Jurnal Sains Dan Teknologi Informasi
Vol 3, No 3, Juni 2024, Hal 119-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v3i3.7694
Website https://ejurnal.seminar-id.com/index.php/jussi

Analisa Perbandingan Algoritma *Unary Code* Dan Algoritma *Prefix Code* Untuk Kompresi *File* PDF

Apni Radiah

Fakultas Ilmu Komputer dan Teknologi Informasi, Program Studi Teknik Informatika, Universitas Budi Darma, Medan, Indonesia
Jalan Sisingamanganraja No. 338, Medan, Sumatera Utara, Indonesia
Email: apni1802@gmail.com
(* : Coressponding Author)

Abstrak- *File* pdf (*Portable Document Format*) adalah jenis format dokumen atau berkas untuk keperluan pertukaran dokumen digital. Masalah pada *file* pdf sering kali terjadi pada saat kita mengirim *file* dengan kapasitas yang berukuran besar sehingga dalam proses mengirim *file* mendapatkan pemborosan dan akan memakan waktu yang lama dalam proses pengiriman *file* pdf. Dalam mengatasi masalah tersebut adalah harus malakukan teknik kompresi terlebih dahulu agar *file* pdf tersebut menghemat dalam pengiriman *file*. Sesudah melakukan proses kompresi, maka akan mendapatkan *file* yang berukuran lebih kecil dari semula. Dan tujuannya agar menghemat ruang penyimpanan dan juga mempercepat proses pengirimannya. Dalam proses pengkompresiannya, Algoritma *Prefix Code* lebih bagus hasil pengkompresiannya dari pada Algoritma *Unary Code* yaitu Algoritma *Prefix Code* memperoleh total nilai sebesar 5,420% sedangkan Algoritma *Unary Code* memperoleh total nilai 5,220% dalam mengkompresi *file* Pdf. Algoritma *Prefix Code* memperoleh hasil lebih besar dibandingkan dengan algoritma *Unary Code.* Maka kesimpulannya yaitu Algoritma *Prefix* lebih bagus dari pada Algoritma *Unary Code* untuk mengkompresi *file* Pdf.

Kata kunci: Kompresi; Perbandingan; Algoritma; Unary; Prefix; Code; Pdf.

Abstract- *A pdf file (Portable Document Format) is a type of document or file format for digital document exchange purposes. Problems with pdf files often occur when we send files with a large capacity so that the process of sending files is wasteful and it will take a long time in the process of sending pdf files. In overcoming this problem, you have to do a compression technique first so that the pdf file saves on sending files. After carrying out the compression process, you will get a smaller file than before. And the goal is to save storage space and also speed up the shipping process.In the compression process, the Prefix Code Algorithm has better compression results than the Unary Code Algorithm, namely the Prefix Code Algorithm obtains a total value of 5.420% while the Unary Code Algorithm obtains a total value of 5.220% in compressing Pdf files. The Prefix Code algorithm obtains greater results compared to the Unary Code algorithm. So the conclusion is that the Prefix Algorithm is better than the Unary Code Algorithm for compressing Pdf files.*

Keywords*: Compression; Comparison; Algorithm; Unary; Prefix; Code; Pdf.*

1. PENDAHULUAN

File pdf (Portable Document Format) adalah jenis format dokumen untuk keperluan pertukaran dokumen digital, yang sering ditemukan diberbagai dokumen dalam kehidupan sehari-hari seperti pendidikan atau pekerjaan yang digunakan oleh mahasiswa atau pegawai dalam perusahaan agar pekerjaannya menjadi lebih mudah. File pdf juga sering digunakan untuk mempresentasikan dokumen dua dimensi seperti grafik vektor dan citra. Kelebihan file PDF adalah bahwa format file PDF tidak hanya dapat menyimpan data teks tetapi juga bisa menyimpan gambar dan foto. Saat menyimpan sebagai file PDF, data yang disimpan lebih kecil dan kemungkinannya tidak mudah terkena oleh virus [1]. Masalah pada file pdf sering kali terjadi pada saat kita mengirim file dengan kapasitas yang berukuran besar sehingga dalam proses mengirim file mendapatkan pemborosan dan akan memiliki waktu yang lama dalam proses pengiriman file pdf. Solusi dalam mengatasi masalah tersebut adalah dengan melakukan teknik kompresi terlebih dahulu agar file pdf tersebut menghemat dalam pengiriman file. Sesudah melakukan proses kompresi, maka akan mendapatkan file yang berukuran lebih kecil dari semula, dan tujuannya agar menghemat ruang penyimpanan dan juga mempercepat proses pengirimannya.

