KETIK : Jurnal Informatika
ISSN: 3026-183X (Media Online)
Vol. 01, No. 04, Maret 2024 Hal. 48-61
Faatuatua Media Karya
https://jurnal.faatuatua.com/index.php/KETIK
DOI: doi.org/10.70404/ketik.v1i04.62

Analisa Perbandingan Algoritma Huffman Dan Algoritma Taboo Code Pada Kompresi File Teks

Ameliorasi Daeli

Universitas Budi Darma, Indonesia, email: ameldaeli@gmail.com

Info Artikel

Diajukan: 13-03-2024
Diterima: 16-03-2024
Diterbitkan: 30-03-2024

Kata Kunci:
Perbandingan;
Algoritma;
Kompresi;
Huffman;
Taboo Code;
File Teks.

Keywords:
Comparison;
Algorithm;
Compression;
Huffman;
Taboo Code;
Text Files.

Abstrak

Perkembangan teknologi yang begitu pesat, Kompresi data sangat penting disebabkan pemanfaatan data dalam bentuk elektronik lebih cepat saat melakukan transmisi. Ukuran file yang besar menjadi masalah penting dalam proses pengolahan data, seperti membutuhkan banyak ruang penyimpanan dan membutuhkan waktu yang lama saat melakukan pertukaran data. Untuk mengatasi masalah tersebut maka perlu diterapkan yang namanya teknik kompresi. Adapaun algoritma yang digunakan dalam penelitian ini adalah algoritma huffman dan algoritma taboo code untuk mengkompresi file teks dan kemudian membandingkan kinerja kedua algoritma tersebut sesuai dengan parameter yang telah ditentukan. Setelah dilakukan implementasi dan pengujian sistem dapat diketahui bahwa taboo code memiliki kinerja yang lebih baik dibandingkan dengan algoritma Huffman dalam mengkompresi file teks, dengan menggunakan metode perbandingan exponential algoritma taboo code dengan total nilai 10,678 sedangkan algoritma Huffman 12,017, algoritma taboo code lebih efektif diterapkan dalam mengkompresi file teks karena semakin kecil total nilai yang didapat maka semakin sedikit proses kompresi algoritma tersebut.

Abstract

The rapid development of technology, data compression is very important due to the utilization of data in electronic form faster when transmitting. Large file sizes are an important problem in data processing, such as requiring a lot of storage space and taking a long time when exchanging data. To overcome this problem, it is necessary to apply a compression technique. The algorithms used in this research are the huffman algorithm and taboo code algorithm to compress text files and then compare the performance of the two algorithms according to predetermined parameters. After implementing and testing the system, it can be seen that taboo code has better performance than the Huffman algorithm in compressing text files, using the exponential comparison method of the taboo code algorithm with a total value of 10.678 while the Huffman algorithm is 12.017, the taboo code algorithm is more effectively applied in compressing text files because the smaller the total value obtained, the less the compression process of the algorithm.

1. PENDAHULUAN

Saat ini dalam dunia komputer teknik kompresi banyak di temukan dalam mendukung perkembangan teknologi, dan memiliki peran penting dalam melakukan pemecahan masalah, membuat pekerjaan lebih mudah dan efisien, salah satunya dalam mentransfer data atau sebaliknya bertukar data dalam media internet. Kompresi file merupakan suatu proses untuk mengubah atau mengurangi ukuran sebelum menyimpan atau memindahkan data tersebut kedalam media penyimpanan, dimana suatu data yang disimpan kedalam media penyimpanan maka semakin bertambah atau semakin berukuran besar, maka bisa jadi media penyimpanan tersebut tidak dapat menyimpan data melebihi kapasitas penyimpan. Selain masalah dalam proses penyimpanan, pemakaian data yang memiliki ukuran yang sangat besar juga memakan waktu dalam proses transmisi atau melakukan transfer sehingga membutuhkan waktu yang cukup lama dibandingkan dengan data yang ukuran lebih kecil, maka dalam ini dilakukan yang namanya kompresi terhadap file teks tersebut.

Beberapa algoritma yang digunakan dalam mengkompresi suatu data, namun semua algoritma memiliki kinerja yang tentunya berbeda- beda dan hasil kompresinya tentunya juga berbeda. Oleh karena itu penulis melakukan sebuah penelitian dengan melakukan kompresi file teks menggunakan dua algoritma.algoritma kompresi yang digunakan adalah algoritma huffman dan taboo code kedua algoritma ini sudah banyak diteliti dan digunakan kedalam berbagai objek seperti file video, file audio, file dokumen, dan lain sebagainya. Namun sampai saat ini belum ada penelitian yang membahas tentang algoritma Huffman untuk membandingkan hasil kompresinya dengan algoritma Taboo code.

Dalam penelitian ini, untuk mendapatkan hasil proses kompresi file teks yang baik dan efektif maka peneliti melakukan proses perbandingan algoritma. Proses perbandingan algoritma dilakukan dengan menggunakan metode ekponensial. Sehingga dengan melakukan perbandingan algoritman menggunakan metode eksponensial dapat membantu dalam memilih algoritma terbaik dengan cepat dan lebih jelas. Metode eksponensial lebih cenderung memberikan bobot yang lebih besar pada perbedaan kinerja yang signifika.

Berdasarkan penelitian yang dilakukan Moch Lazarudi Imani pada tahun 2021 tentang " Penerapan Metode Huffman Dalam Kompresi Data" dalam hasil penelitian yang dilakukan menyimpulkan bahwa metode huffman berhasil. mengkompresi hingga lebih dari 45% ukuran sebelum dikompresi, ratio pengujian memiliki nilai terendah 46,53% dan untuk nilai tertinggi 47,08% [1].

