Pendas : Jurnal Ilmiah Pendidikan Dasar,
ISSN Cetak : 2477-2143 ISSN Online : 2548-6950
Volume 09 Nomor 04, Desember 2024

ANALISA PERBANDINGAN ALGORITMA PREFIX CODE DENGAN ALGORITMA BURROWS-WHEELER TRANSFORM DALAM KOMPRESI FILE VIDEO

Putri Delfi Hardianti[1], Imam Saputra[2], Soeb Aripin[3]
[1,2,3]Universitas Budi Darma, Medan
[1]putridiianti17@gmail.com, [2]saputraimam69@gmail.com,
[3]suefarifin@gmail.com

ABSTRAK

File video adalah jenis *file* yang direkam atau disimpan dalam format digital yang berisi data visual dan audio, termasuk gambar bergerak, suara, dan teks. Ukuran *file* video yang besar sering kali menyebabkan masalah dalam penyimpanan dan pengiriman data, terutama untuk video dengan durasi panjang. Media penyimpanan seperti g-drive dan *cloud storage* digunakan untuk mengatasi kebutuhan ruang penyimpanan, namun solusi ini sering kali tidak cukup efisien. Oleh karena itu, diperlukan teknik kompresi data untuk memperkecil ukuran *file* tanpa kehilangan informasi penting. Untuk mengatasi masalah ini, penelitian ini mengimplementasikan dua algoritma kompresi data, yaitu *Prefix Code* dan *Burrows-Wheeler Transform*. Algoritma *Prefix Code* menggunakan metode pengkodean biner yang unik untuk setiap simbol dalam data, sedangkan *Burrows-Wheeler Transform* melakukan transformasi data teks untuk menghasilkan pola repetitif yang lebih mudah dikompresi. Tujuan dari penelitian ini adalah membandingkan efektivitas kedua algoritma tersebut dalam mengkompresi *file* video, dengan fokus pada parameter *Ratio of Compression* (RC), *Compression Ratio* (CR), dan *Space Saving* (SS). Hasil penelitian menunjukkan bahwa kedua algoritma mampu mengompresi *file* video secara efektif. Namun, perbandingan antara algoritma menunjukkan perbedaan signifikan dalam performa kompresi. Algoritma *Prefix Code* terbukti lebih efisien dalam mengurangi ukuran *file* tanpa mengorbankan kualitas data, sementara algoritma *Burrows-Wheeler Transform* menunjukkan keunggulan dalam mempertahankan integritas data selama proses transformasi. Analisis ini memberikan wawasan yang lebih dalam mengenai efektivitas kedua algoritma dan dapat membantu dalam memilih teknik kompresi yang paling sesuai untuk *file* video.

Kata Kunci : *file video*, *prefix code, burrows-wheeler transform*

ABSTRACT

A video file is a type of file recorded or stored in a digital format that contains visual and audio data, including moving images, sound, and text. The large size of video files often causes issues in storage and data transmission, especially for long-duration videos. Storage media such as Google Drive and cloud storage are used to address storage space needs, but these solutions are often not efficient enough. Therefore, data compression techniques are required to reduce file size without losing important information. To address this issue, this research implements two data compression algorithms: Prefix Code and Burrows-Wheeler Transform. The Prefix Code algorithm uses a unique binary encoding method for each symbol in the data, while the Burrows-Wheeler Transform performs a text data transformation to produce repetitive patterns that are easier to compress. The aim of this research is to compare the effectiveness of these two algorithms in compressing video files, focusing on the parameters of Compression Ratio (CR), Ratio of Compression (RC), and Space Saving (SS). The results indicate that both

algorithms are effective in compressing video files. However, a comparison between the algorithms shows significant differences in compression performance. The Prefix Code algorithm proves to be more efficient in reducing file size without compromising data quality, while the Burrows-Wheeler Transform algorithm shows advantages in maintaining data integrity during the transformation process. This analysis provides deeper insights into the effectiveness of both algorithms and can assist in choosing the most appropriate compression technique for video files.

Keywords: video file, prefix code, burrows-wheeler transform

A. Pendahuluan

File video adalah jenis *file* yang direkam atau disimpan dalam format digital yang berisi data visual dan audio, termasuk gambar bergerak, suara, dan teks. Secara umum, teknologi yang berformat *file* video mengandung informasi yang sangat penting dan sangat disukai oleh banyak orang (Huda 2020);(Afriyadi 2023). File video umumnya memiliki ukuran yang besar, terutama jika durasinya panjang, sehingga membutuhkan ruang penyimpanan yang cukup dan dapat memakan waktu lama untuk proses pengiriman data. Untuk menyimpan data atau file tersebut, diperlukan media penyimpanan yang memadai seperti *Google Drive* atau layanan cloud storage lainnya (Ginting 2021). Untuk mengatasi permasalahan yang terjadi dalam penyimpanan *file* video yang memiliki ukuran besar, maka diperlukan sebuah teknik yang mampu memperkecil ukuran data untuk penyimpanan yaitu Kompresi Data (*Data Compression*).

