KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer)
Volume 6, Nomor 1, November 2022
DOI: 10.30865/komik.v6i1.5683
ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 260-270

Analisis Perbandingan Algoritma Prefix Code Dengan Elias Omega Code Dalam Merancang Aplikasi Pengkompresi File Video Avi

Arby Hartama

Fakultas Ilmu Komputer dan Teknologi Informasi, Teknik Informatika, Universitas Budi Darma Medan, Medan, Indonesia
Email: arbyhartama@gmail.com

Abstrak−Di bidang teknologi informasi, penyimpanan data biasanya dilakukan transmisi data menggunakan sistem transmisi elektronik. Besarnya ukuran file data terkadang menjadi kendala pengiriman data dengan ukuran besar akan memakan waktu yang cukup lama Khususnya pada file video. File video merupakan media informasi yang sering digunakan. Pada umumnya file video AVI merupakan format file yang cukup besar. Maka dengan Penelitian yang dilakukan dengan menggunakan algoritma Prefix code dan algoritma elias omega code untuk mengompresi file video dan untuk melakukan perbandingan, karena kedua algoritma memiliki cara kerja yang berbeda. Mengingat perbedaan ini, pemindaian diperlukan untuk membandingkan kedua algoritma untuk menentukan kompresi file mana yang lebih cepat. Metode eksponensial adalah metode yang memungkinkan Anda untuk melakukan analisis perbandingan dari dua algoritma. Dengan demikian, kita dapat melihat algoritma mana yang terbaik, efisien, dan cepat untuk mengompresi file video.

Kata Kunci: Analisis; Kompresi; File Video; Perbandingan; Prefix Code; Elias Omega Code

Abstract−In the field of information technology, data storage is usually carried out by transmitting data using an electronic transmission system. The large size of the data file becomes an obstacle in sending data with a large size which will take quite a long time, especially in video files. Video files are information media that are often used. In general, AVI video files are quite large file formats. So with the research conducted using the Prefix code algorithm and the elias omega code algorithm to compress video files and to perform comparisons, because the second algorithm has a different way of working. Given this difference, a scan is needed to compare the two algorithms to determine which file compression is faster. The exponential method is a method that allows you to perform a comparative analysis of two algorithms. Thus, we can see which yahoo is the best, efficient and fast to compress video files.

Keywords: Analysis; Compression; Video Files; Comparison; Prefix Code; Elias Omega Code

1. PENDAHULUAN

Pentingnya bagi semua orang dalam pertukaran dan penyimpanan data pada jaman ini yang dimana perkembangan teknologi sangat meningkat seiring berjalanan waktu. Masalah yang sering di hadapi tentunya kapasitas penyimpanan data yang sangat terbatas dan penyimpanan data yang lama. Oleh karna itu dibutuhkan cara kompresi data yang dimiliki untuk mempermudah proses pengiriman dan mengurangi kebutuhan kapasitas media penyimpanan data. Proses kompresi dilakukan dengan membandingkan beberapa algoritma untuk mengatasi permasalahan pada media penyimpanan, dimana metode kompresi menggunakan *Algoritma Prefix Code* dan *Algoritma Elias Omega Code*.

Kompresi data adalah cabang ilmu komputer yang berakar pada teori informasi[1]. Dalam makalahnya tahun 1948, "*Mathematical Theory of Communication*", Claude E. Shannon merumuskan teori kompresi data. Teori informasi berfokus pada berbagai metode penyimpanan dan pemrosesan data. Dalam teori ini juga disebutkan bahwa semakin banyak data yang redundan (data yang tidak dapat digunakan), semakin besar ukuran penyimpanan datanya. Oleh karena itu, untuk mengurangi redundansi data, dikembangkan teori informasi kompresi data *(file compression)* [2].

File Video AVI Pertama kali diperkenalkan pada tahun 1992 oleh Microsoft, video AVI adalah yang paling populer selama tahun 90-an dan awal 2000-an. Wadah ini hanya dapat menampung klip video dan audio dan juga dapat menyimpan beberapa trek dari setiap jenis, tetapi fitur ini jarang digunakan. Pemutaran AVI hampir universal, tetapi memiliki beberapa batasan kompresi yang menghasilkan ukuran file yang lebih besar dari rata-rata [3].

Terjadinya pembesaran ukuran file video disebabkan oleh durasi video yang terlalu lama dan juga dari ukuran Resolusi gambar yang dihasilkan, mengakibatkan banyak orang yang mengurangi resolusi pada video nya mengakibatkan kualitas video menjadi sangat buruk, Solusi yang tepat dari masalah terssebut adalah melakukan kompresi karna tujuan dari kompresi data adalah untuk merepresentasikan suatu data digital dengan sesedikit mungkin bit, dari permasalahan yang diatas akan diselesaikan dengan menggunakan *Algoritama Prefix Code* dan *Elias Omega code* dalam mengkompresi file video AVI. *Prefix Code* adalah jenis sistem kode yang dibedakan dengan memiliki "atribut awalan", yang mensyaratkan bahwa tidak ada seluruh kata kode dalam sistem yang menjadi awalan (awal 'segmen)' dari kata kode[4]. Selain di Sistem, ada empat kode pada awalan kode C1, C2, C3, C dan kode yang digunakan oleh penulis adalah kode C1. Untuk mendapatkan nilai kode C1, C2, C3 dan C sebelumnya, Anda perlu mengetahui nilai kode satuan, B(n) dan $\bar{B}$(n). Kode orde satu untuk bilangan positif n didefinisikan sebagai n - 1 diikuti oleh 0 atau sebagai alternatif bilangan bukan n - 1 diikuti dengan satu [5].