Pada penelitian sebelumnya yang dilakukan oleh Wahyu Pramusinto tahun 2019 tentang "Aplikasi Pengaman File Dengan Metode Kriptografi AES 192, RC4 Dan Metode Kompresi Huffman" dalam peneletiannya menyimpulkan didalam aplikasi keamanan dokumen ini, file penting dapat dirahasiakan karena disimpan dalam server dan hanya dapat diakses melalui aplikasi ini. Metode algoritma kompresi Huffman mengurangi ukuran file [2]

Penelitian terkait yang dilakukan oleh Saiful Simanjutak pada tahun 2022, dalam penelitiannya menyimpulkan dengan mengunakan algoritma Taboo codes untuk mengompresi file video bahwa hasil kompresi dengan prameter analisis kompresi rasio dan penghematan ruang dengan rasio kompresi 58,33% dan hasil space save 0,1467% menunjukan tingkat hasil kompresi yang sangat baik dalam sebuah file video dan sudah baik untuk diterapkan [3]

Berdasarkan penelitian yang dilakukan oleh Arianti Rhamadani yang berjudul "Analisa Perbandingan Algoritma Taboo Codes Dan Algoritma Yamoto's Code Recursive Code Untuk Kompresi File Teks Menggunakan Metode Exponential" memberikan kesimpulan dengan menerapkan metode perbandingan exponential dengan empat kriteria, hasil algoritma taboo codes memiliki total nilai 8.641 sedangkan algoritma Yamamoto's Recursive Code 7,203. Dengan hasil diatas menunujukkan algoritma Yamamoto's Recursive code jauh lebih baik dalam mengkompresi file teks[4].

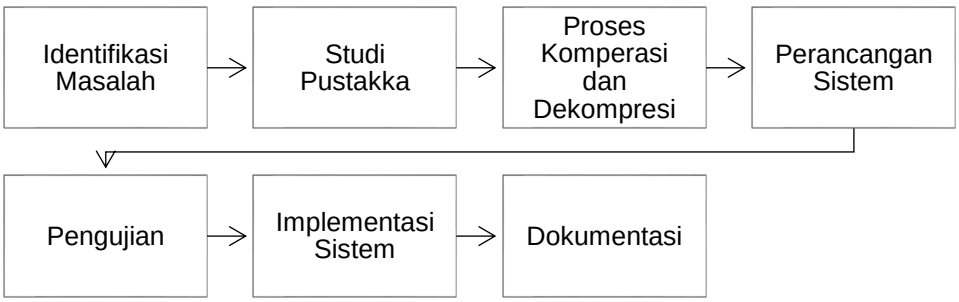
Pada penelitian terdahulu yang dilakukan oleh Abu Sani Tanjung, yang berjudul "Analisa Perbandingan Algoritma Huffman Dengan Algoritma Goldbach Codes Pada Kompresi File Teks Dengan Menggunakan Metode Perbandingan Eksponential" menyimpulkan dalam penerapan metode perbandingan Exponential Dengan empat kriteria, algoritma huffman jauh lebih baik hasilnya dalam mengkompresi file teks[5].

Dari latar belakang masalah yang telah dijelaskan sebelumnya, maka akan dilakukan sebuah penelitian yang nantinya diharapkan mampu menyelesaikan permasalahan yang terjadi tersebut dengan judul "Analisa Perbandingan Algoritma Huffman Dan Algoritma Taboo Code Pada Kompresi File Teks"

2. METODE PENELITIAN

2.1 Kerangka Kerja Penelitian

Metodologi penelitian berisikan tentang tahapan-tahapan dalam penyelesaian penelitian agar penelitian terstruktur, tahapan ini diperlukan untuk mempermudah dalam menyelesaikan penelitian agar sesuai dengan yang dinginkan. Berikut kerangka kerja dalam penelitian ini adalah sebagai



Gambar 1. Tahapan penelitian

1. Identifikasi Masalah
   Identifikasi masalah merupakan langkah awal yang dilakukan oleh penulis adalah mengidentifikasi masalah untuk menentukan rumusan masalah dalam suatu peneletian.

2. Studi Pustaka
   Studi pustaka yaitu metode yang dilakukan penulis untuk mengumpulkan data dan informasi sebagai bahan referensi terhadap objek yang akan diteliti, dari berbagai sumber seperti jurnal, buku, dan lain sebagainya yang berhubungan dengan penelitian saat ini.
3. Proses Kompresi dan Dekompresi
   Analisa proses kompresi dan dekompresi pada *file* teks berformat *dock* yang bertujuan untuk memampatkan suatu ukuan *file* teks.
   a. Algoritma *Huffman,* Proses kompresi dan dekompresi *file* teks dengan menggunakan algoritma *Huffman*
   b. Algoritma *Taboo code,* Proses kompresi dan dekompresi *file* teks dengan menggunakan algoritma *taboo code*
   c. Proses perbandingan yang akan dilakukan yaitu membandingkan algoritma *huffman* dan algoritma *taboo code*
4. Perancangan Sistem
   Pada tahapan selanjutnya yaitu tahapan perancangan sistem, penulis menggambarkan bagaimana rancangan sistem aplikasi kompresi *file* teks, agar mudah dipahami oleh pengguna.
   1. Implementasi sistem
      Pada tahapan implementasi bertujuan untuk dapat mengetahui apakah sistem aplikasi yang telah dirancang pada tahapan sebelumnya sudah sesuai dengan gamabaran yang diiiginkan.
   2. Pengujian
      Proses pengujian terhadap hasil dari kompresi *file* teks, bertujuan untuk mengetahui hasil akhir dari penelitian yang dilakukan.
   3. Dokumentasi
      Proses ini mengumpulkan data dengan menganalisis dokumen-dokumen pada penelitian sebelumnya, bertujuan untuk menjelaskan proses terbentuknya aplikasi.

2.2 Kompresi File Teks

1. Kompresi

Kompresi adalah proses pengkodean informasi, menggunakan bit atau informasi-*Bearingunit* yang lain yang lebih rendah dari pada representasi datayang tidak terkodekan dengan suatu sistem *encoding* tertentu. Proses kompresi merupakan proses meruduksi suatu data untuk menghasilkan representasi digital yang padat atau mampat (*Compact*) namun tetap terdapat mewakili kuantitas informasi yang terkandung pada data tersebut[6]. Kata lain kompresi dapat diartikan sebagai pemadatan, yaitu mengecilkan suatu data dan proses pengubahan sekumpulan data yang menjadi suatu bentuk kode untuk menghemat kebutuhan tempat penyimpanan dan mempercepat dalam melakukan transmisi data. Karena terlalu banyak metode dalam mengkompresi suatu data sehingga muncul ide yang beragam dan cocok untuk semua jenis data yang menghasilkan nilai kompresinya tentu juga tidak sama, karena tujuan dari kompresi memperkecil suatu ukuran *file* tanpa harus mengurangi data-data yang ada didalamnya.