Kompresi data merupakan proses untuk mengurangi jumlah bit yang diperlukan untuk menyimpan atau mentransmisikan sebuah video, dengan tujuan meminimalkan ukuran file tanpa menghilangkan informasi penting (Saragih and Utomo 2020);(Sodikin, Putri, and Hidayat 2022). Teknik ini juga berfungsi untuk meningkatkan efisiensi penyimpanan dan mempercepat pertukaran data (Utari et al. 2016);(Sepandi 2020). Dengan menghilangkan redundansi atau informasi yang tidak perlu, kompresi data membantu dalam mengoptimalkan penggunaan ruang penyimpanan dan mempercepat proses transfer data. Berbagai algoritma digunakan dalam kompresi file video, termasuk Prefix Code, Run Length Encoding, Lempel-Ziv Compression, Burrows-Wheeler Transform, dan lainnya, masing-masing dengan metode dan tingkat efektivitas yang berbeda dalam

mengurangi ukuran file (Sihotang 2022);(Alqori 2024).

Pada penelitian ini, penulis menggunakan dua metode untuk proses kompresi *file* video yaitu, algoritma *Prefix Code* dan algoritma *Burrows-Wheeler Transform*. Namun, dari kedua metode ini akan penulis lakukan melihat hasil metode yang terbaik. Sehingga, penulis akan melakukan perbandingan algoritma yang digunakan untuk kompresi *file* video yaitu, algoritma *Prefix Code* dengan algoritma *Burrows-Wheeler Transform*. Algoritma *Prefix Code* adalah suatu metode pengkodean yang digunakan dalam teori informasi untuk menghasilkan kode biner dimana tidak ada kata kode yang merupakan awalan (*prefix*) dari kata kode lainnya (Hartama 2022). Dalam algoritma ini, setiap simbol atau karakter dalam data diberikan kode biner yang tidak menjadi awalan dari kode biner yang lain (Saragih and Utomo 2020). Algoritma *Burrows Wheeler Transform* (BWT) adalah suatu teknik transformasi pada sebuah string atau teks yang dapat menghasilkan representasi teks yang memiliki pola repetitif yang lebih mudah dikompresi (Cox et al. 2012). Algoritma *Burrows Wheeler Transform* mengurutkan dan mengubah blok data teks menjadi format baru yang memiliki karakter yang sama. Dalam makalah penelitian mereka pada tahun 1994 yang berjudul "*A Block Sorting Lossless Data Compression Algorithm*", *Burrows* dan *Wheeler* menunjukkan algoritma kompresi data yang berbasis pengurutan (Ramadhan 2023). Setelah menerapkan kedua algoritma untuk kompresi, penulis membandingkan hasilnya untuk mengevaluasi mana yang lebih efektif dalam mengurangi ukuran file video. Tujuan dari perbandingan ini adalah untuk memilih algoritma yang paling efisien dan sesuai untuk digunakan dalam proses kompresi file video.

Penelitian yang dilakukan oleh Muhammad Alfarizi dan Soeb Aripin pada tahun 2020 berjudul "Penerapan Algoritma Prefix Code Dalam Kompresi File Video" menunjukkan bahwa Algoritma Prefix Code efektif untuk mengompresi file video dan dapat menghemat memori penyimpanan (Alfarizi and Aripin 2020). Penelitian oleh Muhammad Rizky Ramadhan pada tahun 2023 yang berjudul "Analisa Perbandingan *Algoritma Run Length Encoding*

Dengan Burrows Wheeler Transform Dalam Kompresi File Video" menunjukkan bahwa kedua algoritma tersebut, yaitu Run Length Encoding dan Burrows Wheeler Transform, termasuk kategori terbaik dalam mengompresi file video. Penelitian ini menemukan bahwa hasil kompresi tidak mengurangi isi file, dengan algoritma BWT hanya mentransformasikan data tanpa mengurangi kontennya (Ramadhan 2023). Penelitian oleh Boy Alfredo Silaban pada tahun 2022 berjudul "Analisa Kompresi File Teks Dengan Kombinasi Metode *Burrows Wheeler Transform* dan Shannon Fano" menyimpulkan bahwa kombinasi kedua algoritma tersebut sangat efisien dalam mengompresi file teks. Metode ini menghasilkan ukuran file yang lebih kecil dan meningkatkan kualitas file PDF tanpa merusak isi teksnya (Silaban 2022). Penelitian oleh Arby Hartama pada tahun 2022 berjudul "Analisa Perbandingan *Algoritma Prefix Code* Dengan *Elias Omega Code* Dalam Merancang Aplikasi Pengkompresi *File Video Avi*" menyimpulkan bahwa kedua algoritma tersebut sangat sesuai untuk kompresi file video. Proses kompresi file video berhasil dilakukan dengan menggunakan baik algoritma *Prefix Code* maupun *Elias Omega Code* (Hartama, 2022).

