ANALISA PERBANDINGAN ALGORITMA INTERPOLATIVE CODING DAN RICE CODE UNTUK KOMPRESI FILE AUDIO (MP3)

Lidya Indah Pratama Simanungkalit[1], Imam Saputra[2], Putri Ramadhani[3]
[1,2,3]Universitas Budi Darma Medan
Email: [1]idyaindahpratama07@gmail.com, [2]saputraimam69@gmail.com,
[3]pramadhani@gmail.com

ABSTRAK

Penelitian ini bertujuan untuk menerapkan algoritma interpolative coding dan rice code dalam mengkompresi file audio MP3, serta membandingkan efektivitas kedua algoritma tersebut dalam proses kompresi. Selain itu, penelitian ini juga berfokus pada pengukuran rasio kompresi dari masing-masing algoritma untuk menilai seberapa efisien masing-masing metode dalam mengurangi ukuran file MP3. Saat ini, penggunaan aplikasi sebagai informasi semakin umum digunakan. Tetapi ada masalah yang sering dijumpai, yaitu kebutuhan akan tempat besar sebagai media penyimpanannya. Oleh sebab itu, dibutuhkan kompresi untuk memperkecil ukuran file tersebut. Kompresi file adalah suatu proses pengubahan sekumpulan data menjadi suatu bentuk kode untuk menghemat kebutuhan tempat penyimpanan data. Terdapat beberapa algoritma dalam kompresi data salah antara lain *Interpolative Coding* dan *Rice Code*. Banyaknya algoritma kompresi menyebabkan terjadinya ketidaktahuan algoritma mana sajakah yang lebih akurat dalam melakukan kompresi data. Maka perlu adanya perbandingan algoritma kompresi contohnya perbandingan algoritma *Interpolative Coding* dan *Rice Code*. Dengan dilakukannya perbandingan algoritma tersebut maka akan didapatkan algoritma mana yang lebih akurat dalam melakukan kompresi file audio MP3. Dengan parameter perbandingan algoritma yaitu rasio kompresi dan space saving dari algoritma *Interpolative Coding* dan *Rice Code*.

Kata Kunci : kompresi, perbandingan, interpolative coding, rice code

ABSTRACT

This study aims to apply the interpolative coding and rice code algorithms for compressing MP3 audio files and to compare the effectiveness of these two algorithms in the compression process. Additionally, the study focuses on measuring the compression ratios of each algorithm to evaluate the efficiency of each method in reducing MP3 file sizes. As the use of applications for information management becomes increasingly common, a frequent issue arises: the need for significant storage space. Therefore, compression is required to reduce the size of these files. File compression is the process of transforming a set of data into a coded format to minimize storage requirements. There are several data compression algorithms, including Interpolative Coding and Rice Code. The multitude of compression algorithms creates uncertainty about which algorithms are most accurate for data compression. Thus, a comparison between algorithms, such as Interpolative Coding and Rice Code, is necessary. This comparison will help determine which algorithm is more accurate for compressing MP3 audio files,

using parameters such as compression ratio and space saving of the Interpolative Coding and Rice Code algorithms.

Keywords: compression, comparison, interpolative coding, rice code

A. Pendahuluan

Seiring dengan pesatnya perkembangan teknologi dan pertumbuhan eksponensial dalam volume data yang dihasilkan serta dikelola, kompresi data telah menjadi komponen yang semakin krusial dalam pengelolaan informasi. Dengan semakin banyaknya data yang dihasilkan dari berbagai sumber seperti perangkat *mobile, sensor IoT, media sosial*, dan aplikasi bisnis, tantangan untuk menyimpan, mengelola, dan mentransfer data dengan efisien semakin besar. Kompresi data membantu mengatasi tantangan ini dengan mengurangi ukuran file dan volume data yang harus disimpan dan diproses, sehingga mengurangi kebutuhan akan kapasitas penyimpanan yang besar dan mempercepat proses transmisi data melalui jaringan. Teknik kompresi yang efektif juga berkontribusi pada penghematan biaya infrastruktur penyimpanan dan peningkatan performa sistem, karena data yang lebih kecil dapat diakses dan diproses lebih cepat (Mahendra 2024);(Ramdani 2024). Dengan demikian, inovasi dalam teknik kompresi data tidak hanya meningkatkan efisiensi operasional dan manajerial tetapi juga mendukung kemampuan untuk menangani data dalam skala besar yang semakin berkembang di era digital saat ini.

Kompresi data adalah teknik penting dalam pengolahan informasi yang bertujuan untuk mengurangi ukuran data sehingga lebih efisien dalam hal penyimpanan dan pertukaran. Prinsip dasar dari kompresi data adalah mengurangi jumlah bit yang diperlukan untuk merepresentasikan data tanpa kehilangan informasi penting, atau dengan kehilangan informasi yang sangat minimal dalam kasus kompresi lossy (Martina and Panjaitan 2021);(Imani et al. 2021). Dengan mengompresi data, ruang penyimpanan yang dibutuhkan menjadi lebih kecil, sehingga mengurangi kebutuhan akan kapasitas memori dan mengoptimalkan penggunaan sumber daya penyimpanan (Siswanto et al. 2023);(Alqori 2024). Ini juga berdampak pada kecepatan transfer data, karena file yang lebih kecil dapat dikirim lebih cepat melalui jaringan, meningkatkan efisiensi dalam komunikasi data. Seiring dengan perkembangan teknologi dan pertumbuhan eksponensial dalam volume data yang dihasilkan dan dikelola, kompresi data menjadi semakin krusial. Kapasitas penyimpanan yang lebih besar, yang didorong oleh teknik kompresi yang efektif, dapat mengurangi biaya infrastruktur penyimpanan dan memperbaiki performa sistem dengan mengurangi waktu akses data. Oleh karena itu, inovasi dalam teknik kompresi data memiliki peran yang sangat penting dalam mengelola data besar dan meningkatkan efisiensi operasional dalam berbagai aplikasi teknologi modern.