Metode kompresi data terdapat dalam dua teknik dalam melakukan proses kompresi yakni:

1. *Lossless Compression* dalam metode kompresi ini untuk memampatkan data asli yang sama persis dengan data terkompresi, tidak ada informasi yang hilang selama melakukan kompresi, hasil kompesinya tetap sama dengan data asli.
2. *Lossy Compression* dalam metode ini, data dapat dikompresi tidak seperti kompresi data sebelumnya yang hanya perlu menggunakan ukuran *file* yang lebih kecil dari *lossless*, namun tidak dapat membuat data terkompresi agar sesuai dengan data asli.

Dalam suatu teknik yang digunakan untuk menangani kompresi menggunakan algoritma khusus terdapat tiga ukuran yang bisa digunakan dalam mengukur kualitas atau hasil kompresinya. Adapun tiga prameter tersebut yaitu:[7]

1. *Rasio Of Compression (Rc)*

Rasio Of Compression merupakan sebuah prameter pemampatan untuk membandingkan hasil data sebelum dikompresi dengan data sesudah dikompresi, perbandingan hasil kompresi adalah sebuah indikator untuk dapat melihat hasil dari suatu kompresi. Rumus dari *Rasio Of Compression* yaitu:

$$RC = \frac{ukurana\ data\ sebelum\ dikompresi}{ukuran\ data\ setelah\ dikompresi}$$

2. *Compression Ratio (Cr)*

Compression Ratio adalah presentase perbandingan antara data yang sudah dilakukan pengkompresan dengan data yang belum dikompresi. Rumus dari *Compression Ratio* yaitu:

$$C_R = \frac{ukuran\ data\ setelah\ dikompresi}{ukuran\ data\ sebelum\ dikompresi}$$

3. *Space Saving* (SS)

Merupakan jumlah penyimpanan yang disimpan setelah dikompresi. *Space Saving* adalah presentase perbedaan antara ukuran data sebelum dikompresi dandata setelah dikompresi. Adapun rumus dari *Space Saving* yaitu:

$$RC = \frac{ukuran\ data\ sebelum - ukuran\ data\ setelah}{ukuran\ data\ sebelum}$$

2. file teks

Teks adalah kumpulan dari karakter-karakter atau *string* yang menjadi satu kesatuan. Teks yang memuat banyak karakter didalamnya selalu menimbulkan masalah pada media penyimpanan dan kecepatan waktu pada saat melakukan transmisi data. *File* teks merupakan *file* yang berisikan informasi- informasi dalam bentuk teks. data yang berasal dari dokumen pengolahan kata, angka yang digunakan dalam perhitungan, nama dan alamat dalam basis data merupakan contoh masukan data teks yang terdiri dari karakter, angka dantandabaca[6]

3. Dekompresi

Dekompresi adalah sistem yang digunakan untuk mengembalikan sebuah *file* yang telah mengalami pemampatan (*Compression)* pada bentuk dan ukuran *file* aslinya sehingga isi dari *file* tersebut dapat di lihat kembali. *Lossy Compression* adalah jenis algoritma kompresi yang menghasilkan data yang terkompresi yang berkurang kualitas dari data aslinya. *Lossless Compression* adalah algoritma yang digunakan mengkodekan data secara sempurna, tanpa kehilangan informasi atau data yang hilang dalam pengkodeannya[8]

2.3 Algoritma

1. Algoritma huffman

Algoritma *huffman* adalah suatu algoritma kompresi tertua yang disusun oleh david Huffman pada tahun 1952, Algoritma tersebut digunakan untuk membuat kompresi jenis *lossy compression*, yaitu pemampatan data dimana tidak ada satu *byte* yang hilang sehingga data tersebut utuh dan disimpan sesuai dengan aslinya. Dasar pemikiran algoritma *Huffman* ini adalah bahwa setiap karakter ASCII biasanya diwakili oleh 8 bit. Didalam algoritma *huffman* ada beberapa tahapan proses kompresinya yaitu sebagai berikut.[9]

1. Menghitung banyaknya jenis karakter, dengan urutan dan jumlah dari masing-masing karakter yang terdapat dalam sebuah *file*
2. Susun setiap jenis karakter, dengan urutan jenis karakter yang jumlahnya lebihsedikit ke jumlah yang lebih banyak
3. Buat pohon biner, berdasarkan urutan karakter, berdasrakan jumlah karakterdan beri kode untuk tiap karakter.
4. Ganti data yang ada dengan kode bit berdasarkan pohon biner
5. Simpan jumlah bit untuk kode bit yang terbesar, jenis karakter yg diurutkan dari frekuensi keluarnya terbesar ke terkecil besar data yang sudah berubah menjadi kode bit sebagai data hasil kompresi.

2. Algoritma Taboo Code

Algoritma *taboo code* memiliki pendekatan terhadap panjangnya suatuvariabel. Algoritma *taboo code* di gagaskan oleh Steven Pegeon, dimana prinsip dari algoritma *taboo code* adalah untuk memilih bilangan positif n dipilih *user* mengembalikan pola n untuk mengindikasikan akhir dari kode tersebut. Ada dua jenis tipe yang ada dalam algoritma *tabaoo code* diantarnya sebagai berikut[3].

1. Blok *based* memiliki panjang berupa kelipatan dari n,. blok *based* algoritma *taboo code* dari *integer* adalah *string* dari n- bit *blocks*, dimana niliai n dipilih oleh *user* dan blok terakhir memiliki sedikit pola *taboo* yang tidak dapat muncul pada block lainnya, n bit block memiliki nilai 2, jika salah satu nilai dicadangkan, masing-masing sisa kode blok dapat memiliki salah satu yang ada pola bit 2,-1.
2. *Unconstrained* adalah tipe kedua yaitu panjang total kode ini tidak terbatas pada kelipatan, dan memiliki relasi terhadap nomor *fibonacc*i ke n.
   Rumus dari algoritma *taboo code* sebagai berikut :

$$\underset{i}{gn}(K) = \sum_{i}^{k}(2^n - 1)^i = \frac{[(2^n-1)^K-1](2^2-1)}{2^n-1}$$

Dengan keterangan sebagai berikut : gn(K) : Jumlah total kode dalam rentang kn: Jumlah blok bit

Tabel 1. Pola *Taboo Code*

M	Code	m	Code	m	Code	m	code
0	0100	4	011000	8	101100	12	01010100
1	1000	5	011100	9	110100	13	01011000
2	1100	6	100100	10	111000	14	01011100
3	010100	7	101000	11	111100	….	