Meskipun algoritma prefix code dan Burrows-Wheeler Transform (BWT) telah banyak digunakan dalam kompresi file, penelitian yang membandingkan kinerja keduanya, khususnya dalam konteks kompresi file video, masih terbatas. Untuk meningkatkan pemahaman tentang efektivitas masing-masing algoritma, perlu dilakukan analisis perbandingan yang fokus pada faktor-faktor seperti Ratio of Compression (Rc), Compression Ratio (Cr), dan Space Saving (Ss). Berdasarkan kebutuhan tersebut, penulis tertarik untuk menyusun penelitian dengan judul "Analisa Perbandingan Algoritma Prefix Code Dengan Algoritma Burrows-Wheeler Transform Dalam Kompresi File Video"

B. Metode Penelitian

Pada metodologi penelitian dilakukan klasifikasi tahapan yang dipakai pada penelitian. Metodologi penelitian terdiri atas tahapan yang mempunyai kaitan secara sistematis. Metodologi penelitian menggambarkan kerangka penelitian yang sering disebut sebagai *fase*

penelitian. Dilakukan penelitian pada Algoritma *Prefix Code* dan *Burrows-Wheeler Transform* dalam melakukan proses kompresi *file* video adalah untuk melakukan perbandingan pada saat dilakukannya proses kompresi. Tujuan dari perbandingan ini untuk mengetahui algoritma mana yang melakukan proses kompresi lebih efisien dari kedua algoritma tersebut yaitu Algoritma *Prefix Code* dan *Burrows-Wheeler Transform.*

Sampel Data

Pada penelitian ini data yang diambil sebagai sampel data *file* video berformat mp4 yang selanjutnya akan digunakan pada saat proses kompresi perbandingan Algoritma *Prefix Code* dan Algoritma *Burrows-Wheeler Transform*. Adapun sampel data yang digunakan dapat dilihat pada tabel 1. berikut ini :

Tabel 1 Sampel Data

Nama *File*	Ukuran
Going Seventeen Special. mp4	13,8 MB

C. Hasil Penelitian dan Pembahasan

Kompresi *File* Video MP4 Menggunakan Algoritma *Prefix Code*

Prefix Code adalah sistem kode yang memiliki "Property Prefix," memastikan bahwa tidak ada kode yang merupakan awalan dari kode lainnya. Dalam implementasinya, penulis menggunakan empat kode awalan: C1, C2, C3, dan C4, dengan C1 sebagai contoh. Sebelum mengompresi dan mendekompresi file video, nilai hexadecimal dari gambar diambil sebagai sampel. Selanjutnya, data sampel tersebut digunakan untuk perhitungan sesuai dengan langkah-langkah algoritma Prefix Code.

Tabel 4. Ketentuan *Prefix Code*

N	Unary	B(n)	$\bar{B}$(n)	C1
1	0	1		0|
2	10	10	1	10|0
3	110	11	00	10|1
4	1110	100	01	110|00
5	11110	101	10	110|01
6	111110	110	11	110|10
7	1111110	111	000	110|11
8		1000	001	1110|000

Tabel 5. Nilai *Hexadecimal* Dikompresi *Prefix Code*

N	*Hex*	*Frequency*	*Codeword* (C1)	Bit	Bit x *Frequency*
1	69	2	0	1	2
2	73	2	100	3	6
3	6F	2	101	3	6
4	6D	2	11000	5	10
5	31	2	11001	5	10
6	32	1	11010	5	5
7	61	1	11011	5	5
8	76	1	110000	6	6
9	63	1	110001	6	6

10	70	1	110010	6	6
11	34	1	110011	6	6
Total Bit					68

Tabel 6. *String* Bit Hasil Kompresi dengan *Prefix Code*

No.	*Hexadecimal*	*Codeword* (C1)
1	69	0
2	73	100
3	6F	101
4	6D	11000
5	32	11010
6	61	11011
7	76	110000
8	63	110001
9	31	11001
10	70	110010
11	34	110011

Penambahan padding dan flagging pada string bit yang dihasilkan dapat dilihat dalam uraian berikut:

7 – n + "1"
7 – 4 + "1" = 0001
Bit akhir 9 – n
Bit akhir 9 – 4 = 5
= 00000101

Gambar 2 Perhitungan Penambahan *Padding* dan *Flagging*

01001011 10000100 10111010
11011110 00011000 11100111
00011001 01100111 10010001
00000101

Gambar 3 Hasil Penambahan *Padding* dan *Flagging*

Langkah selanjutnya membagi string bit menjadi per 8 bit lalu merubahnya ke *hexadecimal* dan mengubahnya ke karakter.