Format MP3 (MPEG-1 Audio Layer 3) adalah salah satu format audio digital yang paling populer saat ini (Setiawan and Hartati 2005). MP3 adalah singkatan dari MPEG-1 Audio

Layer 3. Ini adalah format kompresi audio digital yang sangat populer yang digunakan untuk menyimpan, mengirim, dan memutar musik dan audio lainnya (Situmorang 2023). MP3 memungkinkan pengguna untuk mengompresi file audio ke ukuran yang lebih kecil tanpa kehilangan banyak kualitas suara (Sari 2018). MP3 adalah salah satu format audio yang paling banyak digunakan di dunia, karena ukurannya yang kecil dan kualitas suaranya yang baik (Willfrid et al. 2016);(Kusuma 2017). MP3 adalah format standar untuk musik digital dan sering digunakan untuk mengunduh dan memutar musik di perangkat portabel seperti iPod, ponsel, dan pemutar MP3 lainnya (Den Uijl, S., de Vries, H. J., & Bayramoglu 2013). Format ini menggunakan kompresi data untuk mengurangi ukuran file audio tanpa mengorbankan kualitas audio secara signifikan. Kompresi MP3 sangat penting untuk distribusi musik dan audio secara digital, terutama ketika durasi file audio panjang, sehingga ukuran file bisa menjadi besar dan perlu dikompresi untuk menghemat ruang penyimpanan. Dalam kompresi file MP3, dua algoritma yang sering digunakan adalah interpolative coding dan rice code (Irfansyah 2022). *Algoritma interpolative coding* merupakan teknik kompresi lossless yang memungkinkan data terkompresi dikembalikan ke bentuk aslinya tanpa kehilangan informasi (Finola 2019);(Prambudi 2020). Dengan algoritma ini, file audio besar dapat dikompresi menjadi ukuran yang lebih kecil, sehingga proses transmisi menjadi lebih cepat dan efisien dalam hal ruang penyimpanan.

Penelitian ini membahas perbandingan antara *algoritma interpolative coding* dan *rice code* dalam hal kompresi file audio. Perbandingan dilakukan untuk menentukan algoritma mana yang lebih efektif dalam proses kompresi dengan menggunakan variabel seperti rasio kompresi dan penghematan ruang penyimpanan. Setiap algoritma memiliki kelebihan tersendiri, sehingga pemilihan algoritma yang tepat dapat mempengaruhi hasil kompresi. Algoritma interpolative coding adalah metode inovatif yang menetapkan kode dinamis untuk simbol data berdasarkan keseluruhan pesan, bukan hanya frekuensi simbol. Proses ini melibatkan pemindaian pesan ke dalam urutan khusus, yang dapat menghasilkan codeword dengan ukuran yang sangat kecil. Di sisi lain, rice code, yang ditemukan oleh Robert F. Rice, adalah teknik entropy encoding yang juga kompresi data tanpa mengurangi isi data dari file asli.

Penelitian yang dilakukan oleh Finola, (2019) dan Irfansyah, (2022) menunjukkan hasil yang bervariasi. Finola menemukan bahwa *interpolative coding* efektif dalam menghemat memori penyimpanan untuk file audio, sedangkan Irfansyah melaporkan bahwa *rice code* lebih efisien dalam kompresi video dibandingkan *interpolative coding*. Berdasarkan temuan ini, penelitian ini bertujuan untuk menganalisis dan membandingkan kedua algoritma dalam kompresi file MP3 untuk menentukan metode yang paling akurat dan efisien.

B. Metode Penelitian

Pada metodologi penelitian dilakukan klasifikasi tahapan yang dipakai pada penelitian**.** Metodologi penelitian terdiri atas tahapan yang mempunyai kaitan secara sistematis. Berikut merupakan tahapan

sederhana proses pengumpulan data dari penelitian yang dilakukan pada penjelasan diatas dapat dilihat pada gambar dibawah ini. **Tahap 1** Identifikasi, **Tahap 2** Studi Pustaka **Tahap 3** Analisa Proses Kompresi Dan Dekompresi Pada *File* Audio **Tahap 4** Perbandingan **Tahap 5**Perancangan.

Waktu Pelaksanaan

Waktu pelaksanaan penelitian dilaksanakan terhitung sejak diterima usulan penelitian terkait dengan judul penelitian yaitu Analisa Perbandingan Algoritma *Interpolative Coding* dan *Rice Code* dalam kompresi *file* audio sampai dengan selesai. Adapun waktu pelaksaan penelitian akan disajikan dalam bentuk tabel di bawah ini.

