KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer) ISSN 2597-4645 (media online)
Volume 6, Nomor 1, November 2022 ISSN 2597-4610 (media cetak)
DOI: 10.30865/komik.v6i1.5760 Page: 707-715

Analisa Kompresi File Teks Dengan Kombinasi Metode Burrows-Wheeler Transform Dan Shannon-Fano

Boy Alfredo Silaban

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

Abstrak- Dengan berkembangnya Teknologi komputer sekarang ini dapat dimanfaatkan untuk berbagai hal, salah satunya adalah menyimpan file. Semakin besar ukuran file yang disimpan di dalam komputer maka semakin besar pula tempat penyimpanannya. Semakin besar tempat penyimpanan yang dibutuhkan maka semakin besar pula biaya yang dikeluarkan. Selain itu file yang memiliki ukuran besar dapat menimbulkan masalah pada saat transmisi file. Oleh karena itu kemudian muncul metode-metode yang bertujuan untuk memampatkan file agar dapat menghemat tempat penyimpanan file-file tersebut, salah satunya dengan cara kompresi. Dengan adanya kompresi diharapkan dapat menghemat biaya serta waktu yang dikeluarkan guna menambah fasilitas media penyimpanan file pada komputer serta mempercepat proses transfer file. Setelah dilakukan proses kompresi dari algoritma Shannon-Fano dan BWT, maka dapat disimpulkan bahwa Kombinasi Algoritma BWT dan Algoritma Shannon-Fano lebih mampu menurunkan kapasitas ukuran dalam kompresi file teks pdf, dikarenakan proses kompresi menggunakan Kombinasi Algoritma BWT dan Algoritma Shannon-Fano mampu mengurangi ukuran data dari nilai awal. Kombinasi Algoritma BWT dan Algoritma Shannon-Fano mampu mengkompres file teks pdf dengan lebih efisien dibanding aplikasi-aplikasi kompresi dari web online yang banyak beredar.

Kata kunci: Kompresi File Teks:Kombinasi algoritma;algoritma Shanon Fano;alogritma Burrow Wheeler Transform

Abstract-With the development of computer technology today it can be used for various things, one of which is storing files. The larger the size of the file stored on the computer, the greater the storage space. The greater the storage space required, the greater the costs incurred. In addition, files that have a large size can cause problems when transmitting files. Therefore, methods emerged that aimed to compress files in order to save storage space for these files, one of which was compression. With compression, it is hoped that it can save costs and time spent in order to add file storage media facilities on a computer and speed up the file transfer process. After carrying out the compression process of the Shannon-Fano and BWT algorithms, it can be concluded that the combination of the BWT algorithm and the Shannon-Fano algorithm is more capable of reducing the size capacity in compressed pdf text files, because the compression process using the combination of the BWT algorithm and the Shannon-Fano algorithm is able to reduce the size data from the initial value. The combination of the BWT Algorithm and the Shannon-Fano Algorithm is able to compress pdf text files more efficiently than compression applications from the widely circulated online web.

Keywords: Text File Compression;Combination algorithm, Shanon Fano algorithm;Burrow Wheeler Transform algorithm

1. PENDAHULUAN

Karena ukuran file terus bertambah, pengguna komputer terpaksa mencari cara berbeda untuk menyimpan file dalam jumlah besar pada media penyimpanan terbatas. salah satunya adalah file teks. Walaupun file teks tidak memiliki ukuran yang besar seperti file teks namun tetap saja ukuran dari file teks akan mengalami pembesaran mengikuti isi dari file teks tersebut. Semakin banyak file teks yang disimpan, maka juga semakin ikut mengisi ruang penyimpanan. Selain itu, ukuran file juga mempengaruhi transfer data, cepat atau lambatnya suatu proses transfer data dipengaruhi oleh ukuran dari file tersebut.

Masalah yang sering di temui oleh penulis yaitu pada beberapa kelompok orang yang ingin melakukan kompresi file teks dengan ukuran lebih kecil yang akan digunakan untuk keperluan tugas sekolah, tugas kuliah, melamar kerja bahkan kebutuhan pekerjaan yang mengharuskan file teks tersebut sesuai peraturan yang dibuat.

Kendala yang dialami sekelompok orang tersebut adalah seringnya mengkompres file teks dengan menggunakan aplikasi kompresi online dan yang penulis temukan bahwa banyak penyedia layanan aplikasi kompresi online tidak memenuhi standart ukuran file dan standart kualitas yang baik.

Sehingga penulis meneliti algoritma yang cocok dan efisien dalam membuat sebuah aplikasi kompresi dan peneliti tertarik meneliti kombinasi dari 2 algoritma yang dipadukan menjadi sebuah aplikasi yang efisien yang kelak dapat membantu banyak orang untuk mendapatkan ukuran dan kualitas yang baik.

