penggunaan ruang penyimpanan.

File PowerPoint, atau file dengan format PPTX, adalah format standar untuk presentasi yang digunakan sejak PowerPoint 2007 dan versi yang lebih baru (Mentari, 2020);(SURIANSHA, 2018). File PPTX dapat mencakup berbagai elemen seperti teks yang diformat, objek, banyak slide, gambar, dan video (Sandy, 2017). Oleh karena itu, teknik kompresi yang efektif sangat penting untuk mengelola dan mengoptimalkan file-file besar ini, memastikan bahwa mereka tetap mudah dikelola dan dikirim tanpa mengorbankan kualitas visual dan informatif dari presentasi tersebut. Kendala yang terjadi karena ukuran data *file pptx* cenderung lebih besar dari data lainya akibat banyaknya penggunaan slide yang memboroskan memori penyimpanan dan memperlambat transfer data. Berdasarkan hal tersebut untuk mengatasi masalah tersebut, diperlukan suatu teknik pengecilan

data atau sering disebut sebagai kompresi data. Kompresi bertujuan untuk memperkecil ukuran data menjadi ukuran yang lebih kecil, dan parameter kompresi yang didapatkan adalah berupa *ratio of compression*, *compression ratio* dan *space savin* (Putra et al., 2022).

Penulis meneliti *algoritma* yang cocok dan efisien untuk aplikasi kompresi dengan tujuan menggabungkan dua *algoritma*, yaitu LZW dan *Levenshtein*, untuk mencapai hasil kompresi maksimal. *Algoritma* LZW efektif untuk mengompresi teks dan data sederhana dengan cara dinamis membangun kamus kompresi selama proses, sementara *Levenshtein* unggul dalam membandingkan dan menentukan kesamaan atau perbedaan antara dua string, serta dalam koreksi ejaan dan pengenalan pola (Pirnando & Febryansyah, 2022). Dalam penelitian ini, proses kompresi dilakukan dua kali: pertama menggunakan *algoritma* LZW, kemudian hasilnya dikompresi kembali dengan *algoritma Levenshtein*. Metode ini bertujuan untuk mengoptimalkan ukuran file PPTX dengan memanfaatkan keunggulan masing-masing *algoritma*

, sehingga aplikasi kompresi yang dihasilkan dapat membantu banyak orang dalam mencapai keseimbangan antara kapasitas penyimpanan dan kualitas file (Silaban, 2022)

Menurut penelitian Lilik Prabowo pada tahun 2019 tentang analisis perbandingan *algoritma* LZW dan *Levenshtein* pada kompresi file PPT, kompresi data dilakukan untuk menghemat ruang penyimpanan dan mempercepat transfer file melalui jaringan. Penelitian ini fokus pada dua teknik kompresi, yaitu *algoritma* LZW dan *Levenshtein*, yang masing-masing memiliki keunggulan dalam mengompresi data. File dengan ekstensi "pptx" adalah format standar untuk presentasi Microsoft PowerPoint dari versi 2007 dan seterusnya (Adi, 2017). Menurut penelitian Vladimir *Levenshtein* pada tahun 1968, *algoritma Levenshtein*, juga dikenal sebagai *Levenshtein coding*, adalah metode pengkodean universal untuk bilangan bulat non-negatif (Yuliasuti, 2023). Meskipun *algoritma* ini kurang dikenal atau tidak banyak dibahas, ia melibatkan proses tahapan tertentu baik dalam pengkodean maupun pembacaan

sandi, menjadikannya teknik yang sistematis dalam penanganan data.

Menurut penelitian yang dilakukan oleh Abraham Lempel, Jacob Ziv, dan Terry Welch, *algoritma* LZW berkembang dari karya awal Lempel dan Ziv pada tahun 1977 mengenai kompresi data adaptif, dengan kontribusi perbaikan oleh Welch yang diterbitkan pada tahun 1984. *Algoritma* LZW sendiri adalah metode untuk mengurangi ukuran data teks dengan menggantikan serangkaian karakter yang sering muncul dengan kode yang lebih pendek. Berdasarkan latar belakang tersebut diatas, maka penulis akan mengangkat judul “Analisis perbandingan *Algoritma* LZW dengan Levenshtein pada kompresi file PPT”.