Tabel 1. Waktu Pelaksanaan Penelitian

No	Kegiatan	Februari 2024				Maret 2024				April 2024				Mei 2024				Juni 2024			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Idetifikasi Masalah																				
2	Studi Pustaka																				
3	Analisa Proses Kompresi Dan Dekompresi Pada *File* Audio																				
4	Perbandingan																				
5	Perancangan																				
6	Implementasi																				
7	Pengujian																				
8	Dokumentasi																				

C. Hasil Penelitian dan Pembahasan

Berdasarkan analisa, kompresi file audio dengan ekstensi Mp3 memiliki ukuran yang sangat besar. Dengan dilakukannya kompresi file audio, file yang memiliki ukuran lebih besar akan dikompres agar ukuran lebih kecil serta menghemat ruang penyimpanan serta diketahui algoritma yang lebih akurat dalam melakukan kompresi file audio dengan algoritma interpolative coding dan rice code. Pada penelitian ini, akan dibahas dua proses utama ialah proses kompresi serta proses dekompresi, dan peneliti melakukan

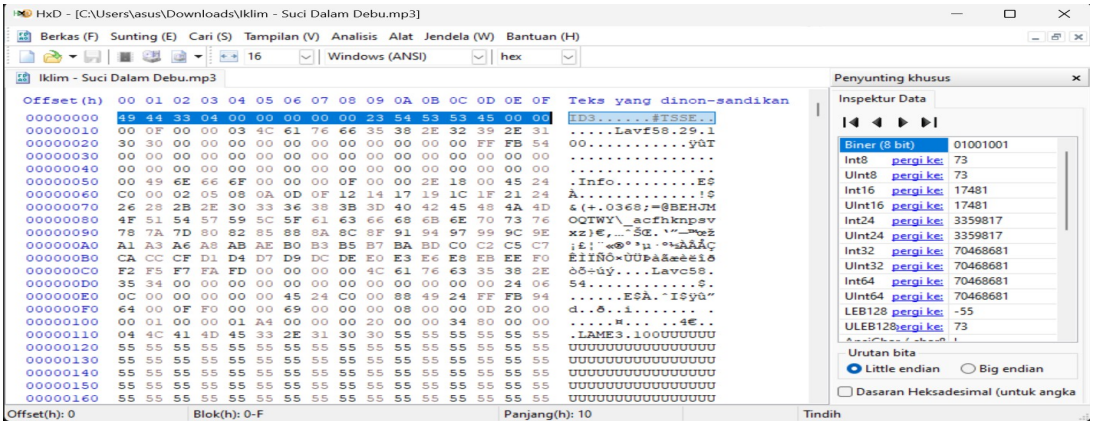
kompresi menggunakan dua algoritma yaitu algoritma interpolative coding dan rice code.

Sebelum *file* audio dikompres, lebih dulu dilakukan pembacaan *file* audio agar didapatkan nilai hexadesimal menggunakan aplikasi HxD. Berikut contoh *file* audio yang dikompres serta didekompresi.

Tabel 2. Sampel File MP3 yang akan dikompresi

Nama File	Iklim – Suci Dalam Debu
Extension File	MP3
Size	4,32 MB

Berdasarkan Audio diatas, maka *file* Audio tersebut dapat dikonversi ke nilai *hexadecimal* menggunakan aplikasi *HxD*.



Gambar 1. Sample

Analisa Proses Kompresi *File* Audio MP3 Menggunakan Algoritma Interpolative Coding

Pada algoritma *Interpolative coding* digunakan sampel data sebanyak 16 karakter untuk keperluan perhitungan manual. Nilai *hexadesimal* diambil dari sisi kiri ke kanan.

Tabel 3. Nilai Bilangan *Hexadecimal*

49	44	33	04	00	00	00	00	00	23
54	53	53	45	00	00				

Tabel di atas berguna untuk menghasilkan angka *hexadecimal* untuk *file* mp3 yang akan dihitung secara manual. Berikut nilai *hexadecimal* yang diurutkan berdasarkan frekuensi, nilai frekuensi tertinggi akan berada di tempat pertama.

Tabel 4. Nilai *hexadecimal* tidak terkompresi

Nilai		Bit	Frek	Bit x Frek
Hexa	Biner			
00	00000000	8	7	56
53	01010011	8	2	16
49	01001001	8	1	8
44	01000100	8	1	8
33	00110011	8	1	8
04	00000100	8	1	8
23	00100011	8	1	8

54	01010100	8	1	8
45	01000101	8	1	8
Total Bit				128

Tabel 5. Kompresi Nilai *File* MP3 *Sample* Dengan *Interpolative Coding*

N	Nilai Hexa	*Codeword*	Bit	Frek	Bit x Frek
1	00	00	2	7	14
2	53	001	3	2	6
3	49	001	3	1	3
4	44	00	2	1	2
5	33	001	3	1	3
6	04	00	2	1	2
7	23	00	2	1	2
8	54	00	2	1	2
9	45	00	2	1	2
Total Bit					36

Pada perhitungan tabel di atas setelah dikompresi dengan menggunakan algoritma *Interpolative Coding,* didapat total bitnya ialah 36 bit. Sebelum dilakukan hasil *string bit Interpolative Coding* menjadi nilai *file*, terlebih dulu lakukan pemeriksaan pada panjang *string bit*. Pembentukan nilai bit baru hasil kompresi pada susunan nilai heksadesimal sebelum kompresi yaitu 49, 44, 33, 04,
00, 00, 00, 00, 00, 23, 54, 53, 53, 45,
00, 00 menjadi nilai bit biner :
"00100001000000000000000000001001
000000"

Lalu sebelum didapat hasil semua akhir kompresi dilakukan penambahan *string bit* yaitu *padding* serta *flag bit* yaitu dengan menggunakan 36 mod 8 = 4. Jika sisa bagi panjang *string bit* terhadap 8 ialah 0 maka ditambahkan 0000001. Dinyatakan dengan bit akhir. Sedangkan jika sisa bagi panjang *string bit* terhadap 8 adalah n(1,2,3,4,5,6,7) maka tambahkan 0 sebanyak 7-n+"1" diakhir *string bit*. Nyatakan dengan L, lalu tambahkan 0 bilangan biner dari 9 –n, nyatakan dengan bit akhir. Karena jumlah string bit 36 tidak habis dibagi 8, maka dinyatakan sisa bagi tersebut dengan nilai *n.* Maka tambahkan 0 sebanyak 7 – n + "1" akhir string bit. Nyatakan dengan L. Lalu tambahkan bilangan biner dari 9 – n. Nyatakan dengan bit akhir.