Ada beberapa algoritma kompresi yang dapat digunakan untuk memampatkan ukuran data dari sebuah file, antaranya adalah BWT dan *Shannon Fano.* Kedua algoritma ini memiliki keunikan masing-masing seperti algoritma *BWT* merupakan algoritma yang mengubah data awal sebelum dilakukan proses pemampatan data. Sedangkan algoritma *Shannon- Fano* merupakan metode yang paling baik tetapi hampir tidak pernah digunakan dan dikembangkan lagi setelah kemunculan algoritma Huffman. Pada dasarnya metode ini menggantikan setiap simbol dengan sebuah alternatif kode biner yang panjangnya ditentukan berdasarkan probabilitas dari simbol tersebut[1].

Penelitian yang dilakukan oleh Asrul Soleh Harahap, (2021) dengan algoritma *BWT* untuk kompresi file citra. Penelitian ini Menganalisis dan Menerapkan Kompresi File Gambar Menggunakan Algoritma Transformasi Burrows-Wheeler Pada penelitian ini, algoritma transformasi Burrows-Wheeler memproses data input dalam antrian daripada mengompresnya, langsung memproses blok data teks sebagai unit, dapat diproses. Surat yang sama berurutan. Hasil kompresi file gambar sebelumnya terdiri dari 72 bit [2]. Namun, mengompresi ke 22 bit menghasilkan rasio besar 69,5%.

Penelitian yang dilakukan oleh Muhammad Rajani Pane, (2017) menggunakan algoritma shannon fano. Penelitian ini telah berhasil membuat aplikasi kompresi teks Shannon Fano sehingga mendapatkan hasil sesuai dengan proses

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 707-715

kompresi yang dikerjakan. Pada saat proses kompresi, aplikasi ini membaca karakter yang ada dalam data teks, kompresi file teks yang efektif untuk digunakan dengan melakukan perbandingan kompresi file teks dengan menggunakan metode Shannon Fano. Dapat dilihat pada data teks berukuran kecil, rasio kompresi teks dengan menerapkan Algoritma Shannon Fano ukuran sama, hal ini disebabkan nilai biner yang dibentuk algoritma hampir menyerupai.

Penelitian yang dilakukan oleh Saputri, (2018) Untuk kompresi file teks menggunakan algoritma Shannon-Fano, proses algoritma Shannon-Fano menghasilkan rasio kompresi sebesar 48,92%, dimana 48,92% data yang dikirimkan berhasil dikompresi. Kemudian algoritma Shannon-Fano dapat dipilih sebagai algoritma yang cocok untuk melakukan proses kompresi data [4].

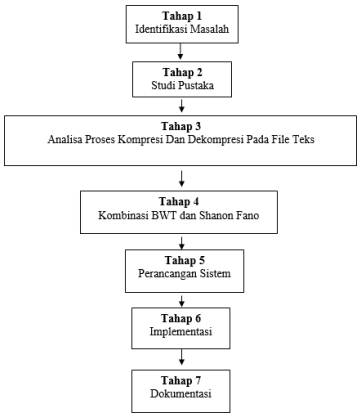
Abdullah, A. R (2016), dalam skripsi ini dapat diambil kesimpulan bahwa, proses kompresi pada algoritma Even Rodeh Code dan algoritma Variable Length Binary Encoding dipengaruhi oleh jumlah variasi karakter. Jika dalam karakter yang sama (homogen) hasil kompresi dan dekompresi pada algoritma *Variable Length Binary Encoding* lebih baik dibandingkan dengan algoritma *Even Rodeh Code*. Sebaliknya jika dalam karakter yang berbeda (heterogen) hasil kompresi dan dekompresi, algoritma *Even Rodeh Code* lebih baik dibandingkan dengan algoritma *Variable Length Binary Encoding*[5].

Yayuk Anggraini pada tahun 2015, penelitian ini melakukan perbandingan hasil kombinasi metode kompresi *BWT + MTF + Shannon-Fano* dengan metode tunggal *LZ77*. Hasil penelitian ini membuktikan kombinasi *BWT + MTF +* Shannon-Fano memiliki rasio kompresi lebih tinggi 0,37% dibandingkan dengan *LZ77*. Pada aspek waktu kompresi, *LZ77* membutuhkan 39 detik. Sedangkan metode kombinasi yang diusulkan hanya membutuhkan setitar 1,3 detik. Metode kombinasi yang diusulkan memiliki kelebihan implementasi yang lebih stabil pada spesifikasi hardware yang lebih rendah[6].