Tabel 7. Pengelompokan Bit dan Pencarian Karakter

No.	*String* Bit Hasil Kompresi	*Hexadecimal*	Karakter
1	01001011	4B	K
2	10000100	84	„
3	10111010	BA	º
4	11011110	DE	Þ
5	00011000	18	„
6	11100111	E7	ç
7	00011001	19	
8	01100111	67	g
9	10010001	91	
10	00000101	05	

Dekompresi Algoritma *Prefix Code*

Proses dekompresi yang dilakukan adalah menganalisa semua bit hasil dari kompresi sebelumnya. Adapun bit keseluruhan hasil kompresi dapat dilihat pada tabel berikut:

Tabel 8 Nilai *Hexadecimal* Video Terkompresi *Prefix Code*

No.	*String* Bit Hasil Kompresi	*Hexadecimal*	Karakter
1	01001011	4B	K
2	10000100	84	„
3	1011101	BA	º

	0		
4	11011110	DE	Þ
5	00011000	18	”
6	11100111	E7	ç
7	00011001	19	
8	01100111	67	g
9	10010001	91	
10	00000101	05	

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

```
01001011100001001011101011011110000110001110011100011001011100111001000100000101
```

Gambar 4. Nilai Biner Dari Hasil Algoritma *Prefix Code*

Selanjutnya, bilangan biner dikembalikan menjadi string bit semula dengan menghilangkan padding dan flagging. Proses ini dilakukan dengan langkah-langkah berikut: pertama, baca 8 bit terakhir dari string bit. Kedua, tentukan nilai \(n \) dari hasil pembacaan, lalu hapus 7 + \(n \) bit dari bagian akhir (misalnya, jika \(n \) adalah 5, maka hapus 12 bit). Terakhir, setelah mengurangi 12 bit dari akhir, bilangan biner dikembalikan menjadi string bit semula.

```
01001011100001001011101011011110000110001110011100011001011001110010
```

Gambar 1 Menghilangkan *Padding* dan *Flagging* pada Bilangan Biner

Kompresi *File* Video MP4 Menggunakan Algoritma *Burrows-Wheeler Transform*

Algoritma Burrows-Wheeler Transform (BWT) mentransformasikan blok data teks menjadi format baru dengan karakter yang serupa, melalui proses pengurutan data. Sebelum mengompresi dan mendekompresi file video, sampel nilai hexadecimal diambil dari gambar. Setelah memperoleh nilai hexadecimal, data sampel tersebut digunakan untuk perhitungan sesuai dengan langkah-langkah algoritma BWT.

Tabel 9. Nilai *Hexadecimal* Sebelum Dikompresi BWT

Hex	***Frequency***	***Binary***	**Bit**	**Bit x *Frequency***
69	2	01101001	8	16
73	2	01110011	8	16
6F	2	01101111	8	16
6D	2	01101101	8	16
31	2	00110001	8	16
32	1	00110010	8	8
61	1	01100001	8	8
76	1	01110110	8	8

Lanjutan Tabel 10 Nilai *Hexadecimal* Sebelum Dikompresi BWT

63	1	01100011	8	8
70	1	01110000	8	8
34	1	00110100	8	8
Ukuran Sebelum Dikompresi				128

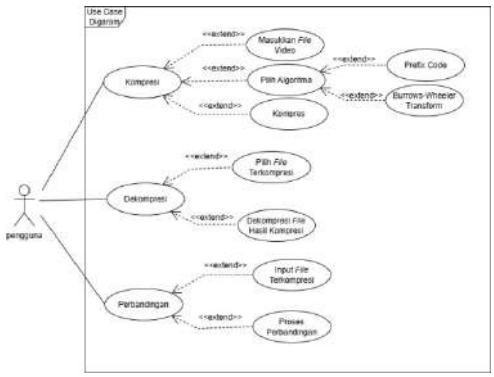
Ambil nilai pada *index* ke delapan.

Tabel13 Hasil Dekompresi

69	73	6F	6D	69	73	6F	32	61	76	63	31	6D	70	34	31

Use Case Diagram

Use case diagram merupakan model fungsional dari sistem yang memaparkan interaksi para pengguna sistem tersebut. *Use case diagram* adalah layanan fungsi-fungsi yang disediakan oleh sistem yang menampilkan gambaran bagaimana itu digunakan dengan mudah. Dalam *use case* pengguna disebut sebagai aktor yang dimainkan seseorang dalam kaitannya dengan sistem. Adapun *use case* diagram pada aplikasi yang akan dirancang dapat dilihat pada gambar di bawah ini:



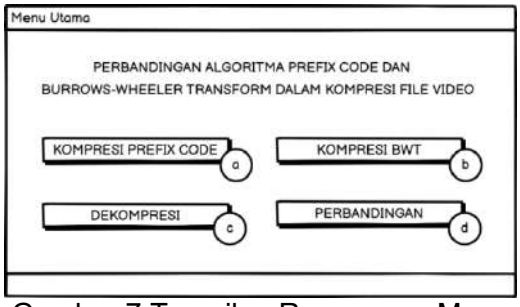
Gambar 2 *Use Case Diagram*

Perancangan Antarmuka (*Interface*)

Perancangan antarmuka atau *interface* adalah gambaran dari perancangan *form* aplikasi kompresi perbandingan algoritma *Prefix Code* dan *Burrows-Wheeler Transform*. Perancangan antarmuka adalah perancangan tahap awal atau gambaran yang menjelaskan isi dari aplikasi kompresi perbandingan algoritma *Prefix Code* dan *Burrows-Wheeler Transform* yang akan dibangun. Bahasa pemrograman Microsoft Visual Basic 2010 akan digunakan dalam pembuatan sistem ini.