$7 - 4 + \text{"1"}$

7 – 4 + "1" = **0001**

Bit Akhir $9 - n$

Bit Akhir = 9 – 4 = 5 = **00000101**

Gambar 1. Perhitungan Penambahan bit

Setelah dilakukan penambahan bit, maka terbentuk string bit sebagai berikut :

00100001, 00000000, 00000000, 00100100, 0000**0001**, **00000101**

Gambar 2. *String* bit yang telah dilakukan penambahan

Total panjang bit keseluruhan setelah ada dilakukan tambah nilai bit ialah 36+4+8 =48. Lalu lakukan pemisahan bit ke beberapa kelompok. Perkelompok terdiri atas 8 bit seperti gambar di bawah ini

00100001 00000000 00000000 00100100 0000**0001 00000101**

Gambar 3. Pembagian *String* bit

Berdasarkan pada pembagian kelompok nilai biner, didapatkan 6 kelompok nilai biner baru (6 *byte*) yang telah dikompresi nilai biner penambahan bit. Lalu dilakukan pembagian.

00100001 00000000 00000000 00100100 0000**0001 00000101**

! **0** **0** **$** **1**

Gambar 4. Karakter nilai Biner Terkompresi

Kemudian string bit biner di atas diubah ke heksadesimal, maka setiap bilangan dibaca dalam karakter heksadesimal.

Tabel 6 Nilai *Hexadecimal* Terkompresi

Urutan Nilai Terkompresi	Nilai *Hexadecimal* Terkompresi
1	21
2	0
3	0
4	24
5	1
6	5

Ukuran data sebelum dikompresi = 128 Ukuran data sesudah dikompresi = 36 Analisa Proses Dekompresi *File* Audio MP3 Menggunakan Algoritma.

Interpolative Coding

Ada beberapa tahap dalam menganalisa proses dekompresi *file* mp3 menggunakan algortima *Interpolative Coding* adalah sebagai berikut:

1. Memasukkan hasil kompresi

   Nilai heksadesimal dari hasil kompresi yaitu 21, 00, 00, 24, 1, 5.
2. Hasil *string bit interpolative coding* yang dijadikan nilai *file* diubah ke dalam bentuk biner. Bit keseluruhan hasil kompresi dapat dilihat pada tabel di bawah ini.

Tabel 7. Nilai *heksadesimal* dan biner kompresi

Urutan Nilai Terkompresi	Nilai *Hexadecimal* Terkompresi	Nilai Biner
1	21	00100001
2	00	00000000
3	00	00000000
4	24	00100100
5	1	00000001

6	5	00000101

3. Berdasarkan tabel di atas maka diambil keseluruhan nilai biner dan digabungkan menjadi 00100001 00000000 00000000 00100100 00000001 00000101 Dikembalikan *binary* menjadi *string bit* awal dengan melakukan penghilangan biner yang ditebalkan. Dalam pengembalian *binary* menjadi *string bit* awal dapat dilakukan dengan langkah berikut. Dilakukan pembacaan *string bit* pada 8 bit terakhir, hasil pembacaan dalam bentuk bilangan desimal. Dinyatakan hasil yang dibaca dengan n. Hilangkan bit dalam bagian akir dengan rumus 7 + n = 7+5=12. Maka sebanyak 12 bit yang dihilangkan pada *string bit*.

Setelah hitungan dilakukan pembacaan bit akhir. Hasil akhir string bitmenjadi *string bit* semula yaitu: **"**00100001 00000000 00000000 00100100 0000"

Dari hitungan diatas menggunakan algoritma *Interpolative coding*, string bit di atas bernilai 36 seperti diawal hingga dilakukan pembacaan string bit awal. Adapun tabel hasil perhitungan di atas ialah:

Tabel 8. Nilai *file* dan *interpolative Coding*

N	Kode Interpolative Coding	Nilai *File*
1	49	001
2	44	00
3	33	001
4	04	00
5	00	00
6	00	00
7	00	00
8	00	00
9	00	00
10	23	00
11	54	00
12	53	001
13	53	001
14	45	00
15	00	00
16	00	00

Dari hasil dekompresi di atas, didapatkan nilai heksadesimal awal sebelum kompresi yaitu 49, 44, 33, 04, 00, 00, 00, 00, 00, 23, 54, 53, 53, 45, 00, 00.