Syarifah Keumala Adriaty (2015) melakukan penelitian - Fano, Kombinasi Algoritma Arithmetic Coding dan Huffman Pada Kompresi Berkas[7]. Pada penelitian ini dilakukan implementasi beberapa algoritma kompresi yang bersifat lossless, yaitu Shannon-Fano, Arithmetic Coding dan Huffman yang bertujuan untuk mengetahui algoritma paling optimal di antara ketiga algoritma tersebut. Parameter perbandingan kinerja algoritma yang digunakan adalah waktu kompresi, rasio kompresi, faktor kompresi, saving percentage, dan kompleksitas algoritma (Big-O). Penelitian ini menyimpulkan kombinasi di antara algoritma *Shannon-Fano, Arithmetic Coding, dan Huffman,* kompleksitas waktu kompresi dan dekompresi berkas teks terbaik adalah algoritma Shannon Fano, sedangkan kompleksitas waktu kompresi dan dekompresi berkas citra digital terbaik adalah algoritma Huffman, dan kompleksitas waktu kompresi dan dekompresi berkas teks dan berkas citra digital terburuk terdapat pada algoritma Arithmetic Coding.

2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian



Gambar 1. Tahapan Penelitian

Berikut ini dijelaskan tahapan-tahapan yang dilakukan pada penelitian ini:
1. Identifikasi Masalah
2. Studi pustaka
3. Analisa proses kompresi dan dekompresi pada file teks
4. Kombinasi BWT dan dekompresi pada file teks
5. Perancangan sistem
6. implementasi

2.2 Kompres File Teks

Berapa banyak jumlah serangkaian bit yang lebih sedikit tersebut ? Ini tergantung pada algoritmanya tetapi juga akan tergantung pada seberapa banyak redundansi dapat diekstraksi dari data asli. Data yang berbeda mungkin memerlukan teknik yang berbeda untuk mengidentifikasi redundansi dan untuk menghilangkan redundansi dalam data. Jelas, ini

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 707-715

membuat masalah kompresi cukup sulit untuk. Oleh karena itu dibutuhkan pertimbangan yang matang untuk merancang suatu algoritma.

Tidak ada satu ukuran yang cocok untuk dijadikan solusi untuk semua masalah kompresi data. Dalam data studi kompresi, kita pada dasarnya perlu menganalisa karakteristik data yang akan dikompresi dan berharap untuk menyimpulkan beberapa pola dalam rangka mencapai representasi yang maksimal. Hal ini menimbulkan berbagai pemodelan data dan berbagai representasi teknik, yang berada di inti teknik kompresi.

Kompresi data dapat dilihat sebagai alat untuk secara efisien mewakili sumber data digital seperti teks, audio, gambar, atau kombinasi dari semua jenis seperti video. Tujuan kompresi data adalah untuk mewakili data digital Formulir II-2 dalam bit sesedikit mungkin sambil memenuhi persyaratan minimum untuk memulihkan data asli. Kompresi data (singkatnya kompresi) dalam konteks ini berarti suatu algoritma untuk mencapai tujuan mengompresi data sumber. Dibalik setiap algoritma ada ide dan model matematika atau pelaksanaan teknik untuk mencapai kompresi. Ketika bekerja pada masalah kompresi, kita perlu mempertimbangkan efisiensi aspek dari algoritma serta efektivitas kompresi. Secara dasar, perilaku algoritma kompresi akan tergantung pada data dan struktur internal data tersebut. Semakin banyak redundansi maka akan semakin efektif suatu algoritma berfungsi. Kompresi data adalah “proses mengubah sebuah input stream data (sumber atau data mentah asli) ke lain aliran data (output, bitstream, atau aliran terkompresi) yang memiliki ukuran yang lebih kecil” .

File teks adalah file yang berisi informasi dalam format teks. Contoh input data tekstual yang terdiri dari huruf, angka, dan tanda baca antara lain data yang berasal dari dokumen pengolah kata, angka yang digunakan dalam perhitungan, serta nama dan alamat dalam database

Input dan output dari data tekstual direpresentasikan sebagai set karakter atau sistem kode yang dikenali oleh sistem komputer [9]. Ada tiga set karakter yang biasa digunakan untuk input dan output

komputer: ASCII, EBCDIC, dan Unicode. ASCII (American Code for Information Interchange) adalah standar internasional untuk pengkodean karakter dan simbol seperti Hex dan Unicode, tetapi ASCII lebih universal. ASCII digunakan untuk merepresentasikan teks di komputer dan alat komunikasi lainnya. Kode ASCII adalah struktur bilangan biner 8-bit dari 00000000 hingga 11111111. Total 256 kombinasi dihasilkan dari kode 0 hingga 255 dalam desimal. EBCDIC (Extended Binary Codec Decimal Interchange Code) adalah kumpulan karakter yang dibuat oleh komputer bermerek IBM. EBCDIC terdiri dari 256 karakter, masing-masing berukuran 8 bit. Keterbatasan kode ASCII dan EBCDIC menciptakan standar kode internasional baru, kode 16-bit yang disebut Unicode.