3. Metode Perbandingan Exponontial

Metode perbandingan *exponential* adalah suatu metode untuk menentukan urutan prioritas alternatif dengan kriteria jamak. Teknik ini digunakan sebagai pembantu individu dalam mengambil keputusan untuk menggunakan rancang bangun model yang telah terdefinisi dengan baik pada tahap proses. Dalam motede perbandingan *exponential* ada beberapa tahapan yang harus dilakukan yaitu:[10]

1. Menyusun alternatif-alternatif yang keputusan yang akan dipilih
2. Menentukan kriteria atau perbandingan kriteria keputusan, yang penting untukdievaluasi.
3. Menentukan tingkat kepentingan dari setiap kriteria.
4. Keputusan atau pertimbangan kriteria.
5. Melakukan penilaian terhadap alternatif pada setiap kriteria.
6. Menghitung skor atau nilai total setiap alternatif.
7. Menetukan urutan prioritas keputusan didasarkan pada skor atau nilai total masing-masing alternatif.

Dalam menghitung dan membandingkan proses kompresi formulasi perhitungan skor untuk setiap alternatif dalam metode perbandingan *exponential* yaitu:

Total Nilai [(TN)] _i) = ∑_(j = 1)^m [([RK] _ij)TKKj]

Dengan keterangan:

TN_i = Total nilai alternatif ke-i

RK_{ij} = Derajat kepentingan kriteria ke-j pada keputusan ke-i

TKK_j = Derajat kepentingan kriteria keputusan ke-j; $TKK_j > 0$;

m = Jumlah kriteria keputusan

n = Jumlah piihan keputusan

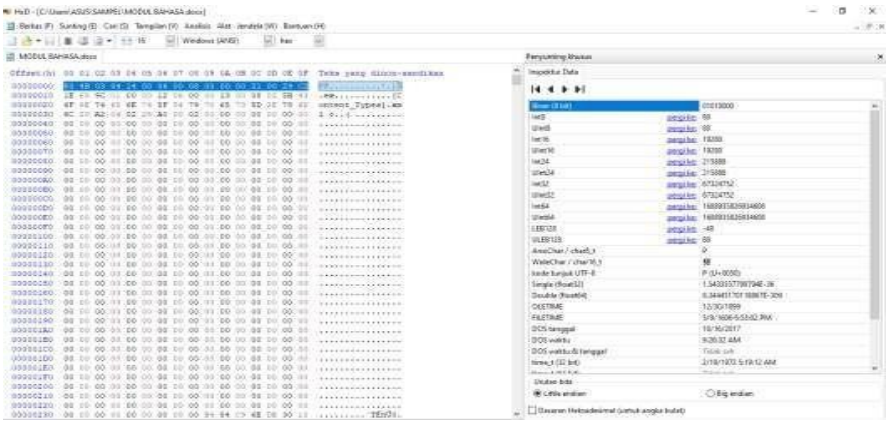
j = 1,2,3,4, 5,…m; m = jumlah kriteria

1, 2, 3….n; n = Jumlah pilihan alternatif

3. HASIL DAN ANALISIS

3.1 Analisa

Analisa masalah merupakakan tahapan awal dalam pengembangan sistem yang mendefinisikan proses sebab dan akibat dari dibangunnya sistem, agar sistem yang dibangun dapat bekerja dengan baik sesuai dengan tujuan sistem yangnantinya akan berfungsi untuk memperkecil suatu ukuran *file* teks dan menghemat ruang penyimpanan. Pada tahapan awal, analisa yang dilakukan penulis adalah memilih sebuah *file* teks yang akan digunakan untuk dikompresi adapun sampel datanya yaitu “MODUL BAHASA.dock”. Untuk mendapatkan nilai *hexadecimal* dari *file* tersebut dengan menggunakan *Software HxD Hex Editor*. Nilai *hexadicimal* yang telah didapat akan dilakukan proses kompresi. Adapaun nilai *hexadecimal* dari *file* “MODUL BAHASA.dock” berupa sampel data dapat dilihat pada gambar berikut:



Gambar 2. Nilai *Hexadecimal file* teks

3.2 Proses Kompresi dan Dekompresi Dengan Algoritma Huffman

Pada proses Penerapan algoritma *huffman* penulis hanya menggunakan 16 sampel nilai *hexadecimal* dari *file* teks "MODUL BAHASA.dock" nilai *hexadecimal* diambil dari sisi kanan atas sampai dengan bilangan yang ke 16, adapun nilai *hexadesimal* dapat dilihat pada tabel berikut :

Tabel 2. Nilai *hexadesimal*

Nama sampel data	Ukuran *file* teks	Nilai *hexadecimal*
MODUL BAHASA.dock	224 KB	**50 4B 03 04 14 00 06 00 08 00 00 00 21 00 29 C2**

Langkah selanjutnya ialah diurutkan nilai-nilai *hexadecimal* kedalam tabel dan mengelompokan nilai *hexadecimal* berdasarkan jumlah frekuensinya. Urutan nilai *hexadecimal* dan frekuensinya dapat dilihat pada tabel 3:

Tabel 3 Urutan Berdsarkan Frekuensinya

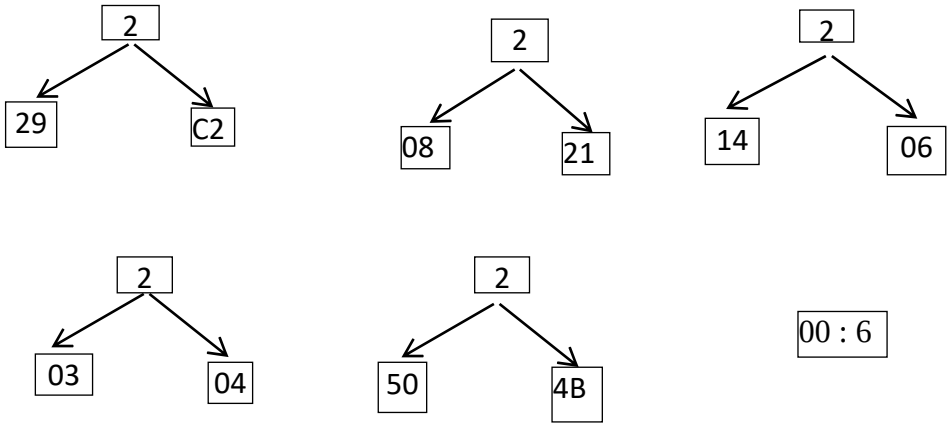
No	*Hexadecimal*	Frekuensi	*Decimal*	*Biner*	*Bit*	Frex x Bit
1	00	6	00	00000000	8	48
2	50	1	80	10010000	8	8
3	4B	1	75	01001011	8	8
4	03	1	03	00000011	8	8
5	04	1	04	00000100	8	8
6	14	1	20	00010100	8	8
7	06	1	06	00000110	8	8
8	08	1	08	00001000	8	8
9	21	1	33	00100001	8	8
10	29	1	41	00101001	8	8
11	C2	1	194	11000010	8	8
			Total bit			128