Analisa Proses Kompresi *File* Audio MP3 dengan Algoritma *Rice Code*

Dari gambar diatas didapat nilai hexadesimal *file* MP3 sampel data. Untuk keperluan perhitungan manual, maka hanya akan diambil sampel nilai sebanyak 16 karakter nilai *hexadesimal file* MP3 sampel. Nilai *hexadesimal* diambil dari sisi kiri ke kanan. Melakukan pembacaan isi *file* nilai hexadesimal *file* MP3 ini ialah 49, 44, 33, 04, 00, 00, 00, 00, 00, 23, 54, 53, 53, 45, 00, 00. Nilai data ini dimasukkan dalam tabel agar dilakukan pembacaan frekuensi. Frekuensi dibaca dengan perhitungan jumlah nilai yang sama disetiap nilai yang muncul. Pembacaan frekuensi

dapat dilihat pada tabel dibawah ini:

Tabel 8. Nilai *File*

Nilai	Frekuensi
00	7
53	2
49	1
44	1
33	1
04	1
23	1
54	1
45	1

Selanjutnya dilakukan pengurutan karakter yang memiliki frekuensi paling besar (banyak nilai yang sama) ke frekuensi paling kecil. Urutan nilai dapat dilihat dalam tabel di bawah:

Tabel 9. Nilai *Bit* MP3 *Sample*

Nilai		Bit	Frek	Bit x Frek
Hexa	Biner			
00	00000000	8	7	56
53	01010011	8	2	16
49	01001001	8	1	8
44	01000100	8	1	8
33	00110011	8	1	8
04	00000100	8	1	8
23	00100011	8	1	8
54	01010100	8	1	8
45	01000101	8	1	8
Total Bit				128

Dari tabel di atas, satu nilai hexadesimal (karakter) bernilai 8 bit bilangan biner. Hingga 16 bilangan *hexadesimal* memiliki nilai biner dengan jumlah 128 bit. Dalam pengubahan satuan menjadi byte maka jumlah seluruh bit dibagi 8. Maka dihasilkan 128/8 = 16. Kemudian dibentuk Tabel *Rice Code* Aturan pada pembentukan kode bilangan dengan memakai *rice code* dapat dilihat pada bab sebelumnya. Adapun kode *rice code* dapat dilihat pada tabel di bawah ini.

Tabel 10 Kode *Rice Code*

N	Kode *Rice Code*
1	000
2	001
3	010
4	011
5	1000

6	1001
7	1010
8	1011
9	10000

Proses berikutnya ialah dilakukan kompresi nilai pada sample dengan kode *rice code* yang didapat pada tabel di bawah. Adapun proses kompresi *file* MP3 sample dapat di lihat pada tabel berikut :

Tabel 11 Kompresi Nilai *File* MP3 *Sample* Dengan *Rice Code*

N	Nilai Hexa	Kode *Rice Code*	Bit	Frek	Bit x Frek
1	00	000	3	7	21
2	53	001	3	2	6
3	49	010	3	1	3
4	44	011	3	1	3
5	33	1000	4	1	4
6	04	1001	4	1	4
7	23	1010	4	1	4
8	54	1011	4	1	4
9	45	10000	5	1	5
Total Bit					54

Pada perhitungan tabel di atas setelah dikompresi dengan memakai *stout code* ialah 54 bit.

Analisa Proses Dekompresi *File* Audio MP3 Menggunakan Algoritma

Rice Code

Ada beberapa tahap dalam menganalisa proses dekompresi *file* mp3 menggunakan algoritma *Rice Code* ialah sebagai berikut:

1. Memasukkan hasil kompresi
   Nilai heksadesimal dari hasil kompresi yaitu 4E, 24, 0, 5, 59, 30, 1, 3, 4E.
2. Hasil *string bit rice code* yang dijadikan nilai *file* diubah ke dalam bentuk biner. Bit keseluruhan hasil kompresi dapat dilihat pada tabel di bawah ini.

Tabel 13 Nilai *heksadesimal* dan biner kompresi

Urutan Nilai Terkompresi	Nilai *Hexadecimal* Terkompresi	Nilai Biner
1	4E	01001110
2	24	00100100
3	0	00000000
4	5	00000101
5	59	01011001
6	30	00110000
7	1	00000**01**
8	3	**00000011**

Berdasarkan tabel di atas maka diambil keseluruhan nilai biner dan digabungkan menjadi

01001110 00100100 00000000 00000101 01011001 00110000 00000**01**

00000011

3. Dikembalikan *binary* menjadi string bit awal dengan melakukan penghilangan biner yang ditebalkan. Dalam pengembalian *binary* menjadi string bit awal dapat dilakukan dengan langkah berikut. Dilakukan pembacaan string bit pada 8 bit terakhir, hasil pembacaan dalam bentuk bilangan desimal. Dinyatakan hasil yang dibaca dengan n. Hilangkan bit dalam bagian akir dengan rumus 7 + n = 7 + 3=10. Maka sebanyak 12 bit yang dihilangkan pada string bit. Setelah hitungan dilakukan pembacaan bit akhir. Nilai biner yang dihilangkan berjumlah 8 bit pada akhir n=1. Dari yang dijelaskan diatas ditunjukkan jika bit akhir harus dihilangkan. Hasil bagi binary menjadi string bit semula adalah : "0100111000100100000000000000101010110010011000000000"

Dari hitungan diatas menggunakan algoritma *rice code*, string bit di atas bernilai 54 seperti diawal hingga dilakukan pembacaan string bit awal. Adapun tabel hasil perhitungan di atas ialah:

Tabel 14 Nilai *file* dan *Rice Code*

N	Kode *Rice Code*	Nilai *File*
1	49	010
2	44	011
3	33	1000
4	04	1001
5	00	00
6	00	00
7	00	00
8	00	00
9	00	00
10	23	1010
11	54	1011
12	53	001
13	53	001
14	45	10000
15	00	00
16	00	00

Dari hasil dekompresi di atas, didapatkan nilai heksadesimal awal sebelum kompresi yaitu 49, 44, 33, 04, 00, 00, 00, 00, 00, 23, 54, 53, 53, 45, 00, 00

Perbandingan Algoritma *Interpoltive Coding* dan *Rice Code*

Perbandingan algortima *Interpolative Coding Rice Code* dan ialah membandingan *Ratio Compression (Rc), Compression Ratio (Cr)* , *redudancy* dan *space saving* dari kedua algoritma. Dimana setelah dilakukan perhitungan dengan algoritma *Rice Code* dan *Interpolative Coding* maka hasil dari *Ratio Compression (Rc)*, *Compression Ratio (Cr)*, *Redudancy* dan *space saving* didapatkan. Setelah hasil dari kedua algoritma didapatkan, lalu dilakukan perbandingan untuk mengetahui algoritma mana yang lebih efektif dalam melakukan proses kompresi *file* audio MP3. Adapun perbandingan

kedua algoritma dapat dilihat pada tabel dibawah ini.