1. Format data teks (*.txt)
   Format data teks adalah format teks yang digunakan untuk menyimpan huruf, angka, karakter kontrol (seperti tab dan baris baru), atau simbol lain yang umum digunakan seperti titik, koma, dan tanda kutip.
2. Format data dokumen (*.doc)
   Doc adalah ekstensi arsip dokumen perangkat lunak Microsoft Word yang paling banyak digunakan untuk menulis laporan, makalah, dan lainnya.
3. Hyper Text Markup Language (*.htm atau *.html) Format teks default untuk melihat dokumen web.
4. Rich Text Format (*.rtf)
   Dikembangkan oleh Microsoft, format teks ini dapat dibaca di berbagai platform seperti Windows, Linux, dan Mac OS. 22 2.1.2 Jenis teks Jenis teks merupakan jenis yang sangat familiar dan mendasar dalam kehidupan sehari- hari.

3. HASIL DAN PEMBAHASAN

3.1 Analisa kerangka Kerja Penelitian

Analisis masalah yang didapat dari pekerjaan ini adalah melakukan kompresi data menggunakan algoritma Burrows-Wheeler Transform (BWT) dan Shannon-Fano. Pengujian dalam penelitian ini meliputi beberapa kasus uji seperti:

1) Pengujian dengan berbagai ukuran buffer pencarian Pengujian dilakukan dengan berbagai ukuran buffer pencarian
2) Pemeriksaan jenis file Periksa setiap jenis file teks (pdf)
3) Menguji beberapa file secara bersamaan Pengujian dijalankan terhadap banyak file dalam memori sekaligus.

Dalam melakukan penelitian yang berhubungan dengan kompresi data, ada beberapa tahapan penelitian yang dilakukan oleh penulis agar sistem yang dibuat tidak mengalami kegagalan. Tahapan penelitian ini terdiri dari 7 Tahap yang terpisah dimana setiap Tahap memiliki fungsi dan kegunaannya masing-masing. Gambar dibawah ini adalah tahapan penelitian yang penulis lakukan

3.2 Sampel Data

Berikut gambar keterangan sampel data yang digunakan pada penelitian ini:

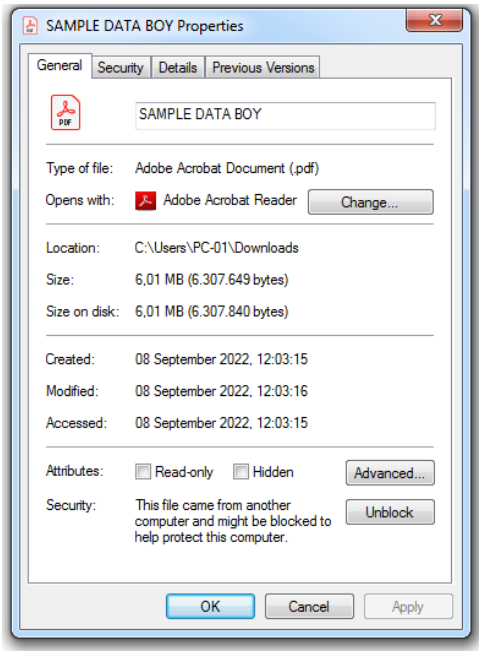
Tabel 1. Keterangan Sample Data

	Keterangan
Nama File	Sample Data Boy
Jenis Item	Pdf

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 707-715

Ukuran	6,01 Mb

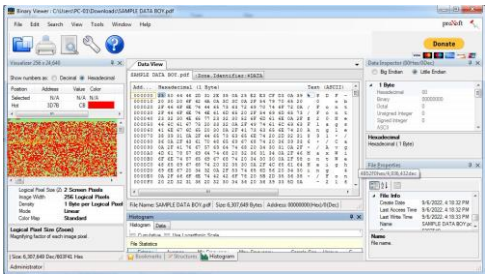
Selanjutnya dijelaskan bentuk sampel data, dapat dilihat pada gambar 3 dibawah ini:



Gambar 3. Sample Data

3.2.1 Perhitungan Ukuran Awal Sampe Data

Sebelum melakukan proses kompresi maka terlebih dahulu dilakukan perhitungan ukuran awal dari sample data. Untuk melakukan perhitungan maka terlebih dahulu *fileteks* akan dibaca nilai hexadecimalnya terlebih dahulu menggunakan aplikasi *binary viewer*.



Gambar 4. Nilai *Hexadecimal* Dilihat Dengan Aplikasi Binary Viewer

Dari aplikasi *binary viewer* diatas, diambil sebanyak 16 nilai *hexadecimal* sebagai sample data yang akan dihitung ukuran awalnya dan untuk proses kompresi.