Kompresi adalah proses mereduksi data yang besar menjadi ukuran yang lebih kecil dari ukuran aslinya. Teknik kompresi ini adalah mengganti karakter yang berulang dengan poin tertentu untuk memperkecil ukuran data [1]. Oleh karena itu data yang berukuran besar yang ingin disimpan perlu dikompres terlebih dahulu supaya ukurannya menjadi lebih kecil. Data yang dimaksud misalnya seperti file pdf (Portable Document Format) yang dikompresi untuk mengurangi ukuran file secara Pribadi. Ruang penyimpanan yang terbatas sering menyebabkan sedikitnya file yang dapat disimpan, sehingga membuat orang-orang berpikir bagaimana cara mengkompresi file pdf agar dapat mengurangi ukuran file yang disimpan dengan melakukan teknik kompresi pada file. Beberapa aplikasi kompresi yang digunakan memiliki hasil yang berbeda-beda, dan kinerja berbeda, hal ini disebabkan pengaruh oleh algoritma yang digunakan untuk mengkompresi dengan mengurangi jumlah bit dari data aslinya. Dalam kompresi terdapat banyak algoritma-algoritma yang digunakan diantaranya adalah algoritma Unary Code dan Algoritma Prefix Code.

Berdasarkan hasil penelitian dari Soumi Rohmah Saragih dengan judul Penerapan Algoritma Prefix Code Dalam Kompresi Data Teks, tahun 2020 pada Jurnal KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer), dalam kompresi data teks dengan Algoritma Prefix Code maka disimpulkan Algoritma Prefix Code cocok untuk dalam mengompresi data teks [2]. Menurut Muhammad Alfarizi, pada KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer) 2020 berjudul Penerapan Algoritma Prefix Code Dalam Kompresi File Video, setelah melakukan penelitian tentang kompresi file video menggunakan Algoritma Prefix Code, Algoritma Prefix code cocok digunakan untuk

Jurnal Sains Dan Teknologi Informasi
Vol 3, No 3, Juni 2024, Hal 119-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v3i3.7694
Website https://ejurnal.seminar-id.com/index.php/jussi

mengkompresi file video. Hasil aplikasi kompresi video menunjukkan video yang diproses oleh proses kompresi atau dekompresi dan disimpan di driver komputer [3].

Menurut Mesiria Sitorus dengan judul Penerapan Algoritma Unary coding Pada Aplikasi Kompresi Short Messsage Service (SMS), tahun 2020 pada jurnal Journal of Computer System and Informatics (JoSYC) menyimpulkan bahwa penerapan algoritma unary code berhasil mengompresi SMS dengan baik, dan algoritma unary code membaca teks untuk dikompres, mencari teks yang sama, menghindari pengulangan karakter yang sama, dan menyimpan teks. Kompresi SMS dapat dirancang dan dibangun menggunakan aplikasi Android [4].

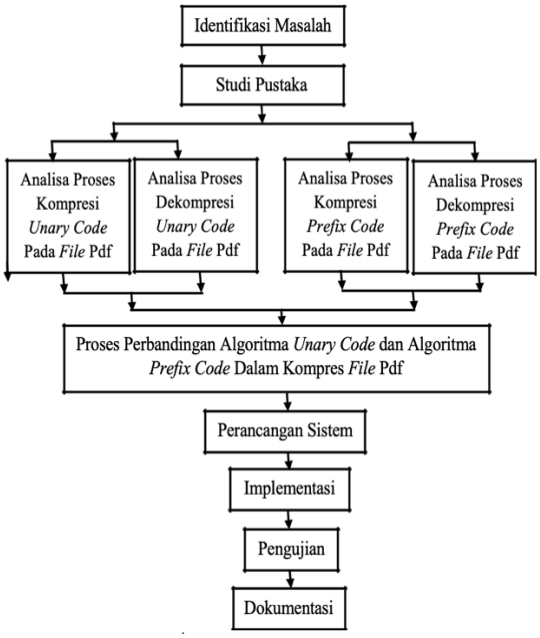
Berdasarkan hasil penelitian Winda Aprianti Harahap pada tahun 2017 dengan judul Studi Komparasi Kinerja Algoritma Goldbach G1 Code dan Algoritma Unary Codes Dalam Kompresi Teks, Kompresi file teks dengan 1 variasi karakter menggunakan rasio kompresi dan parameter menunjukkan bahwa algoritma unary code lebih baik dari algoritma goldbach G1 code dengan rasio kompresi rata-rata 31,44 bit. Waktu kompresi real-time dipengaruhi oleh jumlah karakter dan variasi jumlah karakter, tetapi semakin besar jumlah karakter dan variasi jumlah karakter, semakin lama waktu kompresi [5].