Algoritma Elias Omega code adalah algoritma kompresi yang diperkenalkan oleh Peter Elias pada tahun 1975. Ide utama dari kode ini adalah untuk awalan bilangan bulat yang dikodekan dengan representasi kode urutan besarnya dan menginisialisasi kode dalam bentuk rekursif. Algoritma Elias Omega Code mengurutkan karakter yang paling sering menjadi bit yang lebih kecil dan karakter yang paling langka menjadi bit yang lebih besar. Dengan cara ini, ukuran file dapat diperkecil dari ukuran aslinya [2][6].

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 260-270

Berdasarkan penelitian terdahulu yang dilakukan oleh Muhammad Alfarizi, Soeb Aripin pada tahun 2020 dengan judul "Penerapan *Algoritma Prefix Code* Dalam Kompresi file Video" dapat diambil kesimpulan bahwa kompresi file video dengan algoritma *Prefix code*, bahwa algoritma Prefix code cocok untuk kompresi file video. Hasil dari aplikasi kompresi video menunjukkan video yang telah diproses melalui kompresi dan dekompresi dan kemudian disimpan di driver komputer. Dengan adanya aplikasi ini diharapkan dapat memberikan dan meningkatkan keinginan orang lainnya untuk mempelajari kompresi data [5].

Berdasarkan penelitian terdahulu yang dilakukan oleh Febika Rada Kesuma pada tahun 2021 dengan judul "Implementasi Kombinasi *Algoritma Elias Omega Codes* Dan *Golomb Rice* Dengan Teknik *High Compress* Pada file Text" dapat diambil kesimpulan bahwa *algoritma Elias Omega Code* hanya mengurangi ukuran data sebesar 5-8% [7].

Berdasarkan penelitian terdahulu yang dilakukan oleh Nurul Rizka pada tahun 2021 dengan judul "Penerapan Algoritama Elias Omega Code Untuk Kompresi file Video Pada Aplikasi Rekam Layar" dari penelitian tersebut dapat diambil kesimupulan bahwa kompresi menggunakan algoritma elias omega code berhasil menangani kompresi file video karena perekaman layar dengan ekstensi. sehingga kompresi dapat berjalan pada teknik kompresi. Berdasarkan penerapan algoritma elias omega code, telah dibuktikan bahwa file video dengan ukuran besar dapat dikompresi ke ukuran yang lebih kecil dari [2].

2. METODOLOGI PENELITIAN

2.1 Kompresi *File* AVI

2.1.1 Kompresi

Kompresi data adalah upaya untuk mengurangi jumlah bit yang digunakan untuk menyimpan atau mengirimkan data. Kompresi data terdiri dari teknik kompresi berbeda yang diterapkan dalam bentuk perangkat lunak dan perangkat keras. Dari sudut pandang penggunaan, kompresi data dapat bersifat umum untuk semua tujuan atau khusus untuk beberapa tujuan. Keuntungan dari data terkompresi antara lain: mengurangi kemacetan selama I/O dan transfer data, menghemat ruang penyimpanan data tambahan, mempersulit pihak yang tidak berwenang untuk membaca data, dan memfasilitasi acara distribusi data dengan media yang dapat dipindahkan seperti flash disk, CD, DVD, dll [8][9][10].

2.1.2 File Video *Audio Video Interlaced* (AVI)

Audio Video Interlaced (AVI) pertama kali diperkenalkan pada tahun 1992 oleh Microsoft, video AVI adalah yang paling populer selama tahun 90-an dan awal 2000-an. Wadah ini hanya dapat menampung klip video dan audio dan juga dapat menyimpan beberapa trek dari setiap jenis, tetapi fitur ini jarang digunakan. Pemutaran AVI hampir universal, tetapi memiliki beberapa batasan kompresi yang menghasilkan ukuran file yang lebih besar dari rata-rata[11].

2.2 Algoritma *Prefix Code* dan *Elias Omega Code*

2.2.1 Algoritma *Prefix Code*

Prefix code adalah jenis sistem kode yang dibedakan dengan memiliki "*Prefix property*", yang mensyaratkan bahwa tidak ada seluruh kata kode dalam sistem yang menjadi awalan (awal 'segmen') dari kata kode. Selain di Sistem, ada empat kode pada awalan kode C1, C2, C3, C4 dan kode yang digunakan oleh penulis adalah kode C1. Untuk mendapatkan nilai kode C1, C2, C3 dan C4 sebelumnya, Anda perlu mengetahui nilai kode satuan, B(n) dan $\bar{B}$(n). Kode orde satu untuk bilangan positif n didefinisikan sebagai n - 1 diikuti oleh 0 atau sebagai alternatif bilangan bukan n - 1 diikuti oleh a. B(n) untuk menampilkan representasi biner dari bilangan bulat n, sedangkan $\bar{B}$(n) untuk menampilkan nilai B(n) tanpa bit dengan menghilangkan nilai 1 bit pertama dari setiap bilangan B(n). Setelah mencari nilai B (n) dan $\bar{B}$ (n), mencari nilai kode C1 dengan contoh n = 5 = 1012, ukuran B (5) adalah 3, kemudian mulai dengan kode unari 110 (atau 001) dan tambahkan $\bar{B}$ (5) = 01, sehingga kode lengkapnya adalah 110 | 01 (atau 001 | 10) Untuk mencari nilai kode C2, C3, dan C4 dapat merujuk pada tabel 1 di bawah ini [8][12].