Tabel 15 Nilai Perbandingan

No	Algoritma	Rc	Cr	Rd	Ss
1	*Interpolative Coding*	3,5	28,125%	71,875%	71,875%
2	*Rice Code*	2,3	42,18%	57,8%	57,8%

Berdasarkan penjelasan pada tabel diatas, maka dapat diketahui bahwa algoritma *Rice code* lebih efektif dibandingkan dengan algoritma *Interpolative coding*. Semakin besar persentase dari *Compression ratio* maka akan semakin baik pula sebuah algoritma dalam melakukan proses kompresi. Dari tabel di atas dapat dilihat bahwa persentase *compression ratio* dari algoritma *rice code* lebih besar daripada algoritma *interpolative coding*.

Implementasi Program

Implementasi program merupakan tahap uji coba dari sistem yang dibangun, pada bagian implementasi program membahas spesifikasi perangkat keras, perangkat lunak dan hasil dari tampilan sistem ketika sedang berjalan.

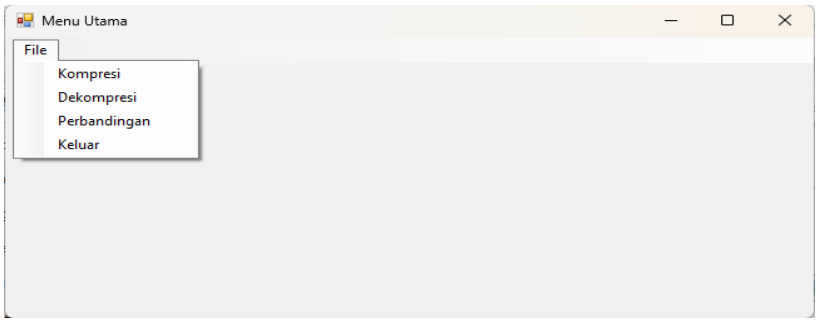
Tampilan *Input*

Tampilan *input* merupakan tampilan yang akan menjelaskan *form* aplikasi yang digunakan untuk kebutuhan program yang akan dibutuhkan.

1. *Form* Halaman Utama

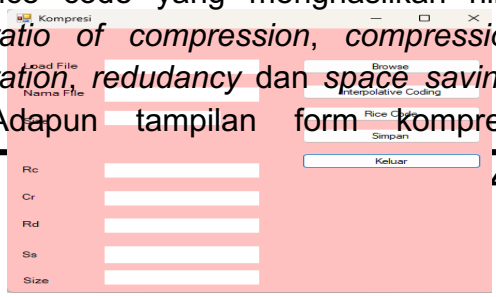
   Form halaman utama adalah *form* yang pertama kali muncul ketika aplikasi perbaningan algoritma interpolative coding dan rice dalam kompresi file MP3 dibuka. Adapun gambar form halaman utama dapat dilihat pada gambar di bawah ini

Gambar 5 Tampilan Form Menu Utama



Form Kompresi

Form kompresi adalah form yang digunakan untuk melakukan proses kompresi dengan algoritma interpolative coding dan algoritma rice code yang menghasilkan nilai *ratio of compression*, *compression ration*, *redudancy* dan *space saving*. Adapun tampilan form kompresi



dapat dilihat pada gambar di bawah ini.

Gambar 6 Tampilan Form Kompresi

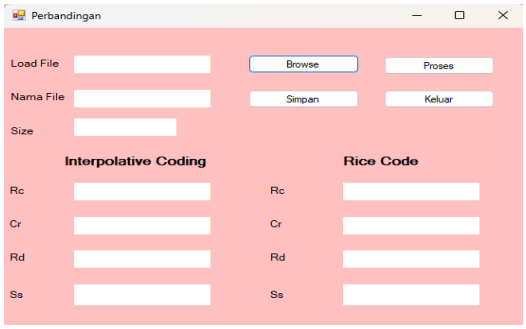
Form Dekompresi

Form dekompresi adalah form yang digunakan untuk melakukan proses dekompresi file dengan algoritma interpolative coding dan algoritma rice code. Adapun tampilan form dekompresi dapat dilihat pada gambar di bawah ini.

Gambar 7 Form Dekompresi

Form Perbandingan

Form perbandingan adalah form yang digunakan untuk menampilkan nilai ratio of compression, compression ratio, redudancy dan space saving dari algoritma interpolative coding dan rice code. Adapun tampilan form perbandingan dapat dilihat pada gambar di bawah ini.



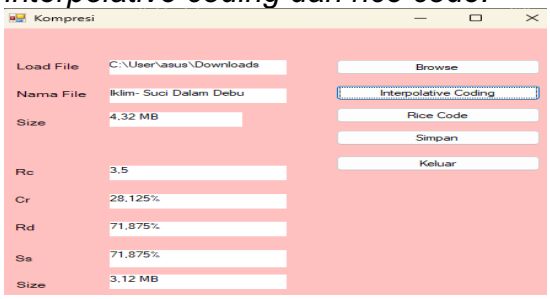
Gambar 8 Form Perbandingan

Tampilan *Output*

Tampilan *output* merupakan tampilan proses dan hasil dari algoritma *interpolative coding* dan rice code yang nantinya akan dilakukan perbandingan. Adapun tampilan output antara lain.