Berdasarkan penelitian terdahulu algoritma unary code memiliki hasil yang efektif dalam mengkomprsi file dokumen. Dimana menghasilkan byte data berkurang hingga 60% sehingga dapat menghemat kapasitas ruang penyimpanan serta mempercepat proses transfer data, sedangkan algoritma prefix code menunjukan bahwa kurang efektif dalam mengkompresi file dokumen karena menghasilkan byte data yang lebih besar dari algoritma unary code. Maka dalam penelitian ini penulis akan melakukan perbandingan hasil kompresi dari algoritma unary code dan algoritma prefix code untuk mengkomresi file pdf, yang dimana untuk mengetahui algoritma mana yang lebih efektif untuk mengkompresi serta mengurangi bytes data dari file pdf sehingga menghemat penggunaan ruang penyimpanan dan mempercepat proses transfer data. Maka dari itu penulis bermaksud melakukan penelitian untuk menggambarkan serta menguraikan langkah-langkah dan solusi yang akan dituangkan kedalam penelitian dengan mengangkat judul "Analisa Perbandingan Algoritma Unary Code dan Algoritma Prefix Code Untuk Kompresi File Pdf"

2. METODOLOGI PENELITIAN

2.1 Metodologi Penelitian

Kerangka kerja penelitian ialah suatu bentuk kerangka kerja yang dapat digunakan sebagai pendekatan untuk pemecahan masalah. Kerangka kerja ini merupakan prosedur yang akan diambil untuk memecahkan masalah yang dibahas. Adapun kerangka kerja penelitian dapat digambarkan pada gambar 1 berikut.



Gambar 1. Kerangka Penelitian

Berdasarkan kerangka penelitian yang telah diuraikan diatas, maka pembahasan setiap tahapan penelitian dapat diuraikan sebagai berikut:

a. Identifikasi Masalah
   Identifikasi masalah dilakukan untuk mendapatkan algoritma kompresi mana yang lebih baik diantara algoritma unary code dan algoritma prefix code yang akan digunakan untuk mengompresi ukuran file pdf.
b. Studi Pustaka
   Studi pustaka dilakukan untuk mencari data dalam buku, jurnal dan mempelajari sumber sumber bacaan yang dapat memberikan informasi terkait masalah yang sedang diteliti.

Jurnal Sains Dan Teknologi Informasi
Vol 3, No 3, Juni 2024, Hal 119-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v3i3.7694
Website https://ejurnal.seminar-id.com/index.php/jussi

c. Analisa Proses Kompresi Pada File Pdf
Pada tahap ini file yang berformat .Pdf akan dikompresi menggunakan algoritma unary code dan algoritma prefix code yang bertujuan untuk memperkecil ukuran data.
d. Analisa Proses Dekompresi Pada File Pdf
Tahap dekompresi merupakan tahap mengembalikan data yang sudah dikempresi ke data asli sesuai dengan ukuran dan susunan-susunan datanya.
e. Proses Perbandingan Algoritma Unary Code dan Algoritma Prefix Code Dalam Kompres File Pdf
Pada tahap ini merupakan proses perbandingan algoritma unary code dan algoritma prefix code dalam kompresi pada file pdf berdasarkan parameter Ratio of Compression (RC) Compression Ratio (CR), Radudancy (RD,) Space Saving (SS).
f. Perancangan Sistem
Pada tahap ini akan diberikan gambaran mengenai perancangan aplikasi kompresi file Pdf yang akan diusulkan. Tahapan perancangan sistem ini menggunakan data yang telah dianalisis kedalam bentuk yang sederhana, mudah dan dapat dipahami oleh pengguna.
g. Implementasi
Tahap implementasi dilakukan untuk proses implementasi sistem yang telah dirancang berdasarkan hasil analisis yang telah dilakukan sebelumnya, dengan tujuan untuk mengetahui apakah sistem dapat berfungsi dengan benar dan sesuai kebutuhan atau masih diperlukannya perbaikan pada sistem tersebut.
h. Pengujian
Tahap pengujian melakukan pengujian hasil kompresi file pdf, apakah hasil yang diperoleh sesuai dengan yang diharapkan atau tidak.
i. Dokumentasi
Tahap Dokumentasi adalah tahap akhir dalam melakukan penelitian yang akan dibuat dalam bentuk laporan. Ditahap ini menjelaskan aplikasi yang telah dirancang agar memudahkan pembaca

2.2 Algoritma *Unary Code*

Unary code, juga dikenal sebagai kode termometer adalah pengkodean entropi yang dimulai dengan angka, diawali dengan n, n berikutnya adalah 0 (jika angka awal dipahami sebagai bilangan bulat non-negatif) atau n-1 diikuti oleh 0 (jika angka pertama dipahami sebagai angka positif). Misalnya, biner 5 adalah 111110 atau 11110. Seringkali n atau n-1, gunakan 1 setelah 0. 1 dan 0 bergantian tanpa kehilangan nilai normalnya. *Unary code* adalah awalan kode dan kode sinkronisasi sendiri. *Unary code* memiliki fungsi encoding dengan mengambil bit dari nilai karakter dan menulis bit b ke outputnya, dan dalam proses decoding fungsi nilai bit yang diambil adalah 0 atau 1 tergantung pada nilai awal dari sebuah berkas yang terkompresi [4]. Langkah-langkah algoritma unary coding adalah sebagai berikut [4].
a. Urutkan frekuensi kemunculan setiap karakter dari frekuensi tertinggi ke frekuensi terendah
b. Tetapkan bilangan non-negatif dari n-1 bit, diikuti dengan bit 0.
c. n terakhir diperoleh hanya dengan menggunakan n-1bit tanpa diikuti oleh sebuah bit 0, 2