Adapun jumlah nilai *hexadecimal* pada *file* teks yaitu sebanyak 16 dengan total bit sebanyak 128. Selanjutnya tahapan proses kompresi awal dengan membentuk mengelompokkan berdasarkan jumlah frekuensi kemunculanterkecil,dapat dilihat pada tabel berikut :

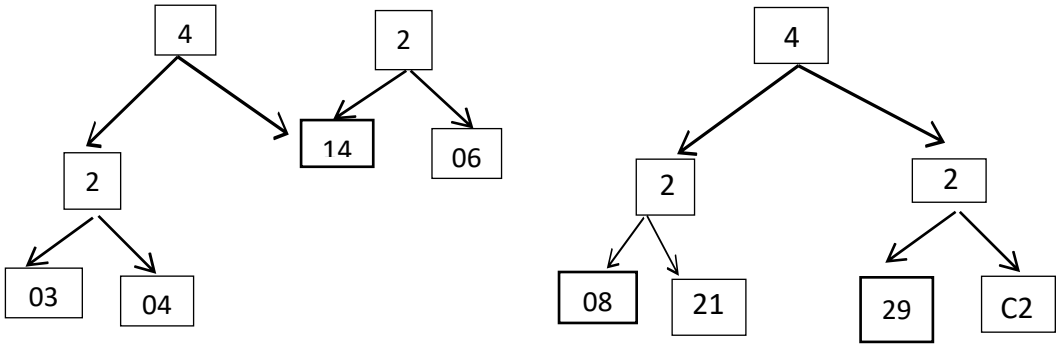
00 : 6	50 : 1	4B : 1	03 : 1	04 : 1

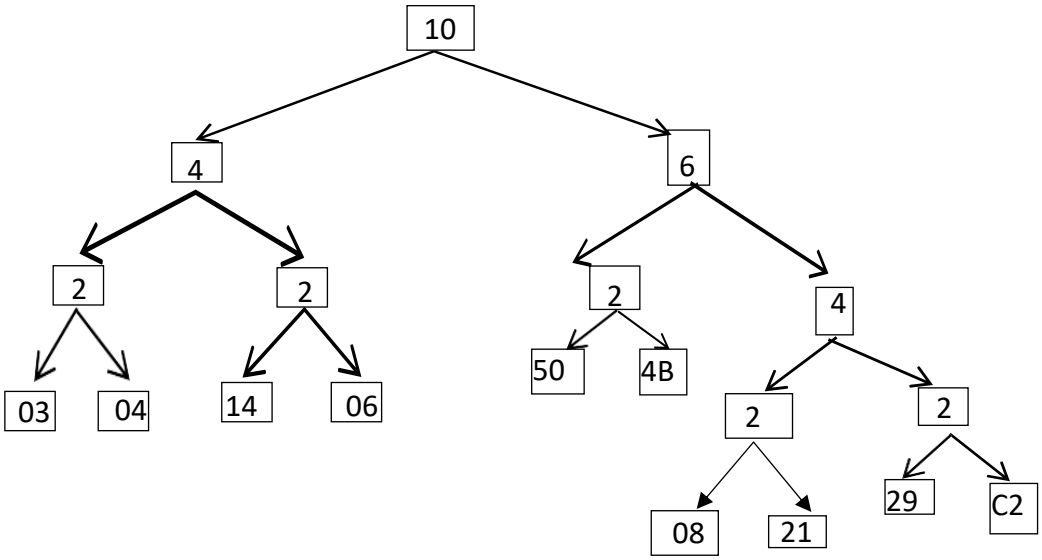
14 : 1	06 : 1	08 : 1	21 : 1	29 : 1	C2 : 1

Langkah selanjutnya gabungkanya nilai frekuensi terkecil

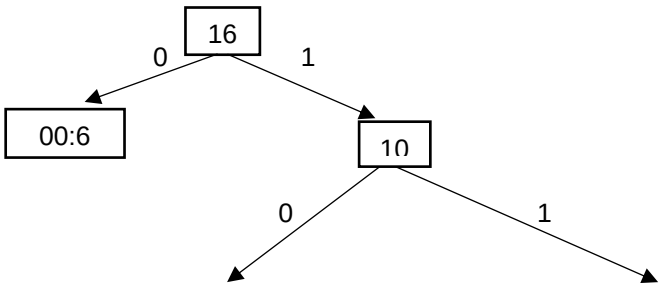


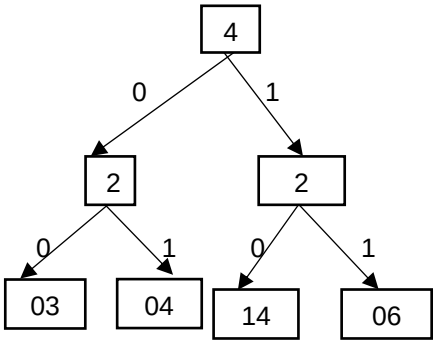
Langkah selanjutnya gabungkanya nilai frekuensi terkecil

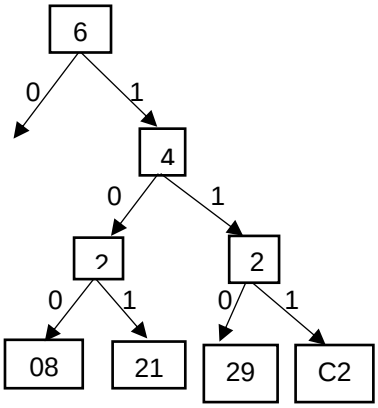




Langkah selanjutnya lakukan hal yang sama sehingga membentukmembentuk pohon *huffman.*