Tabel 1. Tabel ketentuan *Prefix code*

N	Unary	B(n)	$\bar{B}$(n)	C1	C2	C3	C4
1	0	1		0|	0	0|	0
2	10	10	1	10|0	100	100|0	10|0
3	110	11	00	10|1	110	100|1	11|0
4	1110	100	01	110|00	10100	110|00	10|100|0
5	11110	101	10	110|01	10110	110|01	10|101|0
6	111110	110	11	110|10	11100	110|10	10|110|0
7	1111110	111	000	110|11	11110	110|11	10|111|0
8	`	1000	001	1110|000	1010100	10100|000	11|1000|0
9		1001	010	1110|001	1010110	10100|001	11|1001|0

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 260-270

N	Unary	B(n)	$\bar{B}$(n)	C1	C2	C3	C4
10		1010	011	1110|010	1011100	10100|010	11|1010|0
11		1011	100	1110|011	1011110	10100|011	11|1011|0
12		1100	101	1110|100	1110100	10100|100	11|1100|0
13		1101	110	1110|101	1110110	10100|101	11|1101|0

2.2.2 Algoritma *Elias Omega Code*

Algoritma *Elias Omega code* adalah algoritma kompresi yang diperkenalkan oleh Peter Elias pada tahun 1975. Peter Elias menjelaskan tiga kode awalan yang berguna. Ide utama dari kode ini adalah untuk mengawali bilangan bulat yang disandikan dengan representasi ordinal yang disandikan sebagai yang terbesar. Menentukan panjang M diselesaikan dengan berbagai cara dengan berbagai kode Elias. Algoritma itu sendiri menggunakan rekursi untuk mengkodekan awalan M, itulah sebabnya kadang-kadang disebut kode Elias rekursif [7].

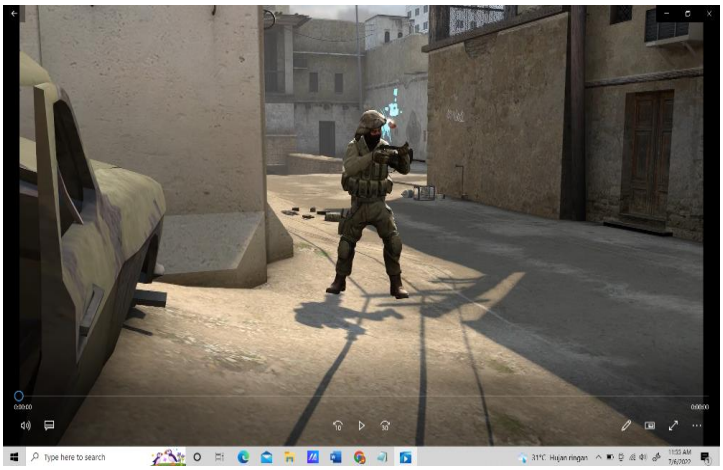
Tabel 2. Kode *Elias Omega Code*

Nilai n	Kode *Elias Omega Code*
1	0
2	10 0
3	11 0
4	10 100 0
5	10 101 0
6	10 110 0
7	10 111 0
8	11 1000 0
9	11 1001 0
10	11 1010 0
11	11 1010 0
12	11 1100 0
13	11 1101 0
14	11 1110 0
15	11 1111 0
16	10 100 10000 0

3. HASIL DAN PEMBAHASAN

3.1 Analisa Masalah

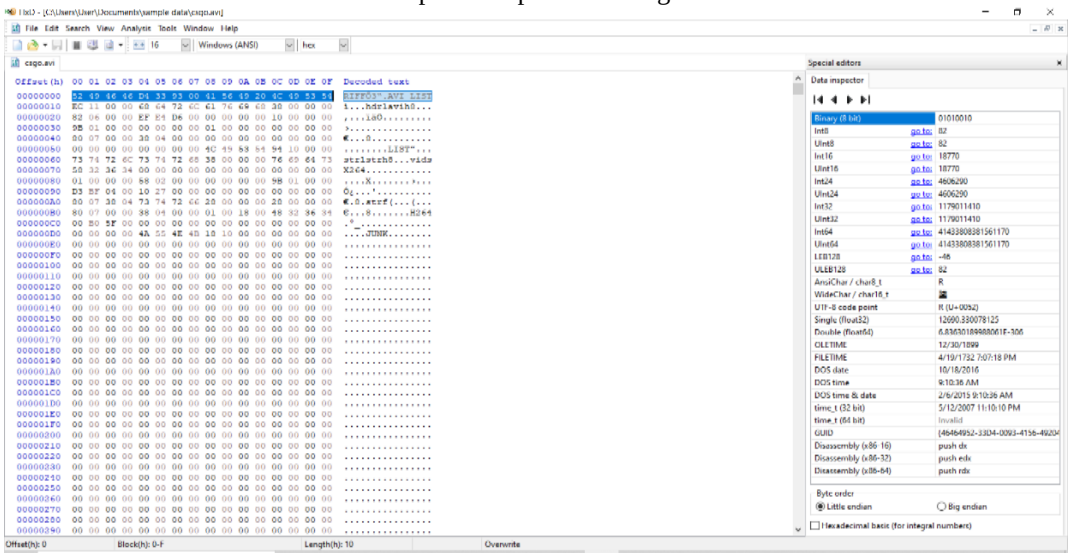
Berdasarkan analisis, kompresi file video dengan ekstensi AVI sangat besar. Dengan mengompresi file AVI, file ukuran yang lebih besar akan dikompresi ke ukuran yang lebih kecil dan menghemat ruang penyimpanan dan algoritma yang lebih tepat dikenal untuk mengompresi file AVI dengan menggunakan algoritma Perfix Code. Dalam penelitian ini akan dibahas dua proses utama, yaitu prosedur kompresi dan proses dekompresi, dan peneliti melakukan kompresi dengan menggunakan dua algoritma yaitu algoritma Prefix Code dengan Elias Omega Code, Pada sistem kompresi file video, terlebih dulu dilakukan pembacaan file MP4 agar didapatkan nilai hexadesimal menggunakan aplikasi HxD. Berikut contoh file MP4 yang dikompres serta didekompres dapat dilihat pada gambar dibawah ini.