Tabel 1. Tabel *Unary Code*

Kode (n) bit 1 diikuti oleh satu bit 0	Kode (n-1) bit diikuti oleh satu bit 0	*Unary Code*	Alternative
0	1	0	1
1	2	10	01
2	3	110	001
3	4	1110	0001
4	5	11110	00001
5	6	111110	000001

2.3 Algoritma *Prefix Code*

Prefix Code adalah jenis sistem kode yang dicirikan dengan memiliki awalan atribut yang mengharuskan tidak semua kata kode menjadi awalan kata kode lain dalam sistem yang ada di sistem. Ada empat kode dalam *prefix code* yaitu C1, C2, C3 dan C4 kode yang digunakan oleh penulis adalah kode C1. Untuk mendapatkan nilai sebelumnya dari kode C1, C2, C3 dan C4, harus mengetahui nilai dari *unary code* B(n) dan B(n). *Unary code* dari bilangan positif n didefinisikan sebagai n - 1 diikuti oleh 0 atau alternatifnya seperti angka nol n – 1 diikuti dengan satu [5]. Seperti tabel dibawah ini.

Tabel 2. *Unary Code*

No	Kode	Kode Alternatif
1	0	1
2	10	01
3	110	001
4	1110	0001

Jurnal Sains Dan Teknologi Informasi
Vol 3, No 3, Juni 2024, Hal 119-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v3i3.7694
Website https://ejurnal.seminar-id.com/index.php/jussi

No	Kode	Kode Alternatif
5	11110	00001
6	111110	000001
7	1111110	0000001
8	11111110	00000001
9	111111110	000000001
10	1111111110	0000000001

B (n) untuk menampilkan representasi biner dari bilangan bulat n, sedangkan B(n) untuk menampilkan nilai B(n) tanpa bit dengan menghilangkan nilai 1bit pertama dari setiap B(n). Setelah mendapatkan nilai B(n) dan B(n) barulah tentukan nilai kode C1 dengan contoh n = 5 = 1012, ukuran B (5) adalah 3 kemudian mulai dengan kode *unari* 110 (atau 001) dan tambahkan B (5) = 01, jadi semua kode adalah 110 | 01 (atau 001 | 10) Untuk mencari nilai kode C2, C3 dan C4, bisa dilihat dibawah ini [5].

Tabel 3. Tabel Ketentuan *Prefix Code*

No	*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
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

3. ANALISA DAN PEMBAHASAN

3.1 Analisa

Analisa dalam penelitian ini adalah menghitung dan merancang sebuah perangkat lunak untuk mengkompres *file* pdf, dalam proses penulisannya menggunakan algoritma *unary code* dan algoritma *prefix code*. Kedua algoritma ini merupakan bagian dari teknik kompresi *lossless*, yang dapat mengubah ukuran kapasitas data sesuai dengan karakter yang terdapat pada objek yang akan dikompresi**.** Penelitian kompresi ini dilakukan dengan menerapkan algoritma *unary code* dan algoritma *prefix codi* pada *file* dokumen pdf yang biasanya berukuran relatif besar. Oleh karena itu, hasil dari penelitian ini akan bermanfaat dalam mengurangi ukuran data *file* pdf. Pada proses analisa yang dilakukan penulis dalam penelitian ini ialah menggunakan *file* dokumen pdf yang akan dikompres kemudian diubah menjadi nilai *hexadecimal* melalui aplikasi HxD, setelah nilai hexadecimal dari *file* dokumen didapat maka akan dilakukan proses kompresi dan akan diperoleh *file* hasil kompresi.

3.1.1 Penerapan Algoritma *Unary Code*

Proses analisa, format *file* Pdf merupakan termasuk *file* yang berukuran besar. Dengan melakukan kompresi ini, *file* semula yang berukuran besar kemudian dikompresi hingga menjadi lebih kecil dari sebelumnya. Tahapan-tahapan algoritma unary coding adalah sebagai berikut:

a. Urutkan frekuensi kemunculan setiap karakter dari frekuensi tertinggi ke frekuensi terendah.
b. Tetapkan bilangan non-negatif dari n-1 bit, diikuti dengan bit 0.
c. n terakhir diperoleh hanya dengan menggunakan n-1bit tanpa diikuti oleh sebuah bit 0, 2

Sebelum melakukan proses teknik kompresi pada *file* pdf. Terlebih dahulu cari *file* pdf yang akan dikompresi, berikut ini contoh *file* pdf yang akan dikompresi dan dekompresi