B. Metode Penelitian

Penelitian analisis perbandingan *algoritma* LZW dan *Levenshtein* pada kompresi file PPT memiliki kerangka kerja yang terstruktur dengan jelas, dimulai dari identifikasi masalah untuk menetapkan fokus utama penelitian, dilanjutkan dengan kajian pustaka untuk mengumpulkan data teoritis dari berbagai sumber terpercaya. Tahapan berikutnya

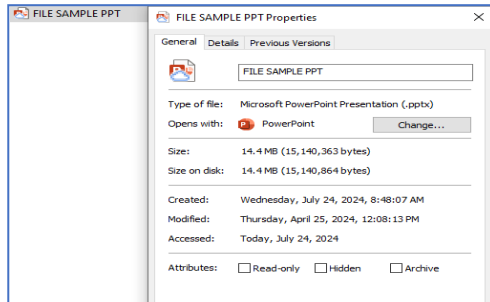
melibatkan analisa penggunaan data dengan sampel file PPT, sebelum melanjutkan ke penerapan metode untuk mengimplementasikan *algoritma* LZW dan *Levenshtein* dalam proses kompresi. Setelah itu, dilakukan analisis perbandingan hasil kompresi dari kedua *algoritma*, diikuti dengan implementasi hasil analisis ke dalam aplikasi. Pengujian dilakukan untuk memastikan kesesuaian hasil penerapan dengan standar yang ditetapkan. Akhirnya, proses dokumentasi/laporan menyajikan kesimpulan hasil penelitian secara menyeluruh serta memberikan saran untuk penelitian lebih lanjut.

Sampel Data

Pengguna sampel data pada penelitian ini menggunakan nilai *hexadecimal* pada objek data antara lain berkas teks dengan ekstensi *PPT. Selanjutnya objek data tersebut di ekstrasi nilai *hexadecimalnya* dengan menggunakan aplikasi Binary View. Proses pengambilan nilai *hexadecimal* sebanyak enam belas pasang nilai *hexadecimal* dari sampel berkas teks *PPT yang telah ditentukan.

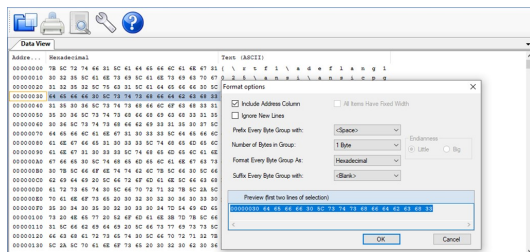
Berkas teks dengan ekstensi *.PPT sebagai sampel yang

digunakan dalam penelitian ini penulis mengambil dari dokumen yang dimiliki penulis, berikut tampilan peroperties berkas teks yang digunakan dalam penelitian ini.



Gambar 1 Properties Berkas Teks Ekstensi *.PPT

Berikut tampilan proses ekstraksi nilai dari objek data berkas teks dengan ekstensi *.PPT, menggunakan aplikasi Binary View.



Gambar 2 Ekstraksi Nilai Hexadecimal dari Berkas Teks Ekstensi *.PPT

Pada gambar 2 berikut nilai *hexadecimal* sampel yang digunakan dalam proses penerapan metode kompresi nila hexadecimalnya diambil dari alamat. 000030 dengan nilai *hexadecimal* berpasangan sebanyak 16 dengan nilai antara lain

64 65 66 66 30 5C 73 74 73 68 66 64
62 63 68 33.

C. Hasil Penelitian dan Pembahasan

Penggunaan Sampel Data

Penggunaan sampel data pada penelitian ini menggunakan berkas teks dengan ekstensi *.PPT yang deskripsi atau properties objek berkas teks telah dijelaskan di BAB III sebelumnya. Adapun nilai sampel hexadecimal yang digunakan yaitu 64 65 66 66 30 5C 73 74 73 68 66 64 62 63 68 33, jika ditabelkan terkait dengan frekuensi nila *hexadecimal* maka tampil seperti berikut ini:

Tabel 1 Frekuensi Kemunculan Nilai Hexadecimal

Nilai	Frekuensi	Nilai Binner	Bit	Frekuensi x Bit
64	2	01100100	8	16
65	1	01100101	8	8
66	3	01100110	8	24
30	1	00110000	8	8
5C	1	01011100	8	8
73	2	01110011	8	16
74	1	01110100	8	8
68	2	01101000	8	16
62	1	01100010	8	8
63	1	01100011	8	8
33	1	00110011	8	8
Total Bit	128 KB			

Seperti terlihat pada tabel di atas kapasitas dari sampel nilai *hexadecimal* yang digunakan untuk menerapkan algoritma LZW dan Levensthein dalam proses kompresi dimana dari sampel data tersebut memiliki kapasitas 128 bit dengan kapasitas byte $128/8 = 16$ byte.