Tabel 2. Nilai Hexadesimal Sample Data

25	50	44	46	2D	31	2E	35
0A	25	E2	E3	CF	D3	0A	39

Selanjutnya perhitungan ukuran awal dijelaskan pada tabel 3 berikut ini:

Tabel 3. Perhitungan Ukuran Awal

No	Nilai Hexadesimal	Frekuensi	Biner	Bit	Bit x Frek
1	0A	2	00001010	8	16
2	25	2	00100101	8	16
3	2E	1	00101110	8	8
4	31	1	00110001	8	8
5	35	1	00110101	8	8
6	46	1	01000110	8	8
7	44	1	01000100	8	8
8	50	1	01010000	8	8

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 707-715

No	Nilai Hexadesimal	Frekuensi	Biner	Bit	Bit x Frek
9	2D	1	00101101	8	8
10	D3	1	11010011	8	8
11	39	1	00111001	8	8
12	CF	1	11001111	8	8
13	E3	1	11100011	8	8
14	E2	1	11100010	8	8
		Total Bit			128

3.2.2 Kompresi Dengan Algoritma *Shanon Fano*

Berikut ini tabel penjelasan pengurusan kode shanon fano:

Tabel 4. Pengurutan Kode *Shanon Fano*

No	Nilai Hexadesimal	Frekuensi	Shannon Fano	Bit	Bit x Frek
1	0A	2	10	2	4
2	25	2	110 0	4	8
3	2E	1	110 1	4	4
4	31	1	1110 0 00	7	7
5	35	1	1110 0 01	7	7
6	46	1	1110 0 10	7	7
7	44	1	1110 0 11	7	7
8	50	1	1110 1 000	8	8
9	2D	1	1110 1 001	8	8
10	D3	1	1110 1 010	8	8
11	39	1	1110 1 011	8	8
12	CF	1	1110 1 110	8	8
13	E3	1	1110 1 111	8	8
14	E2	1	11110 1 000	9	9
		Total Bit			101

Tabel 5. *String Bit* Hasil Kompresi Menggunakan Algoritma *Shanon Fano*

25	50	44	46	2D	31	2E	35
1100	11101000	1110011	1110010	11101001	1110000	1101	1110001
0A	25	E2	E3	CF	D3	0A	39
10	1100	1111010 00	11101111	11101110	11101010	10	11101011

Berdasarkan tabel 3.5 *string bit* yang dihasilkan dari proses pengkompresian *file*teks menggunakan algoritma *Shanon Fano* dapat dituliskan sebagai berikut:
"11001110100011100111110010111010011110000110111100011011001111010
00111011111110111011101010111101011"
Dikarenakan 101 tidak habis dibagi 8 atau bukan kelipatan 8 maka dibentuk variabel yang disebut dengan *padding* dan *flagging* untuk penambahan *bit* data. *Padding* merupakan penambahan *bit* data yang telah dikompres agar seluruh jumlah *bit* data tersebut kelipatan 8. Rumus *padding* ialah 7 − n + "1". Sedangkan *flagging* merupakan penambahan 8 *bit* bilangan biner setelah *padding* yang dimana untuk mempermudah dalam membaca *bit-bit* hasil kompresi pada saat proses dekompresi. Rumus *flagging* ialah 9 −*n*.
Apabila jumlah *bit* habis dibagi 8 atau kelipatan 8, maka tidak perlu dilakukan *padding*, tetapi tetap harus menambahan *flagging.*

Tabel 6. Tabel Padding dan Flagging

Padding	Flagging
101Mod 8 = 6= n	9 – n
7 - 5 + "1"	9 - 6= 3
7 - 5 + "1" = 001	Biner 2 = 00000011

Jadi *string bit* yang terbentuk setelah penambahan *padding* dan *flagging* yaitu :
11001110 10001110 01111100 10111010 01111000 01101111 00011011 00111101 00011101 11111101 11011101 01010111 01011**001 00000011"**

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 707-715

Sehingga total panjang *bit* adalah 88. Selanjutnya lakukan pemisahan *bit* menjadi beberapa kelompok. Setiap kelompok terdiri dari 8 *bit*.