Tabel 4. Sampel *File* Pdf

Nama *File*	Quiz
Ukuran *File*	3,80 Mb

Tabel 5. Tampilan Bilangan Hexadeciamal

25	50	44	46	2D	31	2E	37

Jurnal Sains Dan Teknologi Informasi
Vol 3, No 3, Juni 2024, Hal 119-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v3i3.7694
Website https://ejurnal.seminar-id.com/index.php/jussi

0A	25	B7	BE	AD	AA	0A	31

Dari tabel diatas, dapat diproleh nilai bilangan heksadesimal dari file pdf untuk dihitung secara manual. Kemudian proses kompresi untuk algoritma *unary code* dapat dilihat pada table berikut:

Tabel 6. Nilai Hexadecimal Dengan Frekuensi Binary Code Sebelum Dikompresi

No	Hexadecimal	Frekuensi	*Binary Code*	Jumlah Bit	Total Bit	Kode (n-1) (*Unary Code*)
1	25	2	10010100	8	16	0
2	0A	2	10100000	8	16	10
3	31	2	11000100	8	16	110
4	50	1	10100000	8	8	1110
5	44	1	10001000	8	8	11110
6	46	1	10001100	8	8	111110
7	2D	1	10110100	8	8	1111110
8	2E	1	10111000	8	8	11111110
9	37	1	11011100	8	8	111111110
10	B7	1	10110111	8	8	1111111110
11	BE	1	10111110	8	8	11111111110
12	AD	1	10101101	8	8	111111111110
13	AA	1	10101010	8	8	1111111111110
					128	

Dari tabel diatas, nilai *heksadesimal* berisi delapan bit biner. Jadi 16 nilai *heksadesimal* memiliki nilai biner 128 bit.

Tabel 7. Nilai Hexadecimal Dengan Frekuensi Unary Code Sesudah Dikompresi

No	Hexadecimal	Frekuensi	Kode (n-1) (*Unary Code*)	Jumlah Bit	Total Bit
1	25	2	0	1	2
2	0A	2	10	2	4
3	31	2	110	3	6
4	50	1	1110	4	4
5	44	1	11110	5	5
6	46	1	111110	6	6
7	2D	1	1111110	7	7
8	2E	1	11111110	8	8
9	37	1	111111110	9	9
10	B7	1	1111111110	10	10
11	BE	1	11111111110	11	11
12	AD	1	111111111110	12	12
13	AA	1	1111111111110	13	13
	Jumlah				97

Dari hasil komperesi diatas dengan menggunakan *unary code*, dapat dihitung kinerja kompresinya sebagai berikut:

a. *Ratio of Compression* (RC)

$$RC = \frac{\text{Ukuran Data Sebelum Dikompresi}}{\text{Ukuran Data Sesudah Dikompresi}}$$
$$RC = \frac{128}{97}$$
$$RC = 1{,}391$$

b. *Compression Ratio* (CR)

$$CR = \frac{\text{Ukuran Data Sesudah Dikompresi}}{\text{Ukuran Data Sebelum Dikompresi}} \times 100\%$$
$$CR = \frac{97}{128} \times 100\%$$
$$CR = 0{,}75\%$$

c. *Redudancy* (RD)

$$RD = \frac{\text{Ukuran Sebelum Dikompresi} - \text{Ukuran Sesudah Dikompresi}}{\text{Ukuran Data Sebelum Dikompresi}} \times 100\%$$
$$RD = \frac{128-97}{128} \times 100\%$$
$$RD\ 127\%$$

Jurnal Sains Dan Teknologi Informasi
Vol 3, No 3, Juni 2024, Hal 119-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v3i3.7694
Website https://ejurnal.seminar-id.com/index.php/jussi

d. *Space Saving* (SS)

$$SS = 1\frac{\text{Ukuran Data Sesudah Dikompresi}}{\text{Ukuran Data Sebelum Dikompresi}} \text{ x } 100\%$$

$$SS = 1 - \frac{97}{128} \text{ x } 100\%$$

$$SS = 0{,}24\%$$

Berdasarkan tabel diatas, nilai biner yang dihasilkan kemudian digabungkan menjadi: "01110111101111101111110110111111101111111101001111111110111111111110111111111111011111111111101011000000100001000" Kemudian adalah pengurangan bit ke urutan bit asli, mengurangi bit yang digunakan dengan menghapus *padding* dan *flagging*. Untuk mengembalikan bit menjadi bit string dengan membaca bit terakhir dan kemudian mengubah nilai biner ke decimal. Tampilkan pembacaan dalam bentuk n, kemudian gunakan rumus 7 + n untuk mengembalikan bit string ke bentuk aslinya.
"01110111101111101111110110111111101111111101001111111110111111111110111111111111011111111111101011000000100001000"
8 bit terakhir = **00001000** = 8 = n
7 + n = 7+ 8 = 15
Selanjutnya hilangkan bit string sebanyak 15bit terakhir, menjadi:
"01110111101111101111110110111111101111111101001111111110111111111110111111111111011111111111101011"