Penerapan Algoritma LZW

Setelah proses membaca isi

Tabel 2 Nilai Hexadecimal Yang Belum Dikompresi

Nilai	Frekuensi	Nilai Binner	Bit	Frekuensi x Bit
66	3	01100110	8	24
64	2	01100100	8	16
73	2	01110011	8	16

Lanjutan Tabel 3 Nilai Hexadecimal Yang Belum Dikompresi

Nilai	Frekuensi	Nilai Binner	Bit	Frekuensi x Bit
68	2	01101000	8	16
65	1	01100101	8	8
30	1	00110000	8	8
5C	1	01011100	8	8
74	1	01110100	8	8
62	1	01100010	8	8
63	1	01100011	8	8
33	1	00110011	8	8
Total Bit				128

Setelah melakukan proses pengurutan bilangan heksadesimal berdasarkan frekuensi kemunculannya dan mendapatkan nilai biner yang sesuai, langkah

file, maka dihasilkan bilangan hexadecimal dari file *.PPT yang digunakan untuk menerapkan rumus *algoritma* LZW dan Levensthein. Berikut ini nilai-nilai hexadecimal diurutkan berdasarkan frekuensinya, nilai frekuensi dengan frekuensi tertinggi akan berada di urutan pertama.

berikutnya melakukan penerapan proses kompresi menggunakan *algoritma* LZW sesuai nilai hexadecimal **64 65 66 66 30 5C 73 74 73 68 66 64 62 63 68 33**. Detail

mengenai langkah-langkah kompresi dilihat pada table 4.3 dibawah ini.
 berkas teks *.PPT dengan LZW dapat

Tabel 3 Proses Kompresi LZW

Current (W)	Next (K)	W + K	Output	Dictionary
64	65	64 + 65	64	256
65	66	65 + 66	65	257
66	66	66 + 66	66	258
66	30	66 + 30	258	-
30	5C	30 + 5C	30	259
5C	73	5C + 73	5C	260
73	74	73 + 74	73	261
74	73	74 + 73	74	262
261	68	261 + 68	-	-
68	66	68 + 66	68	263
258	64	258 + 64	256	-

Lanjutan Tabel 4 Proses Kompresi LZW

Current (W)	Next (K)	W + K	Output	Dictionary
256	62	256 + 62	-	-
62	63	62 + 63	62	264
63	68	63 + 68	63	265
263	33	263 + 33	-	-
33	-	33	33	266

Setelah penerapan rumus dari proses kompresi LZW:
 LZW diterapkan maka berikut hasil

Tabel 5 Jumlah Bit Hasil Kompresi

Nilai	Frekuensi	Nilai Binner	Bit	Frekuensi x Bit
64	1	01100100	8	8
65	1	01100101	8	8
66	1	01100110	8	8
258	1	100000010	9	9
30	1	00110000	8	8
5C	1	01011100	8	8

73	1	01110011	8	8
74	1	01110100	8	8
68	1	01101000	8	8
256	1	100000000	9	9
62	1	01100010	8	8
63	1	01100011	8	8
33	1	00110011	8	8
Total Bit				106
Kapasitas bit				106/8 = 13,25 bit

File video berhasil melalui proses kompresi sukses menggunakan *algoritma* LZW, jenis berkas tersebut akan mengalami perubahan format sesuai dengan standar yang telah ditentukan. Hal ini dilakukan sebagai tanda bahwa berkas tersebut telah melewati kompresi melalui aplikasi yang menerapkan *algoritma* LZW. Format berkas adalah *.PPT.