Tabel 4. Proses kompresi awal

n	*Hexadecimal*	Frekuensi	*Codeword*	Bit	Bit x Frek
1	00	6	0	1	6
2	50	1	1100	4	4
3	4B	1	1101	4	4
4	03	1	1000	4	4
5	04	1	1001	4	4
6	14	1	1010	4	4
7	06	1	1011	4	4
8	08	1	11100	5	5
9	21	1	11101	5	5
10	29	1	11110	5	5
11	C2	1	11111	5	5
Total Bit					50

Langkah berikutnya setelah kita melakukan tahapan kompresi awal dan mendapatkan total bit baru sebanyak 50 bit, setiap nilai *hexadecimal* menjadi *codeword*, dapat dilihat pada tabel 5:

Tabel 5. Susunan Bit Baru

No	*Hexadecimal*	Kode *Huffman*
1	50	1100
2	4B	1101
3	03	1000
4	04	1001
5	14	1010
6	00	0
7	06	1011
8	00	0
9	08	11100
10	00	0
11	00	0

Tabel 6. Lanjutan Susunan Bit Baru

No	Hexadesimal	Kode *Huffman*
12	00	0
13	21	11101
14	00	0
15	29	11110
16	C2	11111
Total bit		50

Proses selanjutnya adalah dengan menggabungkan nilai *string* bit baru yang telah didapatkan dengan susunan seperti berikut ini: "1100110110001001101001011011100000111010111101111" dengan total 50 bit. Karena 50 tidak habis dibagi menjadi 8 dan menyisahkan 2 atau dengan kata lain 50 Mod 8 = 2. Selanjutnya dengan melakukan penambahan bit baru dengan cara membentuk *padding* dan *flagging* dapat dilihat pada tabel berikut:

Tabel 7. Penambahan *Padding* dan *Flagging*

Padding	*Flagging*
50 Mod 8 = 2 7 – 2 + "1" 7 – 2 + "1" = **000001**	9 – n 9 – 2 = 7 Biner dari 7 = **00000111**

Jadi dengan penambahan *padding* dan *flagging* maka susunan bit baru: "1100110110001001101001011011100000111010111101111**00000100000111**" dengan total bit baru yaitu 64. Langkah selanjutnya membagi *sring* bitmenjadi per 8 bit, dapat dilihat pada tabel berikut :

Tabel 8. Penglempokan *String biner* baru

11001101	10001001	10100101	00111000
00111010	11110111	11000001	00000111

Proses selanjutnya ialah merubah kelompok *biner* bit kedalam bentukkarakter dengan cara merubah *biner* tersebut kedalam bentuk *decimal* dan *hexadecimal* untuk mendapatkan karakter dari bit tersebut. Dapat dilihat pada tabel 9 dibawah ini:

Tabel 9. Bentuk karakter dari *biner*

No	*String bit*	*Decimal*	*Hexadecimal*	Karakter
1	11001101	205	CD	Í
2	10001001	265	109	
3	10100101	165	A5	¥
4	00111000	56	38	8
5	00111010	58	3A	:
6	11110111	247	F7	÷
7	11000001	193	C1	Á
8	00000111	7	07	BEL

Hasil dari proses kompresi *file* teks dengan nilai *hexadesimal* "50 4B 03 04 14 00 06 00 08 00 00 00 21 00 29 C2" dengan menggunakan algoritma *huffman* adalah :

Gambar 3. Karakter hasil kompresi agoritma *Huffman*

Dari hasil kompresi diatas dengan menggunakan metode *huffman* untuk dapat melihat kinerjanya dapat dilakukan dengan menerapkan parameter yang ltelah ditentukan sebagai berikut:

a. *Ratio Of Compression* (Rc)

$$RC = \frac{ukuran\ sebelum\ dikompresi}{ukuran\ setelah\ dikompresi}$$

$$RC = \frac{128}{50} = 2{,}56$$

b. *Compression ratio* (Cr)

$$Cr = \frac{ukuran\ setelah\ dikompresi}{ukuran\ sebelum\ dikompresi}$$

$$Cr = \frac{50}{128}\ X\ 100\%$$

$$Cr = 0{,}39\ X\ 100\% = 39\%$$

c. *Space Saving* (Ss)

$$Ss = \frac{ukuran\ setelah\ dikompresi}{ukuran\ sebelum\ dikompresi}\ X\ 100\%$$

$$Ss = 1\frac{50}{128}\ X\ 100\%$$

$$Ss = 61\%$$

Setelah kita menghitung *Ratio Of Compression* (Rc), *Compression ratio* (Cr), dan *Space Saving* (Ss), maka dapat kita ketahui hasil ukuran *file* terkompresi, dapat dilihat pada tabel berikut:

Tabel 10. Ukuran *file* terkompresi menggunakan algoritma *Hufmman*

Nama *file* terkompresi	Ukuran *file* terkompresi
MODUL BAHASA.dock	136 KB

Proses dekompresi dengan algoritma *huffman* dengan tujuan mengembalikan *file* teks kebentuk awal, dapat dilihat pada tabel dibawah ini:

Tabel 11. Proses Awal dekompresi

No	Karakter	*Decimal*	*String* bit
1	Í	205	11001101
2		265	10001001
3	¥	165	10100101
4	8	56	00111000
5	:	58	00111010
6	÷	247	11110111
7	Á	193	11000001
8	BEL	7	00000111

Setelah merubah karakter menjadi *string* bit, lalu gabungkan seluruh*string* bit yang telah didapat seperti berikut ini: "1100110110001001101001011011100000111010111101111**00000100000111**" kemudian menghilangkan *padding* dan *flagging* dengan caramengambil 8 bit terakhir seperti beikut ini:
n = **00000111**= 7

7 + n = 7 + 7 = 14

Langkah selanjutnya yaitu hilangkan *string* bit sebanyak 14 bit terakhir,maka sisa *stirng* bit yang didapakan ialah: "1100110110001001101001011011100000111010111011111" kemudian baca *string* bit sesuaikan tabel *codeword* sampai menemukan sebuah karakter,proses dapat dilihat pada tabel 4.11 dibawah ini:

Tabel 12. Pengecekan *Biner* Dekompresi

No	Nilai Biner	Keterangan	*Hexadecimal*
1	1	TIDAK ADA	
2	11	TIDAK ADA	
3	110	TIDAK ADA	
4	1100	ADA	50
5	1	TIDAK ADA	
6	11	TIDAK ADA	
7	110	TIDAK ADA	

8	1101	ADA	4B
9	1	TIDAK ADA	
10	10	TIDAK ADA	
11	100	TIDAK ADA	
12	1000	ADA	03
13	1	TIDAK ADA	
14	10	TIDAK ADA	
15	100	TIDAK ADA	
16	1001	ADA	04