3.1.2 Penerapan Algoritma *Prefix Code*

Prefix Code adalah jenis sistem kode yang dicirikan dengan memiliki awalan atribut yang mengharuskan tidak semua kata kode menjadi awalan kata kode lain dalam sistem yang ada di sistem. Ada empat kode dalam *prefix code* yaitu C1, C2, C3 dan C4 kode yang digunakan oleh penulis adalah kode C1. Pada penelitian ini penulis akan mengerjakan 2 proses utama yaitu kompresi dan dekompresi, penulis akan melakukan kompresi dan dekompresi sebuah *file* pdf menggunakan algoritma *prefix code*. Nilai bilangan hexasadecimal diurutkan berdasarkan frekuensi, nilai frekuensi paling banyak akan berada diurutan paling atas, dapat dilihat pada tabel berikut:

Tabel 8. Nilai Hexadecimal dengan Frekuensi

No	Hexadesimal	Frekuensi
1	25	2
2	31	2
3	0A	2
4	50	1
5	44	1
6	46	1
7	2D	1
8	2E	1
9	37	1
10	B7	1
11	BE	1
12	AD	1
13	AA	1
	Jumlah	16

Setelah nilai *hexadecimal*dan frekuensi diuraikan berdasarkan urutan kemunculannya pada sebuah strinng. Selanjutnya adalah proses kompresi untuk algoritma *prefix code* dapat dilihat pada tabel berikut:

Tabel 9. Nilai *Hexadecimal* Yang Sebelum Dikompresi

No	Hexadesimal	Biner	Frekuensi	Bit	Frekuensi x Bit
1	25	00100101	2	8	16
2	31	00110001	2	8	16
3	0A	00001010	2	8	16
4	50	10110111	1	8	8
5	44	10111110	1	8	8
6	46	10101101	1	8	8
7	2D	10101010	1	8	8
8	2E	00101110	1	8	8
9	37	00110111	1	8	8
10	B7	01000100	1	8	8
11	BE	01000110	1	8	8

Jurnal Sains Dan Teknologi Informasi
Vol 3, No 3, Juni 2024, Hal 119-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v3i3.7694
Website https://ejurnal.seminar-id.com/index.php/jussi

No	Hexadesimal	Biner	Frekuensi	Bit	Frekuensi x Bit
12	AD	10101101	1	8	8
13	AA	01010000	1	8	8
		Jumlah Bit			128

Dari tabel diatas, nilai *hexasadecimal* berisi delapan bit biner. Jadi 16 nilai *hexasacesimal* memiliki nilai biner 128 bit. Selanjutnya, proses kompresi untuk algoritma *prefix code* dapat dilihat dibawah ini:

Tabel 10. Nilai Hexadecimal Sesudah Dikompresi

No	Hexadecimal	Frekuensi	*Codeword* (C1)	Bit	Frekuensi x Bit
1	25	2	0	1	2
2	31	2	100	3	6
3	0A	2	101	3	6
4	50	1	11000	5	5
5	44	1	11001	5	5
6	46	1	11010	5	5
7	2D	1	11011	5	5
8	2E	1	110000	6	6
9	37	1	110001	6	6
10	B7	1	110010	6	6
11	BE	1	110011	6	6
12	AD	1	110100	6	6
13	AA	1	110101	6	6
		Total Bit			70 Bit

Setelah nilai hexal diketahui, untuk mengubah nilai hexal kedalam suatu karakter, dari hasil kompresi tersebut dapat dilihat kinerja algoritma *prefix code* dibawah ini:

a. *Ratio of Compression* (RC)

$$RC = \frac{\text{Ukuran Data Sebelum Dikompresi}}{\text{Ukuran Data Sesudah Dikompresi}}$$
$$RC = \frac{128}{70}$$
$$RC = 1{,}828$$

b. *Compression Ratio* (CR)

$$CR = \frac{\text{Ukuran Data Sesudah Dikompresi}}{\text{Ukuran Data Sebelum Dikompresi}} x100\%$$
$$CR = \frac{70}{128} x\ 100\%$$
$$CR = 0{,}54\%$$

c. *Redundancy* (RD)

$$RD = \frac{\text{Ukuran Data Sebelum Dikompresi} - \text{Ukuran Sesudah Dikompresi}}{\text{Ukuran Data Sebelum Dikompresi}} x\ 100\%$$
$$RD = \frac{128-70}{128} x\ 100\%$$
$$RD = 127\%$$

d. *Space Saving* (SS)

$$SS = 1\frac{\text{Ukuran Data Sesudah Dikompresi}}{\text{Ukuran Data Sebelum Dikompresi}}\ x\ 100$$
$$SS = 1\frac{70}{128}\ x\ 100\%$$
$$SS = 0{,}45\%$$