Untuk mengevaluasi kinerja *algoritma* LZW dalam proses kompresi file teks *.PPT, penulis mengukur hasil kompresi sesuai dengan parameter yang ditentukan. Hasil kompresi file ditentukan oleh beberapa parameter: Ratio of Compression (RC) sebesar 1,208, Compression Ratio (CR) yang mencapai 80,81%, Redundancy (Rd) sebesar 17,19%, dan Space Saving (SS) yang tercatat 19,19%. Dari

perhitungan ini, dapat disimpulkan bahwa ukuran file teks *.PPT yang awalnya berkapasitas 128 bit setelah kompresi menggunakan *algoritma* LZW berkurang menjadi 106 bit, atau 0,151 byte jika dikonversi ke kapasitas byte. Dengan kapasitas awal file *.PPT sebesar 123 MB dan rasio kompresi 80,81%, hasil akhir kompresi menghasilkan ukuran file menjadi 99,40 MB..

Penerapan *Algoritma* Levensthein

Penerapan *algoritma* Levensthein yang menjadi data untuk kompresi dapat dilihat nilai pada tabel 4.2 Nilai Hexadecimal Yang Belum Dikompresi. Kemudian dilakukan proses menambahkan *Levensthein* code (nilai *Levensthein* dapat dilihat pada BAB II) pada nilai data *hexadecimal*, sehingga seperti pada tabel berikut ini:

Tabel 6 Nilai Hexadecimal yang telah dikompresi dengan Levensthein code

No	Nilai	Frekuensi	Codeword Tabo Codes	Bit	Frekuensi x Bit
1	66	3	0100	4	12
2	64	2	1000	4	8
3	73	2	1100	4	8
4	68	2	010100	6	12
5	65	1	011000	6	6
6	30	1	011100	6	6
7	5C	1	100100	6	6
8	74	1	101100	6	6
9	62	1	110100	6	6
10	63	1	111000	6	6
11	33	1	111100	6	6
Total Bit					82 Bit

Setelah melakukan kompresi dengan *algoritma Levensthein*, jumlah bit dalam tabel perhitungan di atas adalah 82 bit. Tahap berikutnya adalah mengorganisir ulang string bit yang dihasilkan dari kompresi berdasarkan dalam bilangan posisi Heksadesimal yaitu **64 65 66 66 30 5C 73 74 73 68 66 64 62 63 68 33** sehingga biner hasil kompresi *codeword Levensthein* sesuai posisi *hexadecimal* di atas sebagai berikut:

Tabel 7Penyusunan String bit hasil kompresi *algoritma Levensthein*

64	1000
65	011000
66	0100

66	0100
30	011100
5C	100100
73	1100
74	101100
73	1100
68	010100
66	0100
64	1000
62	110100
63	111000
68	010100
33	111100

Setelah setiap karakter diatur ulang menggunakan kode kebenaran *Levensthein*, langkah selanjutnya adalah menyusun kembali hasil

kompresi setiap karakter, sehingga diperoleh *string bit* seperti berikut ini:
 1000 011000 0100 0100 011100
 100100 1100 101100 1100 010100
 0100 1000 110100 111000 010100
 111100.

Dikarenakan jumlah bit hasil kompresi *codeword Levensthein* 82 bit, sesuai dengan ketentuan *Levensthein* jika panjang bit data setelah kompresi tidak dapat dibagi habis oleh 8 atau bukan kelipatan 8, maka digunakan variabel yang disebut *padding* dan *flagging* untuk menambahkan bit data. *Padding* adalah tambahan bit data pada hasil kompresi sehingga total bit data menjadi kelipatan 8. Rumus *padding* adalah $7-n+ "1"$. Sementara itu, *flagging* adalah penambahan 8 bit bilangan biner setelah *padding*, yang digunakan untuk mempermudah pembacaan bit-bit hasil kompresi selama proses dekompresi. Rumus *Flagging* adalah $9.-n$.

Jika panjang bit dapat dibagi habis oleh 8 atau merupakan kelipatan dari 8, maka *padding* tidak diperlukan, namun masih memerlukan tambahan *flagging*.

Tabel 8 Penambahan Padding dan Flag bit dengan Algoritma Levensthein

Padding	Flagging bit
---------	--------------

$n = 82 \bmod 8 = 2$ $7 - n + "1"$ $7 - 2 + "1" = 110$	$9 - n$ $9 - 2 = 7 = 111$
--	------------------------------

Sehingga biner dapat disusun setelah penambahan *padding* dan *flagging* bit sebagai berikut 10000110
 00010001 00011100 10010011
 00101100 11000101 00010010
 00110100 11100001 01001111
 00110111.