Rancangan *Form* Menu Utama

Berikut gambaran *form* menu utama.

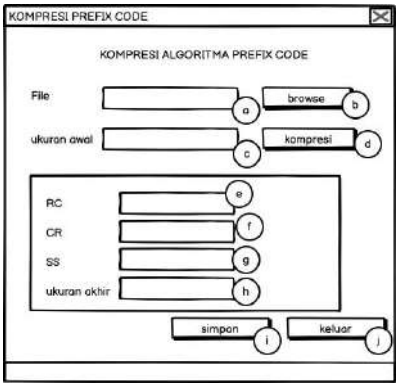


Gambar 7 Tampilan Rancangan Menu Utama

Rancangan *Form* Kompresi

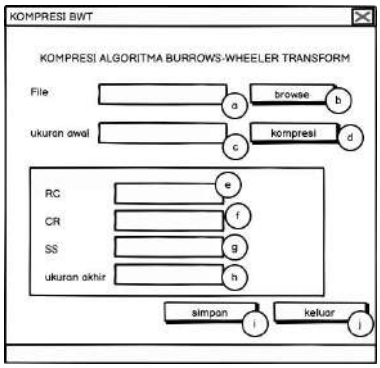
Form kompresi merupakan *form* proses kompresi *file* video dilakukan dengan menampilkan keterangan *file*, ukuran, serta hasil kompresi. Adapun rancangan *form* kompresi dapat dilihat sebagai berikut.

Rancangan *form* kompresi algoritma *Prefix Code* dapat dilihat pada gambar di bawah ini:



Gambar 8 Tampilan Rancangan *Form* Kompresi *Prefix Code*

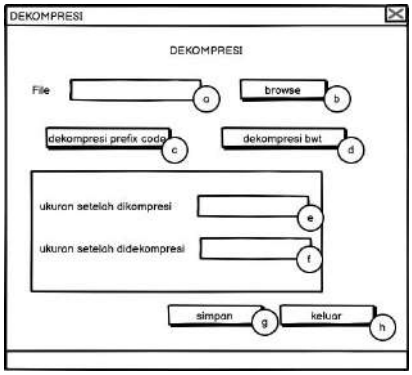
Rancangan *form* kompresi algoritma *Burrows-Wheeler Transform* dapat dilihat pada gambar di bawah ini:



Gambar 9 Tampilan Rancangan *Form* Kompresi BWT

Rancangan *Form* Dekompresi

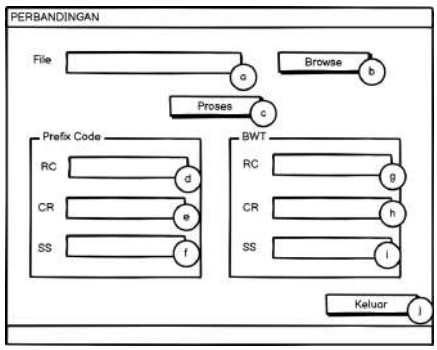
Form dekompresi merupakan *form* proses dekompresi *file* video dilakukan. Pada *form* ini pengguna memasukkan *file* yang telah dikompresi untuk dikembalikan kebentuk semula. Adapun *form* dekompresi dapat dilihat sebagai berikut.



Gambar 10 Tampilan Rancangan *Form* Dekompresi

Rancangan *Form* Perbandingan

Form perbandingan digunakan untuk melihat hasil nilai parameter kedua algoritma tersebut, selain itu *form* perbandingan ini digunakan untuk melihat algoritma mana yang paling efektif. Bentuk menu perbandingan dapat dilihat pada gambar di bawah ini:



Gambar 11 Tampilan Rancangan *Form* Perbandingan

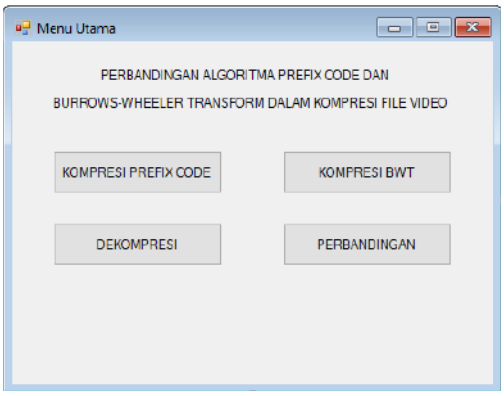
Implementasi Program

Tahap implementasi program adalah saat sistem yang telah dirancang diuji. Bagian ini mencakup spesifikasi perangkat keras (*hardware*) dan perangkat lunak (*software*), serta memberikan