Tabel 7. Karakter ASCI

No.	Biner	Desimal	Karakter
1	11001110	206	Î
2	10001110	142	
3	01111100	124	|
4	10111010	186	º
5	01111000	120	x
6	01101111	111	o
7	00011011	27	
8	00111101	61	=
9	00011101	29	
10	11111101	253	ý
11	11011101	221	Ý
12	01010111	87	W
13	01011001	89	Y
14	00000011	3	

3.2.4 Kompresi Dengan Algoritma *BWT*

BWT yaitu dengan menyajikan data diatas ke dalam bentuk tabel.Sebagai *sample*, maka *hexadecimal* yang digunakan sebanyak 16 yaitu:

25	50	44	46	2D	31	2E	35
0A	25	E2	E3	CF	D3	0A	39

Setelah mendapatkan string sample maka akan dilakukan proses kompresi dengan algoritma *BWT* terlebih dahulu, langkah kompresi dengan algoritma *BWT* adalah sebagai berikut:

1. Pindahkan urutan depan ke urutan belakang sampai yang paling belakang menjadi urutan pertama.

Tabel 8. Tahap Pertama *BWT*

Index	Nilai															
1	25	50	44	46	2D	31	2E	35	0A	25	E2	E3	CF	D3	0A	39
2	50	44	46	2D	31	2E	35	0A	25	E2	E3	CF	D3	0A	39	25
3	44	46	2D	31	2E	35	0A	25	E2	E3	CF	D3	0A	39	25	50
4	46	2D	31	2E	35	0A	25	E2	E3	CF	D3	0A	39	25	50	44
5	2D	31	2E	35	0A	25	E2	E3	CF	D3	0A	39	25	50	44	46
6	31	2E	35	0A	25	E2	E3	CF	D3	0A	39	25	50	44	46	2D
7	2E	35	0A	25	E2	E3	CFD3		0A	39	25	50	44	46	2D	31
8	35	0A	25	E2	E3	CFD3		0A	39	25	50	44	46	2D	31	2E
9	0A	25	E2	E3	CFD3		0A	39	25	50	44	46	2D	31	2E	35
10	25	E2	E3	CFD3		0A	39	25	50	44	46	2D	31	2E	35	0A
11	E2	E3	CF	D3	0A	39	25	50	44	46	2D	31	2E	35	0A	25
12	E3	CF	D3	0A	39	25	50	44	46	2D	31	2E	35	0A	25	E2
13	CF	D3	0A	39	25	50	44	46	2D	31	2E	35	0A	25	E2	E3
14	D3	0A	39	25	50	44	46	2D	31	2E	35	0A	25	E2	E3	CF
15	0A	39	25	50	44	46	2D	31	2E	35	0A	25	E2	E3	CF	D3
16	39	25	50	44	46	2D	31	2E	35	0A	25	E2	E3	CF	D3	0A

2. Urutkan nilai terkecil hingga terbesar
   0A 0A 25 25 2D 2E 31 35 39 44 46 50 E2 E3 CF D3

Tabel 9. Langkah-langkah pengurutan nilai dengan algoritma *BWT*

Index	Nilai															
1	0A	25	E2	E3	CF	D3	0A	39	25	50	44	46	2D	31	2E	35
2	0A	39	25	50	44	46	2D	31	2E	35	0A	25	E2	E3	CF	D3
3	25	50	44	46	2D	31	2E	35	0A	25	E2	E3	CF	D3	0A	39
4	25	E2	E3	CF	D3	0A	39	25	50	44	46	2D	31	2E	35	0A
5	2D	31	2E	35	0A	25	E2	E3	CF	D3	0A	39	25	50	44	46
6	2E	35	0A	25	E2	E3	CF	D3	0A	39	25	50	44	46	2D	31
7	31	2E	35	0A	25	E2	E3	CF	D3	0A	39	25	50	44	46	2D

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 707-715

Index	Nilai															
8	35	0A	25	E2	E3	CF	D3	0A	39	25	50	44	46	2D	31	2E
9	39	25	50	44	46	2D	31	2E	35	0A	25	E2	E3	CF	D3	0A
10	44	46	2D	31	2E	35	0A	25	E2	E3	CF	D3	0A	39	25	50
11	46	2D	31	2E	35	0A	25	E2	E3	CF	D3	0A	39	25	50	44
12	50	44	46	2D	31	2E	35	0A	25	E2	E3	CF	D3	0A	39	25
13	E2	E3	CF	D3	0A	39	25	50	44	46	2D	31	2E	35	0A	25
14	E3	CF	D3	0A	39	25	50	44	46	2D	31	2E	35	0A	25	E2
15	CF	D3	0A	39	25	50	44	46	2D	31	2E	35	0A	25	E2	E3
16	D3	0A	39	25	50	44	46	2D	31	2E	35	0A	25	E2	E3	CF

3. Setelah diurutkan nilai tersebut maka ambil dari nilai yang paling ujung pada hasil *BWT* langkah ke dua, maka didapat string baru yaitu:

Tabel 10. Langkah-langkah pengurutan nilai akhir kompresi

35	D3	39	0A	46	31	2D	2E	0A	50	44	25	25	E2	E3	CF

Selanjutnya dijelaskan nilai *hexadecimal*setelah dikompresi pada tabel dibawah ini:

Tabel 11. nilai *hexadecimal*setelah dikompresi

Hex	Frequency	Binary	Bit	Bit x Frekuensi Bit
0A	2	00001010	8	16
25	2	00100101	8	16
2E	1	00101110	8	8
31	1	00110001	8	8
35	1	00110101	8	8
46	1	01000110	8	8
44	1	01000100	8	8
50	1	01010000	8	8
2D	1	00101101	8	8
D3	1	11010011	8	8
39	1	00111001	8	8
CF	1	11001111	8	8
E3	1	11100011	8	8
E2	1	11100010	8	8
Ukuran setelah dikompresi			128	

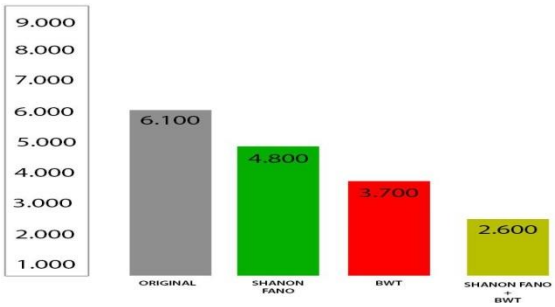
Berdasarkan tabel, maka ukuran akhir dari string adalah **128** bit.

3.2.5 Hasil Pengujian

Saat melakukan pengujian, penelitian ini menggunakan file dengan ekstensi
pdf sebagai contoh. Ujian dilakukan dengan satu file sampel yang berisi kertas. Pengujian berikut dilakukan dengan kombinasi Burrows Whellers Transform (BWT) dan algoritma Shanon Fano

3.2.6 Pengujian Rasio Kompres

Berikut dijelaskan rasio kompres dalam bentuk grafik:



Gambar 5. Grafik Kompresi

Tabel 12. Rasio Kompresi

	Rasio (%)			
Nama File Pdf	Ukuran File	Shanon Fano	Bwt	Bwt + Shanon Fano

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 707-715

Sampel Data Boy	6, 1 Mb	21,32	39,35	57,38

3.2.7 Pengujian Waktu Kompres

kompresi file teks pdf dengan kombinasi *algoritma Burrows Whellers Transform (BWT) dan Shanon Fano*.

Tabel 13. Pengujian Waktu Kompresi

	Waktu (Second)			
Nama File Pdf	Ukuran File	Shanon Fano	Bwt	Bwt + Shanon Fano
Sampel Data Boy	6, 1 Mb	30	55	50

3.2.8 Pengujian Waktu Dekompres

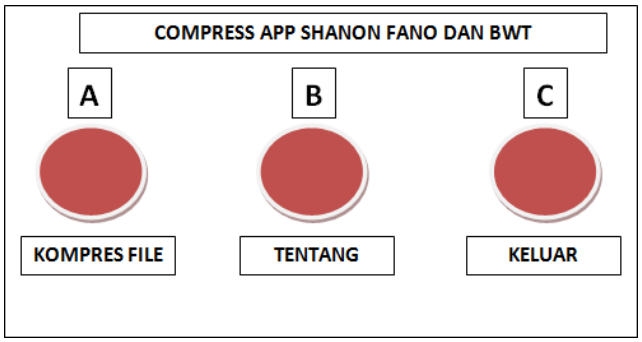
dekompresi file teks pdf dengan kombinasi *algoritma Burrows Whellers Transform (BWT) dan Shanon Fano*.

Tabel 14. pembobotan Alternatif

	Waktu (Second)			
Nama File Pdf	Ukuran File	Shanon Fano	Bwt	Bwt + Shanon Fano
Sampel Data Boy	6, 1 Mb	10	30	20

3.2.9 Perancangan Antarmuka (Interface)

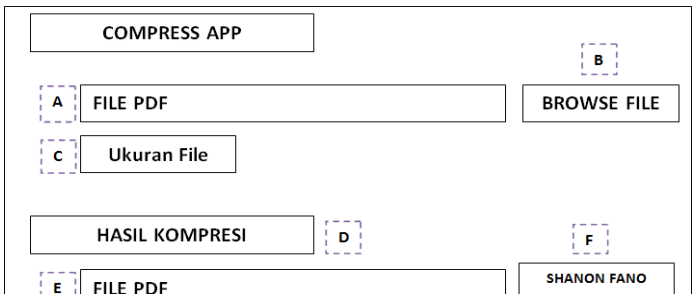
1. Rancangan Halaman Menu Utama
   Menu utama merupakan *form* utama dari sebuah aplikasi, dimana *form* ini nantinya digunakan untuk mengakses *sub-sub* menu pada aplikasi yang dibangun.