Langkah berikutnya mengubah nilai biner ini menjadi nilai heksadesimal untuk menghitung nilai kapasitas yang dihasil setelah di kompresi menggunakan teknik *Levensthein*, berikut nilai hasil kompresi yang dihasilkan:

Tabel 9 Hasil Kompresi Algoritma Levensthein

No	Biner	Heksadesimal
1	10000110	86
2	00010001	11
3	00011100	1C
4	10010011	93
5	00101100	2C
6	11000101	C5
7	00010010	12
8	00110100	34
9	11100001	E1
10	01001111	4F
11	00110111	37
Total Bit	88 Bit	

Setelah penerapan *algoritma Levenshtein* pada kompresi file teks dengan ekstensi *.PPT, berhasil menurunkan kapasitas bit dari 128 bit menjadi 88 bit. Parameter kinerja *algoritma Levenshtein* mencakup Ratio of Compression (RC) sebesar 1,455, Compression Ratio (CR) sebesar 68,75%, Redundancy (Rd) sebesar 31,25%, dan Space Saving (SS) sebesar 31,25%. Berdasarkan perhitungan ini, ukuran file teks *.PPT yang awalnya 123 MB setelah kompresi dengan rasio 68,75% menjadi 84,56 MB, menunjukkan efektivitas *algoritma Levenshtein* dalam mengurangi ukuran file..

Perbandingan Algoritma Kompresi

Metode *exponential* merupakan salah satu metode yang digunakan untuk membandingkan kinerja kedua *algoritma* LZW dan *Levensthein*. Adapun langkah-langkah proses perhitungannya yaitu:

Menentukan alternatif keputusan yang akan dipilih.

Dalam proses perbandingan kinerja antara *algoritma* LZW dan *algoritma Levensthein* pada saat melakukan kompresi, perlu menentukan *algoritma* mana yang harus digunakan.

Tabel 10 Penentuan Alternatif MPE

No	Alternatif
1.	<i>Algoritma LZW</i>
2.	<i>Algoritma Levensthein</i>

Menentukan kriteria

Langkah selanjutnya adalah melakukan penentuan kriteria dalam menganalisa proses dan cara kerjanya, untuk mendapatkan hasil perbandingan dari kedua *algoritma* tersebut. Berikut kriteria yang digunakan:

Tabel 11 Menentukan Kriteria

No	Kriteria	Keterangan
1.	<i>Ratio of Compression (RC)</i>	Mebandingkan hasil data sebelum dikompresi dengan data yang sudah dikompresi
2.	<i>Compression Ratio (CR)</i>	Perbandingan antara data yang sudah dikompresi dengan data yang belum dikompresi
3	<i>Redudancy</i>	Persentase perubahan kualitas data digital yang dikompresi
4.	<i>Space Saving (SS)</i>	Presentase perbedaan antara ukuran data sebelum dikompresi dan setelah dikompresi

Menentukan bobot kriteria

Menentukan bobot kriteria adalah suatu komponen yang paling penting terhadap nilai analisa perbandingan. Bobot dari masing-masing kriteria akan ditentukan oleh *decision maker* (pengambil keputusan) untuk menentukan tingkat kepentingan masing-masing kriteria. Tingkat kepentingan setiap kriteria dinilai berdasarkan skala 1 sampai 4, dimana nilai 1 = Kurang Penting, 2 = Cukup Penting, 3 = Penting dan 4 = Sangat Penting. Tabel berikut ini merupakan tabel tingkat kepentingan yang telah ditentukan oleh *decision maker* (pengambil keputusan).

Tabel 12 Pembobotan Kriteria

No	Kriteria	Tingkat kepentingan	Nilai bobot
1.	<i>Ratio Of Compression (Rc)</i>	Kurang Penting	1
2.	<i>Compression Ratio (Cr)</i>	Cukup Penting	2
3.	<i>Redudancy</i>	Penting	3
3.	<i>Space Saving (Ss)</i>	Sangat Penting	4

Melakukan Penilaian terhadap semua alternatif pada setiap kriteria.