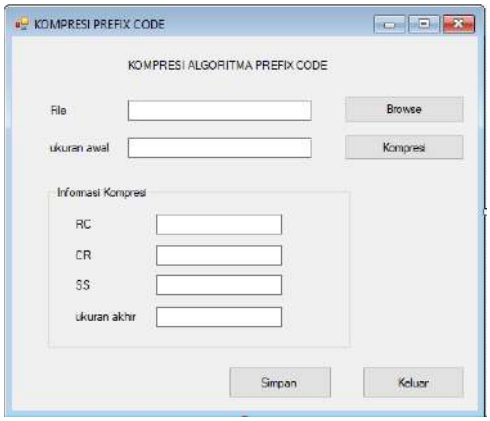
gambaran tentang tampilan sistem saat dijalankan.

Tampilan Sistem

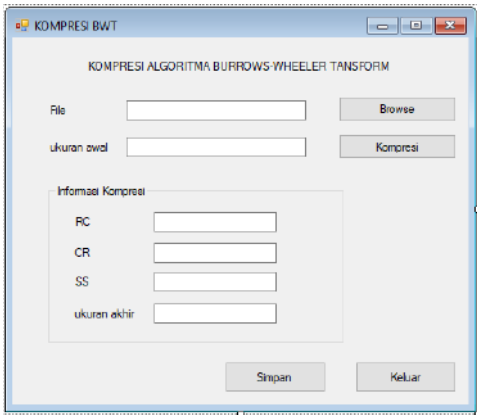
Tampilan sistem merupakan hasil akhir dari program yang telah dikembangkan oleh penulis sesuai dengan perancangan sistem yang telah dibuat. Dengan mengacu pada perancangan sistem tersebut, berikut adalah hasil dari program kompresi *file* video yang menggunakan algoritma *Prefix Code* dan *Burrows-Wheeler Transform*.



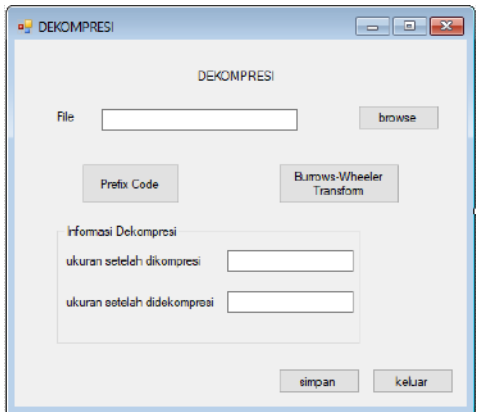
Gambar 12 Tampilan *Form* Menu Utama



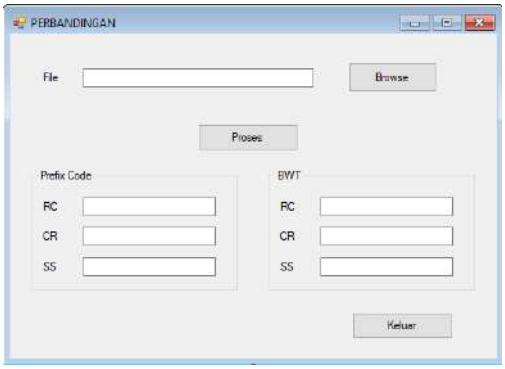
Gambar 13 Tampilan *Form* Kompresi *Prefix Code*