Gambar 6. Menu Utama

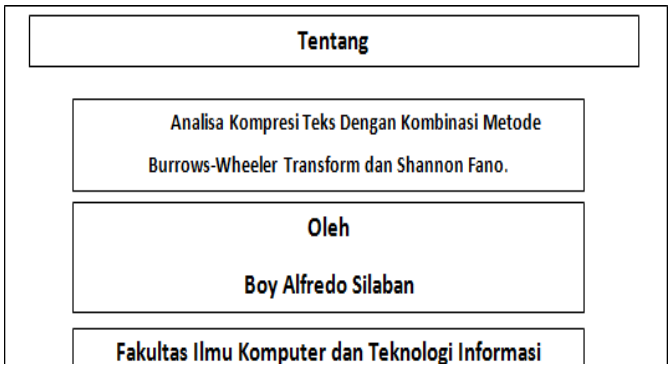
Keterangan:
a. *Menu* kompresi untuk menampilkan *form* kompresi.
b. *Menu* penulis untuk menampilkan *form* data diri penulis.
c. *Menu* keluar untuk keluar atau menutup aplikasi

2. Rancangan Form Kompres
   Form kompresi merupakan *interface* bagi pengguna *(user)* pada saat melakukan kompresi *file*. Pada form ini, pengguna bisa memilih *file* teks pdf yang akan dikompresi. Adapun *interface* dari *form* kompressi:



Gambar 7. Form KompresI

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 707-715



Gambar 7. *Form* Penulis

4. KESIMPULAN

Kesimpulan Kesimpulan yang diperoleh dari hasil penelitian ini ialah Kombinasi Metode Burrows-Wheeler Transform dan Shannon Fano sangat efisien untuk mengkompres file teks dengan ukuran yang lebih kecil dan kualitas file pdf yang lebih baik, Dari uji coba yang telah dilakukan, dengan melakukan pengkompresian Dengan Kombinasi Metode Burrows-Wheeler Transform dan Shannon Fano terhadap beberapa file teks pdf lebih bagus kualitasnya dan cenderung tidak merusak isi file teks pdf. Proses kompresi yang cenderung lebih singkat dan efisien untuk digunakan kalangan awam untuk keperluan tugas-tugas sekolah, kuliah bahkan pekerjaan.

REFRENCES

[1] Adhitama, G. 2009. Perbandingan Algoritma Huffman Dengan Algoritma Shannon-Fano. Institut Teknologi Bandung.
[2] Tri Rahmah Silviani, Ayu Arfiana, Teknik Kompresi Citra Menggunakan Metode Huffman. program pascasarjana. Universitas Negeri Yogyakarta.2016.
[3] Iwan Fitrianto Rahmad, Helmi Kurniawan, 2011, Kompresi File Citra Bitmap Menggunakan Algoritma RLE dan LZ78. Jurnal CSRID, 3(2):81a- 92.
[4] Alakuijala, H., Kliuchnikov, E., Szabadka, Z., & Vandevenne, L. 2015. Comparison of Brotli, Deflate, Zopfli, LZMA, LZHAM and Bzip2 Compression Algorithm.
[5] Alakuijala, J., & Szabadka, Z. 2016. Internet Engineering Task Force (IETF). Tersedia dalam: https://www.ietf.org/rfc/rfc7932.txt. [Diakses tanggal 11 November 2017]
[6] Sapta Aji Sri M, Linda Suvi R, Retno Sundari, meningkatkan rasio kompresi citra digital dengan Huffman coding pada transfer data. Jurnal Teknik Informatika. STMIK PPKIA Pradnya Paramita. Malang. 2014.
[7] Anggriani, M. 2011. Perbandingan Metode Kompresi Huffman dan Dynamic Markov Compression (DMC).
[8] Boedi, D., Rustamaji, H. C., & Nugraha, M. A. 2009. Aplikasi Kompresi SMS Berbasis JAVA ME Dengan Metode Kompresi LZW-Huffman. UPN "Veteran" Yogyakarta,. Dalam Seminar Nasional Informatika.
[9] Darmawan, D. 2014. Pengembangan E-Learning Teori dan Desain.
[10] Salomon, D. 2007. Data Compression The Complete Reference Fourth Edition, hal.2.
[11] Fransisca, C. 2014. Implementasi Algoritma Kompresi Deflate Pada Website Berbasis PHP dan Basis Data Mysql.
[12] Aji, Y. Y., Darwiyanto, E., & Septiana, G. (2016). Analisis Perbandingan Kompresi Dan Dekompresi Data Teks Menggunakan Algoritma Shannon-fano 2 Gram Dan Algoritma Lempel Ziv Welch Pada Terjemahan Hadits Shahih Muslim. eProceedings of Engineering,