Pemberian nilai pada setiap kriteria yang telah ditentukan, nilai didapat berdasarkan analisa LZW dan *algoritma Levensthein*, dapat dilihat pada tabel berikut ini:

Tabel 13 Pemberian Nilai pada setiap kriteria

No	Kriteria	Alternatif		Bobot
		LZW	Levensthein	
1.	<i>Ratio Of Compression (Rc)</i>	1,208	1,455	1
2.	<i>Compression Ratio (Cr)</i>	80,81%	68,75%	2
3.	<i>Redudancy</i>	17,19%	31,25%	3
4.	<i>Space Saving (Ss)</i>	19,19%	31,25%	4

Menghitung skor atau nilai total setiap alternatif.

Setelah kriteria diberikan nilai, langkah selanjutnya adalah perhitungan menggunakan rumus metode eksponensial, yaitu Total

Nilai = $\sum(N)B$, di mana N adalah nilai kriteria dan B adalah bobot. Untuk *algoritma LZW*, perhitungan total nilai dilakukan dengan menjumlahkan hasil perkalian nilai kriteria dengan bobot eksponensial masing-masing:

$(1,208)^1 + (80,81)^2 + (17,19)^3 + (19,19)^4$, yang menghasilkan total nilai sebesar 147,223.597. Sebaliknya, untuk *algoritma Levenshtein*, perhitungan total nilai dilakukan dengan rumus yang sama: $(1,455)^1 + (68,75)^2 + (31,25)^3 + (31,25)^4$, menghasilkan total nilai sebesar 988,919.912. Perhitungan ini menunjukkan bahwa *algoritma Levenshtein* memiliki total nilai yang jauh lebih tinggi dibandingkan *algoritma LZW*.

Menentukan hasil keputusan

Langkah selanjutnya yaitu menentukan prioritas keputusan berdasarkan nilai dari masing-masing alternatif. Hasil penentuan prioritas keputusan dapat dilihat pada tabel di bawah ini :

Tabel 14 Prioritas Keputusan

No	Alternatif	Total Nilai	Rangkin g
1.	<i>Algoritma Levensthein</i>	988919,912	1
2.	<i>Algoritma LZW</i>	147223,597	2

Berdasarkan hasil analisa perbandingan kinerja *algoritma LZW* dan *algoritma Levensthein* menggunakan metode *exponential* di atas, maka dapat disimpulkan bahwa *algoritma Levensthein* menjadi *algoritma* yang lebih efektif dan optimal dalam mengkompresi *file* teks

dengan ekstensi *.PPT dibandingkan dengan *algoritma LZW*.

Perancangan dan Pemodelan Sistem

Perancangan sistem adalah proses perancangan untuk merancang suatu sistem baru atau memperbaiki suatu sistem yang telah ada sehingga sistem tersebut menjadi lebih baik dan biasanya proses ini terdiri dari proses merancang input, output dan file. Sedangkan Pemodelan sistem adalah proses membangun atau membentuk suatu model dari suatu sistem nyata dalam bahasa formal tertentu. Tujuan dari perancangan dan pemodelan sistem secara umum adalah untuk memberikan gambaran umum kepada user tentang sistem yang baru. Desain sistem secara umum mengidentifikasi komponen-komponen sistem informasi yang akan didesain secara rinci.

Implementasi

Implementasi program merupakan tahapan uji coba dari suatu sistem yang telah dirancang. Bagian ini membahas tentang spesifikasi perangkat keras (*hardware*), perangkat lunak (*Software*) dan hasil dari tampilan

sistem sesuai dengan yang dirancang ketika sedang dijalankan.

1. *Hardware*

Untuk merancang dan menjalankan sistem dengan baik, spesifikasi perangkat keras yang diperlukan adalah sebagai berikut: Processor AMD Athlon Silver 3050U dengan Radeon Graphics 2.30 GHz, RAM minimal 2 GB, Harddisk minimal 250 GB, Monitor dengan resolusi 1366 x 768 piksel, serta Keyboard dan Mouse. Spesifikasi ini memastikan bahwa sistem dapat berjalan secara efisien dan mendukung kinerja yang optimal.