Gambar 14 Tampilan *Form* Kompresi *Burrows-Wheeler Transform*



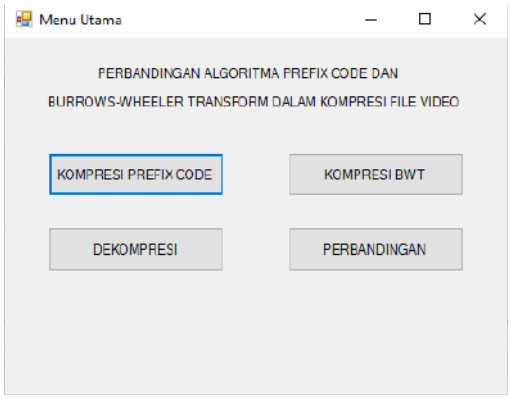
Gambar 15 Tampilan *Form* Dekompresi



Gambar 16 Tampilan *Form* Perbandingan

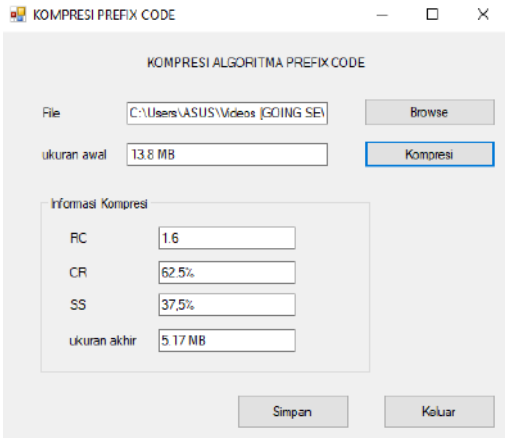
Pengujian Sistem

1. Tampilan *Form* Menu Utama
   Tampilan *form* menu utama pada sistem dapat dilihat pada gambar dibawah ini:



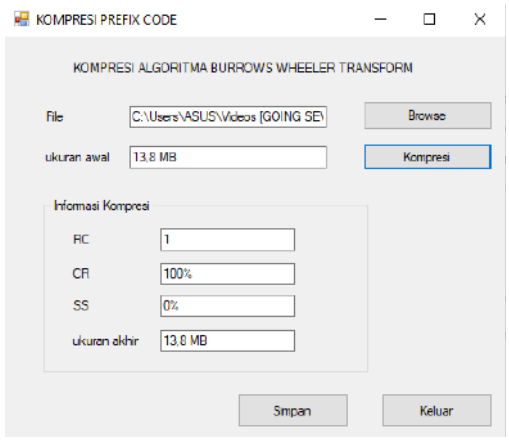
Gambar 17 Tampilan Menu Utama

2. Tampilan *Form* Proses Kompresi
   a. Tampilan proses kompresi algoritma *Prefix Code* pada sistem dapat dilihat pada gambar di bawah ini:



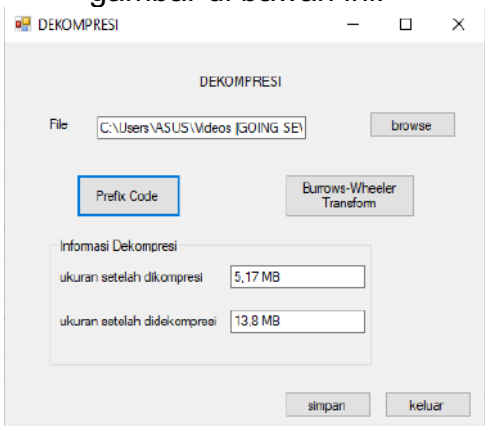
Gambar 18 Tampilan Proses Kompresi *Prefix Code*

   b. Tampilan proses kompresi algoritma *Burrows-Wheeler Transform* pada sistem dapat dilihat pada gambar di bawah ini:



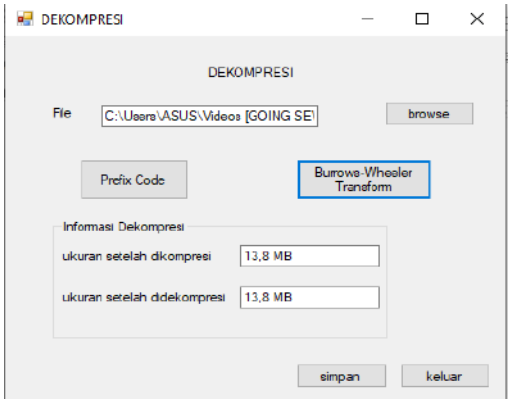
Gambar 19 Tampilan Proses Kompresi *Burrows-Wheeler Transform*

3. Tampilan *Form* Proses Dekompresi
   a. Tampilan proses dekompresi algoritma *Prefix Code* pada sistem dapat dilihat pada gambar di bawah ini:

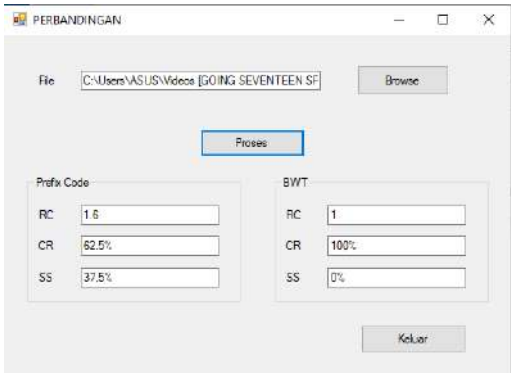


Gambar 20 Tampilan Proses Dekompresi *Prefix Code*

   b. Tampilan proses dekompresi algoritma *Burrows-Wheeler Transform* pada sistem dapat dilihat pada gambar di bawah ini:



Gambar 21 Tampilan Proses Dekompresi *Burrows-Wheeler Transform*



Gambar 22 Tampilan Proses Perbandingan

4. Tampilan *Form* Proses Perbandingan

   Tampilan proses perbandingan algoritma *Prefix Code* dan *Burrows-Wheeler Transform* pada sistem dapat dilihat pada gambar di bawah ini:

Hasil Pengujian

Perbedaan antara *file* video sebelum kompresi dan setelah kompresi serta hasil perbandingan dari hasil pengujian dapat dilihat di bawah ini:

Tabel 15 Tabel Hasil Pengujian

Algoritma	Ukuran Awal	Ukuran Setelah Kompresi	Parameter			Algoritma Efektif
			RC	CR	SS	
Prefix Code	13,8 MB	5,17 MB	1,6	62,5 %	37,5%	1
Burrows-Wheeler Transform	13,8 MB	13,8 KB	1	100%	0%	2

D. Kesimpulan

Berdasarkan analisis komparatif antara algoritma Prefix Code dan Burrows-Wheeler Transform (BWT) dalam kompresi file video, dapat disimpulkan bahwa algoritma Prefix Code lebih unggul dibandingkan dengan BWT. Proses kompresi dimulai dengan pemilihan file video, yang kemudian dikompresi menggunakan kedua algoritma untuk mengukur efektivitas masing-masing. Hasil percobaan menunjukkan bahwa algoritma Prefix Code berhasil menghasilkan Ratio of Compression sebesar 1,6, Compression Ratio

sebesar 62,5%, dan Space Saving sebesar 37,5%, yang berarti ukuran file berkurang dari 13,8MB menjadi 5,17MB. Sebaliknya, BWT menghasilkan Ratio of Compression sebesar 1, Compression Ratio sebesar 100%, dan Space Saving sebesar 0%, yang menunjukkan bahwa BWT tidak mengurangi ukuran file melainkan hanya mengubah posisi data di dalam file tersebut. Aplikasi yang dikembangkan menggunakan Microsoft Visual Studio 2010 mampu melakukan kompresi file video dengan kedua algoritma dan membandingkan hasilnya secara efektif, menggarisbawahi keunggulan algoritma Prefix Code dalam hal pengurangan ukuran file.

DAFTAR PUSTAKA

Afriyadi. 2023. "Media Pembelajaran Berbasis Digital (Teori & Praktik)." in *PT. Sonpedia Publishing Indonesia.*

Alfarizi, M., and S. Aripin. 2020. "Penarapan Algoritma Prefix Code Dalam Kompresi File Video." *KOMIK (Konferensi Nasional …* 4:249–52. doi: 10.30865/komik.v4i1.2686.

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.

Cox, Anthony J., Markus J. Bauer, Tobias Jakobi, and Giovanna Rosone. 2012. "Large-Scale Compression of Genomic Sequence Databases with the Burrows – Wheeler Transform." *BIOINFORMATICS* 28(11):1415–19.

Ginting, Septriani Br. 2021. "Perbandingan Algoritma Yamamoto ' s Recursive Code Dan Additive Code Dalam Kompresi File Video." *KOMIK (Konferensi Nasional Teknologi Informasi Dan Komputer)* 5. doi: 10.30865/komik.v5i1.3819.

Hartama, Arby. 2022. "Analisis Perbandingan Algoritma Prefix Code Dengan Elias Omega Code Dalam Merancang Aplikasi Pengkompresi File Video Avi." *KOMIK* 6(1):260–70.

Huda. 2020. "Media Animasi Digital Berbasis Hots (Higher Order Thinking Skill) ." in *Unp Press.*

Ramadhan, Muhammad Rizky. 2023. "Analisa Perbandingan Algoritma Run Length Encoding Dengan Burrows-Wheeler Transform Dalam Kompresi File Video." *KOMIK (Konferensi Nasional Teknologi Informasi Dan Komputer)* 6(1):322–32. doi: 10.30865/komik.v6i1.5716.

Saragih, S. R., and D. P. Utomo. 2020. "Penarapan Algoritma Prefix Code Dalam Kompresi Data Teks." *KOMIK (Konferensi Nasional …* 4(1):249–52.

Sepandi. 2020. "Penerapan Algoritma Adaptive Huffman Coding Pada Aplikasi Kumpulan Komik

Berbasis Android." *Pelita Informatika : Informasi Dan Informatika* 9(1):136–41.

Sihotang, Agustrina. 2022. "Implementasi Algoritma Prefix Codes Untuk Kompresi File Video Hasil Ekstra Aplikasi Kinemaster." *BIOSTech: Bulletin of Computer Science and Information Technology* 1(1):22–29.

Silaban, Boy Alfredo. 2022. "Analisa Kompresi File Teks Dengan Kombinasi Metode Burrows-Wheeler Transform Dan Shannon-Fano." 6(November):707–15. doi: 10.30865/komik.v6i1.5760.

Sodikin, Luthfia, Tineke Fatma Putri, and Taufik Hidayat. 2022. "Analisa Kompresi File Teks Menggunakan Algortima Huffman." *Jurnal ICTEE,* 3(1):10–19.

Utari, Cut Try, Program Studi, Magister Teknik, Universitas Sumatera Utara, and Kompresi Citra. 2016. "IMPLEMENTASI ALGORITMA RUN LENGTH ENCODING UNTUK PERANCANGANAPLIKASI KOMPRESI DAN DEKOMPRESI." *Jurnal TIMES* 5(2):24–31.