Gambar 1. File Video AVI

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 260-270

Berdasarkan gambar dengan nama file Csgo.avi dan memiliki ukuran 9.19 MB terdapat nilai biner viewer, adapun nilai hexadecimal dari contoh file video pada sampel Video Csgo adalah :



Gambar 2. Kode Hexadecimal file video dari hasil Binary Viewer

Tabel 3. Tampilan Nilai *Hexal* Berdasarkan Biner *Viewer*

52	49	46	46	D4	33	93	00
41	56	49	20	4C	49	53	54

Tabel 4. Pendataan Simbol

No	Hexadecimal	Frekuensi
1	49	3
2	46	2
3	D4	1
4	33	1
5	93	1
6	00	1
7	41	1
8	56	1
9	20	1
10	4C	1
11	53	1
12	54	1
13	52	1
	Jumlah	16

Berdasarkan tabel di atas, kita mendapatkan jumlah nilai biner yang sama. Sebelum melakukan kompresi file video, langkah pertama adalah membaca nilai Binner kemudian membuat array nilai Hexa yang diurutkan dari nilai frekuensi terbesar (nilai Binner sama) hingga terkecil. Urutan nilai Hex dapat dilihat pada tabel di bawah ini:

Tabel 5. Hexa Yang Belum Dikompresi

No	Hexadecimal	Biner	Frekuensi-	Bit	Frekuensi x Bit
1	49	01001001	3	8	24
2	46	01000110	2	8	16
3	D4	11010100	1	8	8
4	33	00110011	1	8	8
5	93	10010011	1	8	8
6	00	00000000	1	8	8
7	41	01000001	1	8	8
8	56	01010110	1	8	8

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 260-270

No	Hexadecimal	Biner	Frekuensi-	Bit	Frekuensi x Bit
9	20	00100000	1	8	8
10	4C	01001100	1	8	8
11	53	01010011	1	8	8
12	54	01010100	1	8	8
13	52	01010010	1	8	8
		Total BIT			128

3.1.1 Penerapan Algoritma *Prefix Code*

Menurut tabel di atas, nilai desimal (karakter) sama dengan 8 bit bilangan biner. Jadi, 16 bilangan desimal memiliki nilai biner 128 bit. Untuk mengkonversi satuan ke byte, jumlah total bit dibagi 8, sehingga dihasilkan 128/8 = 16 byte. Langkah selanjutnya adalah membentuk tabel pengkodean C1 dalam algoritma Prefix Code. Kode C1 dapat dilihat pada Tabel 3.5 di bawah ini:

Tabel 6. Kode C1 Pada Algoritma *Prefix Code*

n	Unary	B(n)	$\bar{B}$(n)	C1
1	0	1		0|
2	10	10	0	10|0
3	110	11	1	10|1
4	1110	100	00	110|00
5	11110	101	01	110|01
6	111110	110	10	110|10
7	1111110	111	11	110|11
8	11111110	1000	000	110|000
9	111111110	1001	001	110|001
10	1111111110	1010	010	110|010
11	11111111110	1011	011	110|011
12	111111111110	1100	100	110|100
13	1111111111110	1101	101	110|101

Berdasarkan pengkodean C1 pada algoritma pengkodean awalan di atas, hanya menampilkan kode hingga nilai karakter 7. Cara mendapatkan kode C1, jika jumlah karakter yang dihasilkan lebih besar dari 7 karakter. misalnya, pada n=17 dengan nilai biner 17=100012 Jadi ukuran B (17) adalah 5, jadi kita mulai dengan kode satuan 11110 (atau 00001) dan kita tambahkan nilai B(n). maka untuk mendapatkan nilai B(n), biner 1 harus dihilangkan, maka hasil dari B (17) = 0001, sehingga kode lengkap yang didapat adalah 11110 | 0001 (atau 00001 | 0001. Prosedur berikut adalah mengompresi nilai sampel hex dengan nilai kode C1 yang diperoleh dari Tabel 4.5 di atas. Anda dapat melihat contoh prosedur kompresi gambar pada tabel di bawah ini:

3.1.2 Membentuk Tabel *Prefix Code*

Tabel 7. Nilai *Hexal* Yang Sudah Dikompresi Dengan Kode C1 Pada Algoritma *Prefix Code*

n	Nilai Hexa	Frex	Codeword (C1)	Bit	Frek x Bit
1	49	3	0	1	3
2	46	2	100	3	6
3	D4	1	101	3	3
4	33	1	11000	5	5
5	93	1	11001	5	5
6	00	1	11010	5	5
7	41	1	11011	5	5
8	56	1	110000	6	6
9	20	1	110001	6	6
10	4C	1	110010	6	6
11	53	1	110011	6	6
12	54	1	110100	6	6
13	52	1	110101	6	6
		Total			62

Dapat dibentuk nilai bit baru hasil kompresi dari susunan nilai hexadecimal sampel awal sebelum kompresi yaitu, 52, 49, 46, 46, D4, 33, 93, 00, 41, 56, 49, 20, 4C, 49, 53, 54, (tanpa tanda koma dan spasi) menjadi nilai bit biner.

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 260-270

10100100100100101000110010001101101010000110011100100110000000000100000101010110010010010010000001001100010010010010100110101010100

Kemudian sebelum didapatkan hasil keseluruhan akhir kompresi dilakukan penambahan stringbit itu sendiri yaitu padding bit dan flag bit. Jika sisa bagi panjang string bit terhadap 8 adalah 0 maka tambahkan 00000001. Nyatakan dengan bit akhir. Sedangkan jika sisa bagi panjang string bit terhadap 8 adalah n(1,2,3,4,5,6,7) maka tambahkan 0 sebnyak 7 – n + "1" di akhir string bit. Nyatakan dengan L. Lalu tambahkan bilangan biner dari 9 – n. nyatakan dengan bit akhir. karena jumlah string bit 62 tidak habis dibagi delapan dan sisanya 6 bit, nyatakan sisa bagi tersebut dengan nilai n. maka tambahkan 0 sebanyak 7 – n + "1" di akhir string bit. Nyatakan dengan L. Lalu tambahkan bilangan biner dari 9 – n. Nyatakan dengan bit akhir.

7 – n + "1"
7 – 6 + "1" =**01**
Bit Akhir 9 – n
Bit Akhir = 9 – 6 = 3 = **00000011**

Gambar 3. Perhitungan penambahan bit

10100100100100101000110010001101101010000110011100100
10000000001000001010101100100100100100000010011000100
10010100110101010**0100000011**

Gambar 4. *String Bit* yang telah dilakukan penambahan

Total panjang bit keseluruhan setelah ada penambahan bit adalah 62+ 10 = 72 bit. Selanjutnya lakukan pemisahan bit manjadi beberapa kelompok. Setiap kelompok terdiri dari 8 bit seperti gambar di bawah ini.

10100100 10010010 10001100 10001101 10101000 01100111 00100110
00000000 10000010 10101100 10010010 01000000 10011000 10010010
10100110 10101000 10000001 1

Gambar 5. Pembagian *String Bit*

Berdasarkan pada pembagian kelompok nilai biner, didapatkan 18 kelompok nilai biner baru (18 byte) yang telah dikompresi nilai biner penambahan bit. Lalu dilakukan pembagian, maka nilai yang telah dibagikan diubah dalam suatu karakter terlebih dahulu dengan cara dilakukannya pencarian nilai hexadecimal serta string bit ini memakai kode ASCII untuk mengetahui nilai yang sudah dikompresi.

10100100	10010010	10001100	10001101	10101000	01100111
¤	□	□	□	¨	g
00100110	00000000	10000010	10101100	10010010	01000000
&	0	□	¬	□	@
10011000	10010010	10100110	10101000	10000001	1
1	%	M	Q	□	

Gambar 6. karakter String bit

Tabel 8. Nilai Hexadecimal Terkompresi

Urutan Nilai Terkompresi	Nilai Hexadecimal Terkompresi
1	A4
2	92
3	8C

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 260-270

Urutan Nilai Terkompresi	Nilai Hexadecimal Terkompresi
4	8D
5	A8
6	67
7	26
8	0
9	82
10	AC
11	92
12	40
13	98
14	92
15	A6
16	A8
17	81
18	1

Persentase ukuran data yang dikompresi dapat dilihat pada hasil perbandingan antara ukuran data sesudah dikompresi menggunakan ukuran data sebelum dikompresi.
Ukuran data sebelum dikompresi = 128
Ukuran data sesudah dikompresi = 72
Dari data ini dapat dilakukan perhitungan kinerja kompresinya yaitu :

1. *Ratio Compression (Rc)*
   $Rc = \frac{Ukuran\ data\ sebelum\ dikompresi}{Ukuran\ data\ sesuda\ dikompresi}$
   $Rc = \frac{128}{72}$
   Rc = 1,77
2. *Compression Ratio (Cr)*
   $Cr = \frac{Ukuran\ data\ sesudah\ dikompresi}{ukuran\ data\ sebelum\ dikompresi}$ x100%
   $Cr = \frac{72}{128}$ x100%
   Cr = 56,25%
3. *Space Saving*
   SS -> 100% - Ukuran sesudah dikompresi
   SS = 100% - 56,25%
   SS = 43,75%