2. *Software*

Perangkat lunak (*Software*) yang digunakan dalam merancang sistem dan menjalankannya agar berjalan dengan baik, maka diperlukan spesifikasi perangkat keras (*Software*) yaitu sebagai berikut :

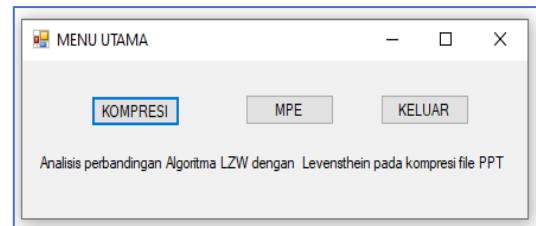
- a. Sistem Operasi : Minimal Windows 7
- b. Aplikasi : *Microsoft Visual Studio 2010*

3. Tampilan Implementasi Sistem

Tampilan sistem adalah tampilan visual sebuah produk yang menghubungkan sistem dengan

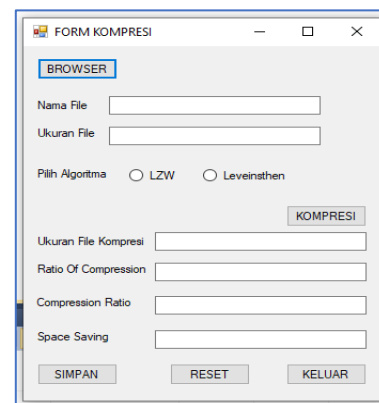
pengguna (*user*). Berikut merupakan tampilan sistem yang telah dirancang yaitu sebagai berikut:

a. Tampilan Implementasi Form Utama



Gambar 3 Implementasi
Form Utama

b. Tampilan Implementasi Form Kompresi



Gambar 4 Implementasi
Form Kompresi

c. Tampilan Implementasi Form MPE (Metode Perbandingan Eksponensial)

Gambar 5 Implementasi Form MPE
Pengujian

Tampilan sistem adalah tampilan visual sebuah produk yang menghubungkan sistem dengan pengguna (*user*). Berikut merupakan tampilan sistem yang telah dirancang yaitu sebagai berikut:

1. Tampilan Pengujian Form Utama

Gambar 6 Implementasi Form Utama

Gambar di atas tampak proses pengujian terhadap form utama aplikasi. Pada tampilan form utama ada tiga tombol proses antara lain proses mengakses form kompresi, proses mengakses form MPE dan tombol proses keluar dari form utama.

2. Tampilan Pengujian Form Kompresi

Gambar 7 Implementasi Form Kompresi

Gambar di atas tampak proses pengujian terhadap form kompresi. Pada tampilan form kompresi dilakukan proses pengaksesan objek yang akan di kompresi dengan cara mengklik tombol browser sehingga sistem mengarahkan ke direktori penyimpanan dimana objek diletakkan. Setelah proses browser objek kemudian memilih metode kompresi sesuai yang ada dikomponen *radio button* pada form, kemudian dilakukan proses kompresi dengan menekan tombol “KOMPRESI” maka sistem akan menjalankan proses tersebut dengan menampilkan hasil kompresi antar lain ukuran file setelah dikompresi, *ratio of compression*, *compression ratio* dan *spacing saving*.

Tampilan Pengujian Form MPE
 (Metode Perbandingan Eksponensial)

The screenshot shows the 'FORM PERBANDINGAN' window. It has two main sections for LZW and Levensthein algorithms. Each section includes fields for 'Masukkan File Tekompres', 'Size', 'Ratio Of Compression', 'Compression Ratio', and 'Space Saving'. The 'PROSES PERBANDINGAN' button is highlighted in blue. Below these are fields for 'Rangking 1' and 'Rangking 2', and a 'PROSES PERANGKINGAN' button. At the bottom are 'RESET' and 'KELUAR' buttons.

Gambar 8 Pengujian Form MPE

Pada gambar di atas tampak proses pengujian form MPE, dimana untuk melakukan proses pebandingan metode eksponensial, tahapan pertama yang dilakukan mengunggah file hasil kompresi metode LZW dan file hasil kompresi metode Leveinsthen. Jika sudah diunggah/browser maka sistem akan membaca spesifikasi file hasil kompresi baik hasil kompresi metode LZW maupun metode Leveinsthen. Setelah itu masuk tahapan perbandingan dengan menekan tombol “PROSES PERBANDINGAN” sehingga menampilkan nilai hasil proses metode eksponensial untuk masing-masing metode. Lebih jelas dapat dilihat pada gambar dibawah ini.