Tabel. 13. Lanjutan pengecekan biner dekompresi

No	Nilai Biner	Keterangan	*Hexadecimal*
17	1	TIDAK ADA	
18	10	TIDAK ADA	
19	101	TIDAK ADA	
20	1010	ADA	14
21	0	ADA	00
22	1	TIDAK ADA	
23	11	TIDAK ADA	
24	101	TIDAK ADA	
25	1011	ADA	06
26	010	TIDAK ADA	
27	0101	TIDAK ADA	
28	01011	ADA	06
29	0	ADA	00
30	1	TIDAK ADA	
31	11	TIDAK ADA	
32	111	TIDAK ADA	
33	1110	TIDAK ADA	
34	11100	TIDAK ADA	
35	00001	TIDAK ADA	
36	000011	ADA	08
37	0	ADA	00
38	0	ADA	00
39	0	ADA	00
40	1	TIDAK ADA	
41	11	TIDAK ADA	
42	111	TIDAK ADA	
43	1110	TIDAK ADA	
44	11101	ADA	21

Tabel 14. Lanjutan pengecekan biner dekompresi

No	Nilai Biner	Keterangan	*Hexadecimal*
45	0	ADA	00
46	1	TIDAK ADA	
47	11	TIDAK ADA	
48	111	TIDAK ADA	
49	1111	TIDAK ADA	
50	11110	ADA	29
51	1	TIDAK ADA	
52	11	TIDAK ADA	
53	111	TIDAK ADA	
54	1111	TIDAK ADA	
55	11111	TIDAK ADA	C2

Hasil dari dekompresi sesuai dengan string bit semula yaitu dengan bilangan hexadecimal “50 4B 03 04 14 00 06 00 08 00 00 00 21 00 29 C2”.

3.3 Metode Perbandingan Exponential

Metode perbandingan *exponential* dalam melakukan perhitungnan ataumembandingkan proses kerja kedua algoritma tersebut ada beberapa tahapan yang harus dilakukan yaitu:

1. Menentukan alternatif
   Dengan membandingkan algoritma *huffman* dan *taboo code* pada saat melakukan kompresi, perlu menetukan algoritma mana yang harus digunakan.
2. Menetukan kriteria
   Perlu menentukan y kriteria sebagai perbandingan keputusan untuk dievaluasi,kriteria dapat dilihat dalam tabel 4.26 beikut:

Tabel 15. Menentukan Kriteria

No	Kriteria	Keterangan
1	*Ratio of Compression(RC)*	Untuk mebandingkan hasil data sebelum dikompresi dengan data yang sudah dikompresi
2	*Compression Ratio (CR)*	Perbandingan antara data yang sudah dikompresi dengan datayang belum dikompresi
3	*Space Saving (SS)*	Presentase perbedaan antara ukuran data sebelum dikompresi dan setelah dikompresi

3. Menentukan bobot kriteria
   Pemberian bobot setiap kriteria untuk menentukan tingkat kepentingan dari setiap kriteria keputusan, pembobotan kriteria dapat dilihat pada tabel berikut:

Tabel 16. Pembobotan kriteria

No	Kriteria	Bobot range (0-5)	Presentase pengaruh kriteria
1	*Ratio Of Compression (Rc)*	0.2	20%
2	*Compression Ratio (Cr)*	0.3	30%
3	*Space Saving (SV)*	0.5	50%

4. Penilaian terhadap kriteria
   Pemberian nilai pada setiap kriteria yang telah ditentukan, nilai didapat berdasarkan analisa algoritma *huffman* dan algoritma *taboo code*, dapat dilihat pada tabel berikut ini:

Tabel 17. Pemberian nilai kriteria

Alternatif	Kriteria				
	RC		CR		SS
	SB	SD	SB	SD	
Hufmman	128 (bit)	50 (bit)	50 (bit)	128 (bit)	1 – CR * 100%
Taboo Code	128 (bit)	80 (bit)	80 (bit)	128 (bit)	1 – CR * 100%

5. Menghitung Skor
   Langkah selanjutnya dengan melakukan perhitungan menggunakan rumusmetode perbandingan *exponential*.

Tabel 18. Perhitungn Metode perbandingan *Exponential*

Kriteria	Bobot	Algoritma *huffman*	Algoritma T*aboo code*
RC	0.2	2,56	1,6
CR	0.3	39	62,5
SS	0.5	61	37,5

a. Algoritma *Huffman*

$$= (2{,}56)^{0{,}2} + (39)^{0{,}3} + (61)^{0.5}$$

$$= 1{,}206 + 3.001 + 7{,}810$$

$$= 12{,}017$$

b. Algoritma *Taboo Code*

$$= (1{,}6)^{0{,}2} + (62{,}5)^{0{,}3} + (37{,}5)^{0.5}$$

$$= 1{,}098 + 3.457 + 6{,}123$$

$$= 10{,}678$$

6. Menentukan hasil keputusan
   Menentukan prioritas keputusan berdasarkan nilai dari masing-masing alternatif. Hasil penentu prioritas keputusan dapat dilihat pada tabel 19 :

Tabel 19. Prioritas Keputusan

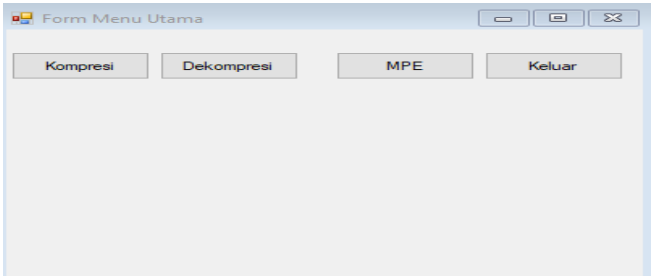
Alternatif	Total Nilai	Rangking
Algoritma *Huffman*	12,017	2
Algoritma *Taboo Code*	10,678	1

Berdasarkan pada perhitungan menggunakan metode perbandingan *exponential* dapat diketahui nilai dari kedua algoritma tersebut, algoritma taboo code merupakan algoritma yang efesien digunakan, karena semakin kecil nilai yang diperoleh maka semakin sedikit proses kompresi algoritma tersebut.