3.2 Implementasi

Implementasi adalah proses mengimplementasikan sistem yang telah dirancang berdasarkan hasil analisis penelitian yang telah dilakukan sebelumnya. Pada bagian ini membahas tentang perangkat keras (*hardware*), perangkat lunak (*software*) dan spesifikasi tampilan sistem. Sistem tidak akan dapat berjalan dengan baik jika tidak didukung oleh *hardware* dan *software* yang memadai, berikut adalah spesifikasi *hardware* yang penulis gunakan dalam merancang system:

a. *Processor* Intel(R) Celeren(R) CPU N3150 @ 1.60GHz 1.60GHz
b. Memori 4.00 GB
c. *Harddisk* 500 GB
d. Monitor 1366 x 768 pixel
e. Keyboard dan mouse

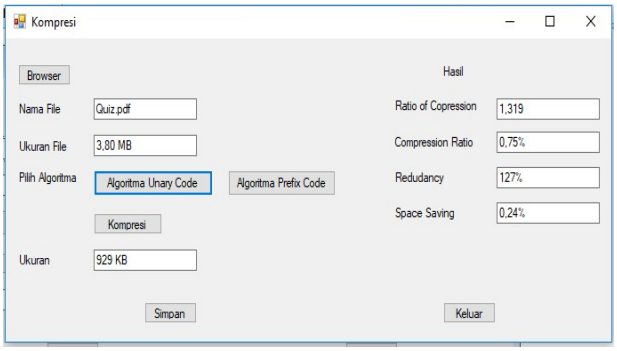
Spesifikasi perangkat lunak (*software*) yang digunakan dalam menjalankan sistem agar berjalan dengan baik, maka

Jurnal Sains Dan Teknologi Informasi
Vol 3, No 3, Juni 2024, Hal 119-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v3i3.7694
Website https://ejurnal.seminar-id.com/index.php/jussi

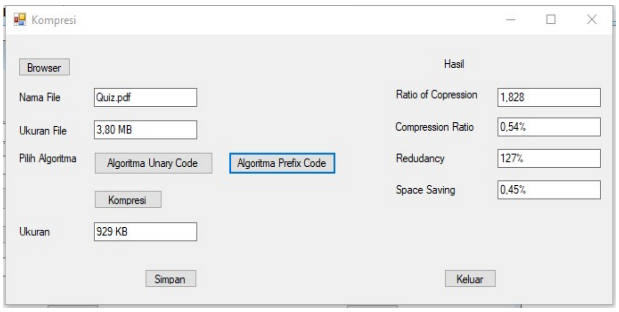
diperlukan spesifikasi minimum perangkat lunak (*software*) sebagai berikut:

a. SO (Sistem operasi) Windows 10 Pro
b. Aplikasi Microsoft Visual Studio 2008

Pada proses menampilkan pengkompresian *file* pdf dengan menggunakan algoritma *unary code* dan algoritma *prefix code* dimana untuk melakukan proses kompresi langkah pertama yang dilakukan adalah menginput *file* pdf. Adapun proses kompresi dapat dilihat pada gambar berikut:

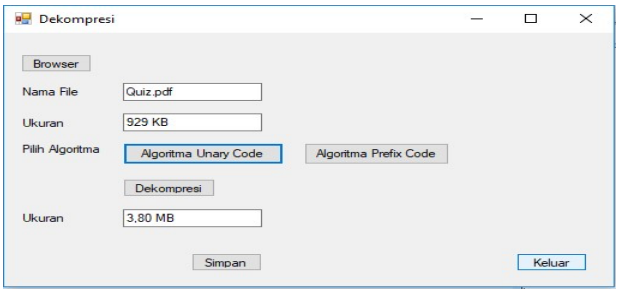


Gambar 2. Tampilan Kompresi Algoritma *Unary Code*



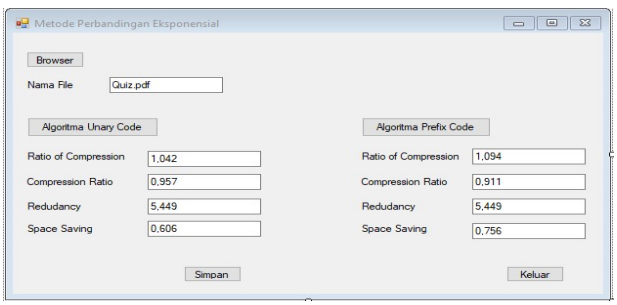
Gambar 3. Tampilan Kompresi Algoritma *Prefix Code*

Pada proses menampilkan penginputan nilai dekompresi pada *file* pdf, dimana untuk melakukan proses dekompresi langkah pertama yang dilakukan adalah menginput *file* pdf yang sudah terkompresi sebelumnya. Adapun proses dekompresi dapat dilihat pada gambar berikut:



Gambar 4. Tampilan Dekompresi Algoritma *Unary Code* Dan *Prefix Code*

Tampilan hasil perbandingan antara algoritma *unary code* dan algoritma *prefix* code pada *file* pdf, dimana untuk mendapatkan proses perbandingan eksponensial langkah pertama yang dilakukan adalah menginput *file* pdf. Adapun proses perbandingan eksponensial dapat dilihat pada gambar berikut:



Gambar 5. Tampilan Metode Perbandingan Eksponensial

Jurnal Sains Dan Teknologi Informasi
Vol 3, No 3, Juni 2024, Hal 119-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v3i3.7694
Website https://ejurnal.seminar-id.com/index.php/jussi

4. KESIMPULAN

Berdasarkan hasil penelitian penulis yang telah dilakukan, telah membuaat beberapa kesimpulan yang berkaitan dengan analisa perbandingan algoritma *unary* code dan algoritma *prefix code* untuk kompresi *file* pdf sebagai berikut yaitu Sesudah melalukan prosedur kompresi *file* pdf dengan menggunakan algoritma *unary code* dan algoritma *prefix code* dapat menghasilka suatu *file* pdf yang memiliki ukuran yang cukup besar maka dapat dikompresi menjadi *file* pdf yang berukuran lebih kecil. Hasil rasio kompresi dipengaruhi oleh *file* yang dikompresi. Semakin banyak tipe karakter yang berulang dalam *file* terkompresi, maka semakin tinggi nilai rasio kompresinya. Dalam perancangan aplikasi kompresi *file* pdf dengan algoritma *unary code* dan algoritma *prefix* menggunakan Microsoft Visual Studio 2008, semuanya berjalan dengan baik. System yang dirancang dalam penelitian ini masih memiliki kekurangan sehingga dapat dikembangkandan diperbaiki lebih lanjut. Adapun saran yang dapat penulis sebagai berikut, yaitu Dalam penelitian ini menggunakan algoritma *unary code* dan algoritma dapat dikembangkan lebih baik lagi dengan menambahkan objek seperti *file* dokumen, *file* video dan lain-lain. Sehingga memberikan maanfaat yang lebih baik di masa mendatang. Dalam penelitian ini, perancangan aplikasi kompresi *file* pdf hanya dapat mengkompresi *file* berektensi pdf dan diharapkan kedepannya dapat mengkompresi *file* pdf dengan ekstensi yang lain seperti word, excel dan lainya. Aplikasi kompresi *file* pdf kedepannya dapat dikembangkan menjadi berbasis web dan android sehingga dapat digunakan secara *online* dan kapan saja.

REFERENCES

[1] F. E. Naibaho, "Implementasi Algoritma J-Bit Encoding Pada Kompresi File Pdf," *Jurnal Sistem Komputer dan Informasi (JSON),* Vols. 1, Nomor 3, Mei 2020, no. 2685-998x, pp. 278-283, 2020.
[2] D. P. U. Soumi Rohmah Saragih, "Penerapan Algoritma Prefix Code Dalam Kompresi Data Teks," *Komik (Konfrensi Nasional Teknologi Informasi dan Komputer),* Vols. 4, Nomor 1, Oktober 2020, no. 2597-4645, pp. 249-252, 2020.
[3] S. A. Muhammad Alfarizi, "Penerapan Algoritma Prefix Code Dalam Kompresi File Vidio," *KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer),* Vols. 4, Nomor 1, Oktober 2020, no. 2597-4645, pp. 232-235, 2020.
[4] M. Sitorus, "Penerapan Algoritma Unary Coding Pada Aplikasi Kompresi Short Message Service (SMS)," *Journal of Computer System and Informatics (JoSYC),* Vols. 1, No 2, Februari, no. 2714-8912, pp. 39-45, 2020.
[5] W. A. Harahap, "Studi Komparasi Kinerja Algoritma Goldbach G1 Code Dan Algoritma Unary Codes Dalam Kompresi Teks," 2017.
[6] Mawar, "Perancangan Aplikasi Kompresi File Pdf Dengan Menerapkan Algoritma Punctured Elias Codes," *Jurnal Informasi dan Teknologi Ilmiah (INTI),* Vols. 7, No 3, Juni 2020, no. 2301-9425 (Media Cetak), pp. 217-223, 2020.
[7] K. H. M. Yulia Darnita, "Kompresi Data Teks Dengan Menggunakan Algoritma Sequitur," *Jurnal SISTEMASI,* Vols. 8, Nomor 1 Januari 2019 : 104 - 113, 2019.
[8] S. Kaur, "Entropy Coding and Different Coding Techniques," *Journal of Network Communications and Emerging Technologies (JNCET),* vol. 6, no. 5, May 2016.
[9] G. M. David Salomon, Handbook of Data Compression 5th,, 2019.