This screenshot shows the same application window after the comparison process. The 'PROSES PERBANDINGAN' button is highlighted in blue. The 'Rangking 1' and 'Rangking 2' fields now contain the results of the comparison: 'Rangking 1' is 'LEVENSTHEIN' and 'Rangking 2' is 'LZW'. The 'PROSES PERANGKINGAN' button is also visible.

Gambar 8 Implementasi Proses Perbandingan

Setelah proses menekan tombol “PROSES PERBANDINGAN” maka selanjutnya melakukan proses perangkingan dengan menekan tombol “PROSES PERANGKINGAN” seperti pada gambar berikut ini.

This screenshot shows the application window after the ranking process. The 'PROSES PERANGKINGAN' button is highlighted in blue. The 'Rangking 1' and 'Rangking 2' fields now contain the results of the ranking: 'Rangking 1' is 'LEVENSTHEIN' and 'Rangking 2' is 'LZW'. The 'PROSES PERBANDINGAN' button is also visible.

Gambar 9 Implementasi Proses Perangkingan

D. Kesimpulan

Berdasarkan hasil analisis dan pembahasan sebelumnya, dapat disimpulkan beberapa poin penting: pertama, proses kompresi file PPT melibatkan tahapan penentuan objek dokumen, ekstraksi nilai hexadecimal menggunakan aplikasi binary view,

dan penerapan formula *algoritma* LZW serta *Levenshtein* pada sampel file. Kedua, penerapan *algoritma* LZW dan *Levenshtein* pada kompresi dokumen *.PPT menunjukkan hasil yang berbeda, dengan *algoritma* LZW mencapai Compression Ratio sebesar 80,81%, sedangkan *algoritma Levenshtein* mencapai 68,75%. Ketiga, pengujian yang dilakukan melalui aplikasi kompresi dokumen *.PPT dengan kedua *algoritma* menunjukkan bahwa, berdasarkan analisis kinerja menggunakan metode eksponensial, *algoritma Levenshtein* terbukti lebih efektif dan optimal dibandingkan *algoritma* LZW dalam kompresi file teks dengan ekstensi *.PPT.

DAFTAR PUSTAKA

- Adi. (2017). The Best Tips & Trik MS Office: PowerPoint. In *Elex Media Komputindo*.
- Akbar. (2017). POWER POINT, CANVA DAN MOVIE MAKER. In *CV Cendekia Press*.
- Alqori, N. F. (2024). Penerapan Algoritma Elias Delta Code Pada Aplikasi Kompresi File Gambar Berbasis Desktop. *VIRTUAL : Jurnal Teknik Informatika Dan Komputer*, 1(1), 9–19.
- Mentari. (2020). The Application Of Powerpoint Learning Media To Improve Learning Outcomes Of Science Subjects In SMP Negeri 1 Kahayan Kuala. *Jurnal Pendidikan Teknologi Informasi*, 4(1), 55–62.
- Pirnando, A. A., & Febryansyah, R. (2022). PARAREL DENGAN ALGORITMA LEMPEL ZIV WELCH. *Teknologipintar*, 2(10), 1–16.
- Putra, R., Sembiring, A. S., & Siregar, S. R. (2022). Analisa Perbandingan Algoritma Yamamoto ' s Recursive Code Dan Algoritma Fibonacci Code Dalam Mengkompresi File Pptx. *KOMIK (Konferensi Nasional Teknologi Informasi Dan Komputer) Volume*, 6(November), 448–459.
- Sandy. (2017). Power Point Android. In *Ahlimedia Book*.
- Silaban, B. A. (2022). Analisa Kompresi File Teks Dengan Kombinasi Metode Burrows-Wheeler Transform Dan Shannon-Fano. 6(November), 707–715.
<https://doi.org/10.30865/komik.v6i1.5760>
- Sipapaga, R. L. (2024). Penerapan Algoritma Elias Omega Code Pada Kompresi Data Teks. *VIRTUAL : Jurnal Teknik Informatika Dan Komputer*, 1(2), 30–35.
- SURIANSHA. (2018). Pelatihan Presentasi Dengan Menggunakan Microsoft PowerPoint Pada PT ALTOHA-Jakarta Utara. In *CV. AE MEDIA GRAFIKA* (p. 2018).
- Yuliastuti, G. E. (2023). Implementasi

Algoritma *Levenshtein* untuk
Kompresi File Audio. *SNESTIK*,
5(2), 314–320.