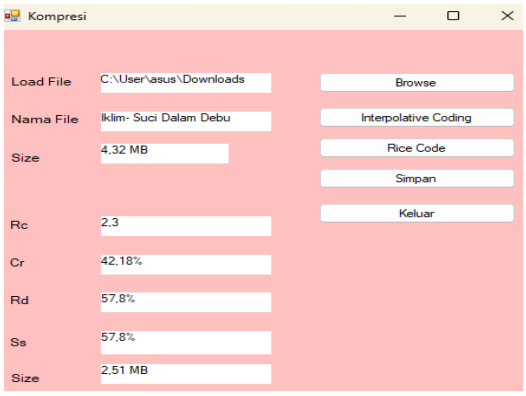
1. Form Kompresi

Tampilan output form kompresi berisi nilai ratio compresion, compression of ratio, redudancy dan space saving dari algoritma *interpolative coding dan rice code.*



Gambar 9 Hasil Kompresi *Algoritma Interpolative Coding*

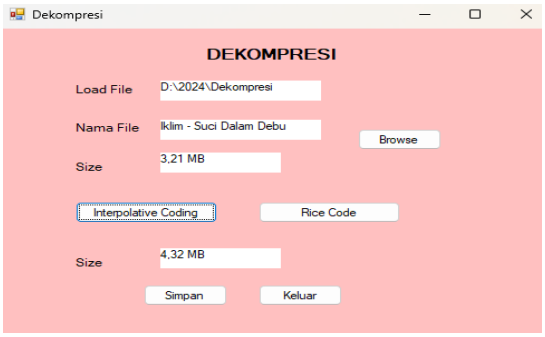
Dari gambar di atas maka dapat dilihat hasil yang didapatkan setelah dilakukan proses kompresi dengan algrotima *interpolative coding*. Tombol browse pada gambar di atas berfungsi untuk mengambil file yang akan dilakukan proses kompresi. Tombol interpolative coding digunakan untuk melakukan proses kompresi dengan *algoritma interpolative coding*. Tombol *rice code* digunakan untuk melakukan proses kompresi dengan *algoritma rice code*. Tombol simpan digunakan untuk menyimpan file yang sudah dilakukan proses kompresi. Tombol keluar digunakan untuk keluar dari form kompresi. Adapun proses kompresi dengan algoritma rice code dapat dilihat pada gambar di bawah ini.



Gambar 10 Hasil Kompresi *Algoritma Rice Code*

Form Dekompresi

Form output dekompresi adalah form yang berisi hasil pengembalian file yang telah dilakukan proses kompresi dengan algoritma interpolative coding dan rice code. Adapun tampilan form dekompresi dapat dilihat pada gambar di bawah ini.

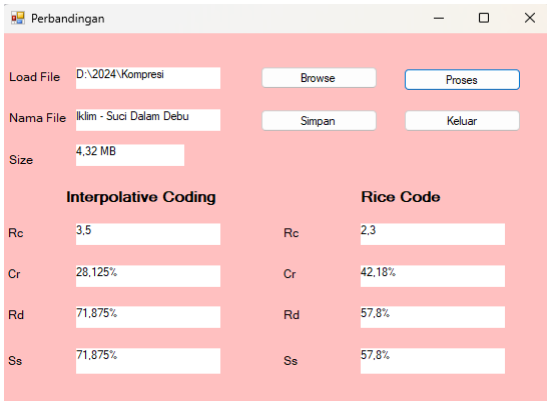


Gambar 11 Hasil Dekompresi Interpolative Coding

Gambar di atas merupakan hasil dekompresi dari algoritma algoritma interpolative coding. Dekompresi algoritma interpolative coding yaitu mengamblikan file yang sudah dikompresi kembali ke file ukuran awal.

Form Perbandingan

Form perbandingan adalah form yang berisi hasil perbandingan dari *algoritma interpolative coding* dan *rice code*. Adapun tampilan form perbandingan algoritma *interpolative coding* dan *rice code* dapat dilihat pada gambar di bawah ini.



Gambar 13 Tampilan Hasil Perbandingan

Dari gambar di atas didapatkan hasil dari perbandingan algoritma interpolative coding dan algoritma rice code dengan nilai *ratio compression, compression of ratio, redudancy dan space saving* dari masing-masing algoritma. Tombol browse pada form perbandingan digunakan untuk mengambil file yang akan dilakukan proses kompresi. Tombol proses digunakan untuk melakukan proses kompresi dari algoritma interpolative coding dan rice code. Tombol simpan digunakan untuk melakukan proses simpan file yang telah dikompresi sedangkan tombol keluar digunakan untuk keluar dari form perbandingan.