Maka berdasarkan perhitungan diatas hasil setelah dikompresi 56,25%

3.1.3 Penerapan Algoritma *Alias Omega Code*

Adapun bilangan hexadecimal dari file video tersebut adalah 52, 49, 46, 46, D4, 33, 93, 00, 41, 56, 49, 20, 4C, 49, 53, 54. Nilai data ini dimasukan kedalam tabel untuk dilakukan pembacaan frekuensi. Pembacaan frekuensi dilakukan dengan menghitung jumlah nilai yang sama di setiap nilai data yang muncul. Adapun pembacaan frekuensi dapat dilihat pada tabel di bawah ini :

Table 9. Pendataan Simbol

No	Hexadecimal	Frekuensi
1	49	3
2	46	2
3	D4	1
4	33	1
5	93	1
6	00	1
7	41	1
8	56	1
9	20	1
10	4C	1
11	53	1
12	54	1
13	52	1

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 260-270

Jumlah	16

Mengurutkan dari karakter yang memiliki frekuensi terbesar (banyak nilai yang sama) ke frekuensi terkecil. Urutan nilai dapat dilihat pada tabel berikut ini:

Tabel 10. Nilai Bit *File* Video Sample

No	Hexadecimal	Biner	Bit	Frekuensi	Frekuensi x Bit
1	49	01001001	8	3	24
2	46	01000110	8	2	16
3	D4	11010100	8	1	8
4	33	00110011	8	1	8
5	93	10010011	8	1	8
6	00	00000000	8	1	8
7	41	01000001	8	1	8
8	56	01010110	8	1	8
9	20	00100000	8	1	8
10	4C	01001100	8	1	8
11	53	01010011	8	1	8
12	54	01010100	8	1	8
13	52	01010010	8	1	8
		Total bit			128 bit

Berdasarkan tabel diatas, satu nilai hexadecimal (karakter) bernilai 8 bit bilangan biner. Sehingga 16 bilangan hexadecimal mempunyai nilai biner sebanyak 128 bit. Untuk mengubah satuan menjadi byte maka jumlah keseluruhan bit dibagikan 8. Maka dihasilkan 128/8 = 16 byte.

3.1.4 Membentuk Tabel *Elias Omega Code*

Aturan dalam pembentukan kode bilangan dengan menggunakan elias omega code dapat dilihat pada sub landasan teori bab sebelumnya. Adapun kode elias omega code dapat dilihat pada tabel 4.12

Tabel 11. Kode Elias Omega

N	Kode Elias Omega
1	0
2	10 0
3	11 0
4	10 100 0
5	10 101 0
6	10 110 0
7	10 111 0
8	11 1000 0
9	11 1001 0
10	11 1010 0
11	11 1011 0
12	11 1100 0
13	11 1101 0
14	11 1110 0
15	11 1111 0
16	10 100 10000 0
17	10 100 10001 0
18	10 100 10010 0

Berdasarkan pada tabel kode elias omega di atas hanya menunjukan kode sampai nilai karakter ke 18. Bagaimana proses yang dilakukan untuk mendapat kode Elias omega, jika jumlah karakter yang dihasilkan lebih dari 18 karakter. Misalnya pada nilai = 19, nilai desimal 19 dikodekan dengan cara (1) berikan angka 0, (2) tambahkan dengan representasi biner dari 19, yaitu 10011(2) (terdiri dari 5 bit), (3) tambahkan dengan nilai 3-bit dari 5-1 = 1002, dan (4) tambahkan nilai 2-bit dari 3-1 = 102. Hasilnya adalah 10|100|10011|0. Proses selanjutnya adalah melakukan kompresi nilai dari sampel dengan nilai kode elias omega yang di dapat dari tabel 4.12. di atas. Adapun proses kompresi file video sampel dapat dilihat pada tabel berikut :

Tabel 12. Kompresi Nilai *File* Video Sample Dengan *Elias Omega Code*

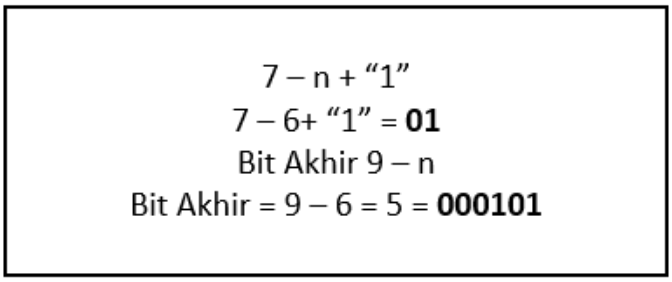
N	Hexadecimal	Kode Elias Omega	Bit	Frekuensi	Bit x frekuensi
1	49	0	1	3	3

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 260-270

N	Hexadecimal	Kode Elias Omega	Bit	Frekuensi	Bit x frekuensi
2	46	10 0	3	2	6
3	D4	11 0	3	1	3
4	33	10 100 0	6	1	6
5	93	10 101 0	6	1	6
6	00	10 110 0	6	1	6
7	41	10 111 0	6	1	6
8	56	11 1000 0	7	1	7
9	20	11 1001 0	7	1	7
10	4C	11 1010 0	7	1	7
11	53	11 1011 0	7	1	7
12	54	11 1100 0	7	1	7
13	52	11 1101 0	7	1	7
Total bit					78