3.4. Pengujian Sistem

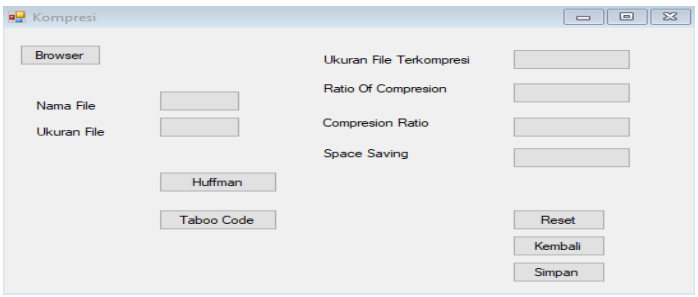
Pengujian Sistem yang telah dihasilkan tampilannya pada perangkat lunak dapat dilihat pada tampilan form gambar berikut ini:

1. Tampilan Form menu utama



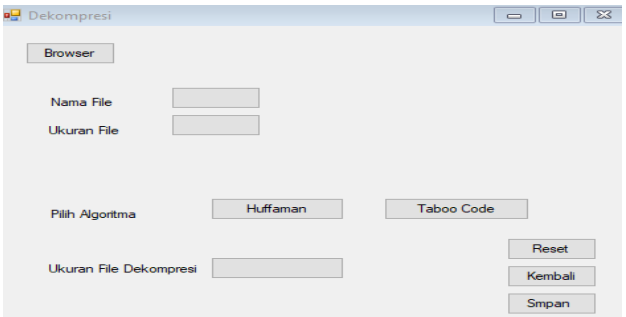
Gambar: 4. Tampilan Form Menu Utama

2. Tampilan Form Menu Kompresi



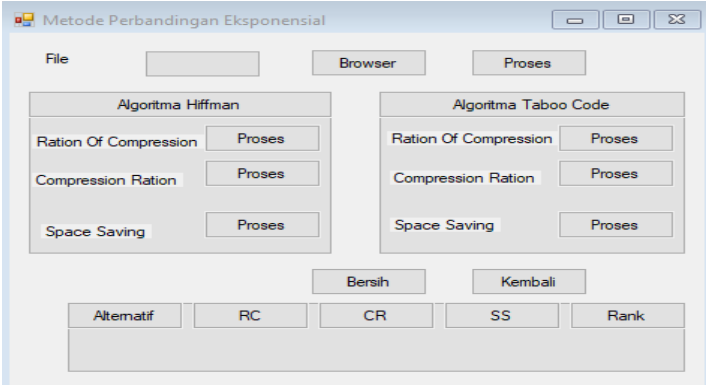
Gambar 5. Tampilan Form Menu Kompresi

3. Tampilan Form Menu Dekompresi



Gambar 6. Tampilan Form Menu Dekompresi

4. Tampilan Form Menu Metode Perbandingan



Gambar 7. Form Perbandingan Metode Eksponensial

3.5 Hasil Pengujian

Hasil dari pengujian setelah dibandingkan kedua algoritma dengan menggunakan perbandingan metode Eksponensial dapat dilihat pada tabel dibawah ini.

Tabel 20. Hasil Pengujian

Alternatif	Total Nilai	Rangking
Algoritma *Huffman*	12,017	2
Algoritma *Taboo Code*	10,678	1

4. KESIMPULAN

Berdasarkan pembahasan dan evaluasi dari bab-bab sebelumnya dan analisis terhadap kompresi file teks, maka dapat disimpulkan sebagai berikut: (1) Penerapkan algoritma Huffman dan algoritma Taboo Code, dalam mengkompresi file teks termasuk kategori baik, karena hasilnya tidak ada pengurangan pada isi file nya. (2) Aplikasi yang dirancang melakukan kompresi file Teks dengan menggunakan algoritma Huffman dengan algoritma Taboo Code, dan melakukan dekompresi terhadap hasil kompresi dengan Huffman dan algoritma Taboo Code menjadi file asli. (3) Dari hasil perbandingan metode Eksponensial hasil yang paling tinggi dari kedua algoritma yang dibandingkan adalah algoritma Taboo Code, dimana algoritma Huffman memiliki nilai 12,017 sedangkan algoritma Taboo Code memiliki nilai 10,678.

REFERENSI

[1] M. L. Imani, R. R. Muhima, S. Agustini, T. Informatika, I. Teknologi, and
[2] A. Tama, “Page 1 Penerapan Metode Huffman dalam Kompresi Data,” pp. 1–7, 2021.
[3] A. Pengamanan, F. Dengan, M. Kriptografi, and R. C. Dan, “JURNAL BIT,” pp. 1–8, 2019.
[4] S. Simanjuntak, “Implementasi Metode Taboo Code Untuk Kompresi File Video,” pp. 4–11, 2022.
[5] K. Konferensi, N. Teknologi, and A. Rhamadani, “Analisa Perbandingan Algoritma Taboo Codes Dan Algoritma Yamamoto ’ s Recursive Code Untuk Kompresi File Teks Menggunakan Metode Exponential,” pp. 1–12, 2022, doi: 10.30865/komik.v6i1.5728.
[6] A. S. Tanjung and S. D. Nasution, “Comparison Analysis with Huffman Algorithm and Goldbach Codes Algorithm in File Compression Text Using the Method Exponential Comparison,” pp. 1–7, 2020.
[7] L. V Simanjuntak, Perbandingan Algoritma Elias Delta Code dengan Levenstein Untuk Kompresi File Teks, vol. 1, no. 3. 2020.
[8] A. A. L. Tobing, “Implementasi algoritma goldbach code g1 dalam mengkompresi file dokumen,” 2022.
[9] R. N. Ibrahim, Perbandingan Kompresi File Menggunakan Algoritma Run Length Dengan Two Level Hoshing, vol. 1, no. 2. 2007.
[10] A. Lempel, J. Ziv, and T. Welch, “ANALISIS PERBANDINGAN ALGORITMA HUFFMAN DENGAN ALGORITMA (LEMPEL-ZIP- WELCH) PADA KOMPRESI GAMBAR MENGGUNAKAN METODE EXPONENSIAL,” pp. 21–25, 1984.
[11] Y. Devianto and S. Dwiasnati, “Aplikasi Pengambilan Keputusan Indeks Kepuasaan Masyarakat Dengan Metode Perbandingan Eksponensial (MPE) Pada Unit Pelayanan Masyarakat Dengan Alat Microcontroller Sebagai Alat Bantu Survey,” J. Ilm. FIFO, vol. 10, no. 1, p. 13, 2018 .