Hasil Pengujian

Perbedaan file MP3 sebelum dan sesudah dikompresi dari hasil pengujian dapat di lihat pada tabel di bawah ini :

Tabel 16 Hasil Pengujian Kompresi

Nam a File	Ukuran File	Interpolative Coding Rice Code									
		Interpolative Coding					Rice Code				
		Rc	Cr	Rd	SS	Siz e	Rc	Cr	Rd	SS	Size
Letto	4,91 MB	4,5	30,1 2	75, 80	75, 80	3,5 4 MB	3,3	44, 175	61, 725	61,7 25	3,16 MB
Padi	91,76 MB	5,5	40,1 4	80, 25	80,2 5	66, 07 MB	4,3	54, 195	66, 175	66,1 75	53,2 3 MB
Suci Dala m Debu	4,21	3,5	28,1 2	71, 87	71,8 7	3,1 2 MB	2,3	42, 18	57, 8	57,8	2,51 MB

D. Kesimpulan

Setelah menyelesaikan analisis, beberapa kesimpulan dapat ditarik. Pertama, kompresi file MP3 dapat dilakukan dengan mengonversi file ke dalam bentuk heksadesimal untuk memudahkan perhitungan, yang kemudian diproses menggunakan algoritma interpolative coding dan rice code. Proses ini menghasilkan nilai baru yang mengubah ukuran file MP3. Kedua, perbandingan antara algoritma *interpolative coding* dan *rice code* dilakukan dengan menggunakan parameter seperti *ratio of compression, compression ratio, redundancy, dan space saving*. Hasil perbandingan menunjukkan bahwa algoritma rice code lebih unggul dalam hal proses kompresi dibandingkan algoritma interpolative coding, terutama pada nilai *compression* ratio. Ketiga, hasil pengukuran rasio kompresi dan space saving untuk file MP3 yang diolah dengan interpolative coding menunjukkan nilai *Ratio of compression* sebesar 3,5, compression ratio 28,125%, *redundancy* 71,875%, dan *space saving* 71,875%. Sebaliknya, algoritma *rice code* menunjukkan nilai *Ratio of compression* sebesar 2,3, *compression ratio* 42,18%, *redundancy* 57,8%, dan *space saving* 57,8%.

DAFTAR PUSTAKA

Alqori, Nurma Fitri. 2024. “Penerapan Algoritma Elias Delta Code Pada Aplikasi Kompresi File Gambar Berbasis Desktop.” *VIRTUAL : Jurnal Teknik Informatika Dan Komputer* 1(1):9–19.

Finola, Riyo Oktavianty. 2019. “PENERAPAN ALGORITMA INTERPOLATIVE CODING UNTUK KOMPRESI FILE.” *KOMIK (Konferensi Nasional Teknologi Informasi Dan Komputer)* 3(1):378–84.

Imani, Moch Lazuardi, Rani Rotul Muhima, Siti Agustini, Teknik Informatika, Institut Teknologi,

and Adhi Tama. 2021. "Penerapan Metode Huffman Dalam Kompresi Data." *Seminar Nasional Sains Dan Teknologi Terapan IX* 1(1):457–62.

Irfansyah, Refo. 2022. "Perbandingan Algoritma Golomb Rice Dan Interpolative Coding Dalam Mengkompresi File Video." *KOMIK (Konferensi Nasional Teknologi Informasi Dan Komputer)* 6(2):250–59.

Kusuma, Indra Jaya. 2017. "ANALISIS TEKNIK STEGANOGRAFI PADA AUDIO MP3 MENGGUNAKAN." *Jurnal Elektronik Sistem Informasi Dan Komputer 1* 3(2).

Mahendra. 2024. "Teknologi Big Data: Pengantar Dan Penerapan Teknologi Big Data Di Berbagai Bidang." in *PT. Green Pustaka Indonesia.*

Martina, Jesica, and Br Panjaitan. 2021. "Penerapan Algoritma Fibonacci Codes Pada Kompresi Aplikasi Audio Mp3 Berbasis Dekstop." *BIMASATI (Bulletin of Multi-Disciplinary Science and Applied Technology)* 1(1):27–33.

Prambudi, Muhammad Eko. 2020. "Penerapan Algoritma Interpolative Coding Pada Aplikasi Kompresi File Teks." *Jurnal Pelita Informatika,* 8(1):444–48.

Ramdani. 2024. "INFORMATIKA TERAPAN JILID 2." in *CV. Intake Pustaka.*

Sari. 2018. "PENERAPAN ALGORITMA LEVENSTEIN PADA APLIKASI KOMPRESI FILE MP3." *KOMIK (Konferensi Nasional Teknologi Informasi Dan Komputer)* 2(1):1.

Setiawan, Mukhammad Andri, and Sri Hartati. 2005. "APLIKASI BERBASIS WEB UNTUK PENCARIAN MP3." *Media Informatika* 3(1):39–45.

Siswanto, Siswanto, Maya Utami Dewi, Siti Kholifah, Greget Widhiati, and Universitas Sains. 2023. "Penggunaan Model Deep Learning Untuk Meningkatkan Efisiensi Dalam Aplikasi Machine Learning Dan Hanya Berfokus Pada Pengoptimalan Arsitektur Perangkat Keras . Dalam Penelitian Ini." *Jurnal Penelitian Sistem Informasi* 1(4).

Situmorang, Tia Betsaida. 2023. "Perancangan Aplikasi Kompresi File MP3 Dengan Menggunakan Algoritma Lempel Ziv Welch (LZW) JURNAL MEDIA INFORMATIKA [JUMIN]." *JURNAL MEDIA INFORMATIKA [JUMIN]* 5(1):11–21.

Den Uijl, S., de Vries, H. J., & Bayramoglu, D. 2013. "The Rise of MP3 as the Market Standard: How Compressed Audio Files Became the Dominant Music Format." *International Journal of IT Standards and Standardization Research* 11(1):2013.

Willfrid, Darno, Midukta Simamora, Garuda Ginting, and Yasir Hasan. 2016. "IMPLEMENTASI ALGORITMA RUN LENGTH ENCODING PADA KOMPRESI FILE MP3." *Jurnal Riset Komputer (JURIKOM)* 3(4):5–9.