Dari perhitungan tabel diatas setelah dikompresi dengan menggunakan elias omega code adalah 78 bit. Untuk diubah menjadi satuan byte maka dibagi 8 yaitu 78/8 = 9.75 byte. Tahap salanjutnya melakukan hasil string bit elias omega code menjadi nilai file. Sebelum melakukan hasil string bit elias omega code menjadi nilai file, terlebih dahulu dilakukan pemeriksaan terhadap panjang string bit. Pembentukan nilai bit baru hasil kompresi dari susunan nilai hexadecimal sebelum dikompresi yaitu 52, 49, 46, 46, D4, 33, 93, 00, 41, 56, 49, 20, 4C, 49, 53, 54,

"10100100100100101000110010001101101010000110011100100110000000001000001010101100100100100000001001100010010010101001101010100010000001" Kemudian sebelum di dapatkan hasil keseluruhan akhir kompresi dilakukan penambahan string bit itu sendiri yaitu padding bit dan flag bit. Jika sisa bagi panjang string bit terhadap 8 adalah 0 maka tambahan 00000001. Nyatakan dengan bit akhir. Sedangkan jika sisa bagi panjang string bit terhadap 8 adalah n(1,2,3,4,5,6,7) maka tambahkan 0 sebnyak 7 – n + "1" di akhir string bit. Nyatakan dengan L. Lalu tambahkan bilangan biner dari 9 – n. nyatakan dengan bit akhir. Karena jumlah string bit 78 tidak habis dibagi 8 dan sisanya 6 bit, nyatakan sisa bagi tersebut dengan nilai n. maka tambahkan 0 sebanyak 0 sebanyak 7 – n + "1" di akhir string bit. Nyatakan dengan L. Lalu tambahkan bilangan biner dari 9 – n. Nyatakan dengan bit akhir.



Gambar 7. Perhitungan penambahan bit

10100100100100101000110010001101101010000110011100100110000000001000001010101100100100100000001001100010010010101001101010100010000001**01000101**

Total panjang bit keseluruhan setelah ada penambahan bit adalah 78+6+8= 92. Selanjutnya lakukan pemisahan bit manjadi beberapa kelompok. Setiap kelompok terdiri dari 8 bit seperti gambar di bawah ini

10100100 10010010 10001100 10001101 10101000 01100111 00100110 00000000 10000010 10101100 10010010 01000000 10011000 10010010 10100110 10101000 10000001 1**0100010 1**

Berdasarkan pada pembagian kelompok nilai biner, didapatkan 13 kelompok nilai biner baru (13 byte) yang sudah terkompresi beserta nilai biner penambahan bit. Setelah pembagian dilakukan, maka nilai yang sudah dibagi iubah kedalam suatu karakter dengan terlebih dahulu mencari nilai hexadecimal dari string bit tersebut menggunakan kode ASCII untuk mengetahui nilai yang sudah terkompresi. Adapun nilai yang sudah terkompresidapat dilihat pada tabel di bawah ini.

Tabel 13. Tabel Nilai Hexadecimal Terkompresi

Urutan Nilai Terkompresi	Nilai Hexadecimal Terkompresi
1	A4
2	92
3	8C
4	8D
5	A8
6	67

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 260-270

Urutan Nilai Terkompresi	Nilai Hexadecimal Terkompresi
7	26
8	0
9	82
10	AC
11	92
12	40
13	98
14	92
15	A6
16	A8
17	81
18	A2
19	1

Persentase ukuran data yang telah dikompresi dan didapat dari hasil perbandingan antara ukuran data setelah dikompresi dengan ukuran data sebelum dikompresi.
Ukuran data sebelum dikompresi = 128/8 =16 byte
Ukuran data sesudah dikompresi = 92/8 = 11 byte
Berdasarkan data tersebut dapat dihitung kinerja kompresinya yaitu :

1. *Ratio Compression (Rc)*
   $Rc = \frac{Ukuran\ data\ sebelum\ dikompresi}{Ukuran\ data\ sesuda\ dikompresi}$
   $Rc = \frac{16}{11}$
   $Rc = 1.45$
2. *Compression Ratio (Cr)*
   $Cr = \frac{Ukuran\ data\ sesudah\ dikompresi}{ukuran\ data\ sebelum\ dikompresi}$ x100%
   $Cr = \frac{11}{16}$ x100%
   $Cr = 68{,}75\%$
3. *Space Saving*
   SS -> 100% - Ukuran sesudah dikompresi
   SS = 100% - 68,75%
   SS = 31,25%

Maka berdasarkan perhitungan diatas hasil setelah dikompresi 68,75%

3.1.5 Perbandingan Algoritma *Prefix Code* dan *Elias Omega Code* Menggunakan metode Eksponensial

Perbandingan algortima Prefix Code dan Elias Omega Code ialah membandingan Ratio Compression (Rc), Compression Ratio (Cr), redudancy dan space saving dari kedua algoritma. Dimana setelah dilakukan perhitungan dengan algoritma Prefix Code dan Elias Omega Code maka hasil dari Ratio Compression (Rc), Compression Ratio (Cr), Redudancy dan space saving didapatkan lalu akan menggunakan algoritma eksponensial untuk memandingkan kedua algoritma tersebut. untuk perbandingannya dapat dilihat pada tabel dibawah ini.

Tabel 14. Nilai Perbandingan

Kriteria	Bobot	Algoritma Prfix Code	Algoritma Elias Omega Code
RC	0,20	1,77	1.45
CR	0,30	56,25	68,75
RD	0,15	43,75	31.25
SS	0,35	43,75	31,25
Total nilai $\sum (N)^B$	1	9,983	9,644

Langkah – langkah perhitungan sebagai berikut :

1. Nilai Algoritma Prefix Code :
   $=(1{,}77)^{0{,}20} + (56{,}25)^{0{,}30} + (43{,}75)^{0{,}15} + (43{,}75)^{0{,}35}$
   =1,120 + 3,349 + 1,762 + 3,752
   =9,983
2. Nilai Algoritma Elias Omega Code :
   $=(1{,}45)^{0{,}20} + (68{,}75)^{0{,}30} + (31{,}25)^{0{,}15} + (31{,}25)^{0{,}35}$

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 260-270

=1,077 + 3,557 + 1,675 + 3,335
9,664

Berdasarkan penjelasan pada tabel diatas, maka Algoritma Elias Omega code yang menjadi algoritma yang efektif dalam melakukan kompresi file video AVI, hal ini dikarenakan semakin kecil total nilai yang diperoleh maka semakin dikit jumlah usaha yang dilakukan oleh algoritma tersebut dalam melakukan kompresi.

4. KESIMPULAN

Kesimpulan yang dapat ditarik setelah dilakukan pengerjaan pada bab-bab sebelumnya yaitu Algoritma Prefix Code dan juga algoritma Elias Omega Code adalah algoritma yang sangat sesuai untuk melakukan kompresi file video. Dalam menganalisis kedua algoritma ini, nilai perbandingan dihitung menggunakan metode Eksponensial. Proses kompresi file video AVI berhasil dilakukan dengan menggunakan dua algoritma yaitu Prefix Code dan juga Elias Omega Code

REFERENCES

[1] T. P. Sari, S. D. Nasution, and R. K. Hondro, "Penerapan Algoritma Levenstein Pada Aplikasi Kompresi File Mp3," *KOMIK (Konferensi Nas. Teknol. Inf. dan Komputer)*, vol. 2, no. 1, pp. 439–443, 2018, doi: 10.30865/komik.v2i1.946.
[2] N. Rizka, S. D. Nasution, and K. Ulfa, "Penerapan Algoritma Elias Omega Code Untuk Kompresi File Video Pada Aplikasi Rekam Layar," vol. 9, no. 4, pp. 257–265, 2021.
[3] A. Setiawan, "Veteksi Gerak Satu Obyek Pada Vidio AVI," *Veteksi Gerak Satu Obyek Pada Vidio AVI*, 2011.
[4] E. N. Simanjuntak, "Penerapan Algoritma Prefix Code Pada Kompresi File Gambar," *BEES Bull. Electr. Electron. Eng.*, vol. 1, no. 3, pp. 96–100, 2021.
[5] S. A. Muhammad Alfarizi, "Penarapan Algoritma Prefix Code Dalam Kompresi File Video," *KOMIK (Konferensi Nas. Teknol. Inf. dan Komputer)*, vol. 4, no. 1, pp. 2597–4645, 2020.
[6] I. Hasan, T. Tommy, and N. I. Syahputri, "Analisis Parameter Kompresi Algoritma Elias Omega Code dan Fibonacci Code Pada File Digital," *Algoritm. J. ILMU Komput. DAN Inform.*, vol. 5, no. 1, 2021.
[7] F. R. Kesuma, "Implementasi Kombinasi Algoritma Elias Omega Codes Dan Golomb Rice Dengan Teknik High Compress Pada File Teks," *Inf. dan Teknol. Ilm.*, vol. 8, no. 2, pp. 76–81, 2021.
[8] M. Alfarizi and S. Aripin, "Penarapan Algoritma Prefix Code Dalam Kompresi File Video," *KOMIK (Konferensi Nas. …*, vol. 4, pp. 249–252, 2020, doi: 10.30865/komik.v4i1.2686.
[9] B. W. Transform and A. S. Harahap, "Analisis Dan Implementasi Kompresi File Citra Menggunakan Algoritma," vol. 1, no. 1, pp. 6–12, 2021.
[10] E. Prayoga and K. M. Suryaningrum, "IMPLEMENTASI ALGORITMA HUFFMAN DAN RUN LENGTH ENCODING PADA APLIKASI KOMPRESI BERBASIS WEB," *J. Ilm. Teknol. Infomasi Terap.*, vol. 4, no. 2, pp. 92–101, 2018, doi: 10.33197/jitter.vol4.iss2.2018.154.
[11] S. Yunitasari and M. F. Sabilillah, "Pengaruh Penerapan Animated Video (AVI) and Bulk Toothbrush (Booth) terhadap Pengetahuan, Sikap dan Perilaku Kesehatan Gigi pada Siswa Kelas 4," *J. Ilm. Keperawatan Gigi*, vol. 3, no. 2, pp. 332–343, 2022.
[12] E. N. Simanjuntak, "Penerapan Algoritma Prefix Code Pada Kompresi File Gambar," *BEES Bull. Electr. Electron. Eng.*, vol. 1, no. 3, pp. 96–100, 2021.