

## Implementasi Algoritma Elias Gamma Kompresi Pada File Teks

Dina Cahayati<sup>1</sup>, Akim M.H. Pardede<sup>2</sup>, Husnul Khair<sup>3</sup>

Program Studi Sistem Informasi, STMIK KAPUTAMA Binjai

Jl. Veteran No. 4A-9A, Binjai 20714, Sumatera Utara

Email : <sup>1</sup>dinacahayati6@gmail.com

### ABSTRAK

Ukuran data yang besar mengakibatkan pemborosan memori dan lambatnya proses pemindahan data. Kompresi bertujuan untuk mengurangi ukuran data tersebut menjadi sekecil mungkin algoritma *Elias Gamma* merupakan jenis kompresi *lossless* yang digunakan dalam penelitian ini, yang mana akan diukur kinerjanya dengan *Ratio of Compression* (RC), *Compression Ratio* (CR), *Redundancy* (Rd), waktu kompresi (detik) dan waktu dekompresi (detik) pada *file* teks. Kompresi *file* teks dilakukan dengan membaca *string* pada *file* teks dan mengkodekan *string* menggunakan *Elias Gamma*, kemudian melakukan proses kompresi. Hasil akhir dari kompresi adalah *file* berekstensi \*.eg yang berisi informasi karakter dan *string bit* hasil kompresi yang dapat didekompresi. Algoritma *Elias Gamma* dipengaruhi oleh jumlah variasi karakter, Dalam proses kompresi pada *string Elias Gamma* Rasio Kompresi rata-rata sebesar 2.192%.

**Kata Kunci :** Dekompresi, *Elias Gamma*, File Teks, Kompresi.

### ABSTRACT

*Large data sizes result in wasted memory and slow data transfer processes. Compression aims to reduce the size of the data to be as small as possible. Elias Gamma algorithm is a type of lossless compression used in this study, whose performance will be measured by Ratio of Compression (RC), Compression Ratio (CR), Redundancy (Rd), compression time ( seconds) and decompression time (seconds) on the text file. Text file compression is done by reading the string in the text file and encoding the string using Elias Gamma, then performing the compression process. The final result of the compression is a file with \*.eg extension which contains character information and a compressed bit string that can be decompressed. Elias Gamma's algorithm is influenced by the number of character variations. In the compression process on Elias Gamma's strings the average compression ratio is 2.192%.*

**Keywords:** Decompression, *Elias Gamma*, Text Files, Compression.

## 1. LATAR BELAKANG MASALAH

Seiring dengan perkembangannya, teknologi data dari *file-file* terus berkembang dan mengalami peningkatan ukuran menjadi semakin besar dari waktu ke waktu. Perkembangan teknologi yang semakin maju dan penambahan jumlah pengguna komputer yang semakin banyak menyebabkan perpindahan data dari satu perangkat ke perangkat lain. Data-data tersebut umumnya dikompresi terlebih dahulu agar proses pertukaran tidak memakan waktu yang terlalu lama.

Teks merupakan kumpulan karakter atau *string* dalam satu kesatuan, Isinya banyak karakter di dalamnya yang selalu menyebabkan masalah pada perangkat penyimpanan terbatas dan kecepatan data setiap saat. Meskipun penyimpanan bisa diganti yang lain lebih besar, membuat semua orang berusaha mencari cara yang bisa digunakan untuk mengkompres teks.

Beberapa penelitian terdahulu yang berhubungan dengan penelitian ini antara lain tentang algoritma Huffman dan Lz78, Kompresi data dapat digunakan untuk mengurangi penggunaan memory yang terlalu besar pada saat menyimpan data, juga untuk mempercepat proses pengiriman data untuk kompresi dalam bentuk teks, Algoritma huffman lebih tepat digunakan dari pada algoritma LZ78 dengan menyesuaikan type data yang akan dikompresi.

## 2. LANDASAN TEORI

### 2.1 Pengertian Kompresi

Kompresi adalah suatu teknik pemampatan data sehingga diperoleh file dengan ukuran yang lebih kecil daripada ukuran aslinya. Kompresi bekerja dengan mencari pola-pola perulangan pada data dan

menggantinya dengan sebuah penanda tertentu. Metode kompresi (*compression*) atau metode dengan pemampatan data pada dasarnya dapat dikelompokkan ke dalam dua kelompok besar, yaitu:

#### 1. Metode Lossless

Metode Lossless merupakan metode dimana data kompresi adalah kelas dari algoritma data kompresi yang memungkinkan data yang asli dapat disusun kembali dari data kompresi. Kompresi data lossless digunakan dalam berbagai aplikasi seperti format ZIP dan GZIP. Lossless juga sering digunakan sebagai komponen dalam teknologi kompresi data lossy. Kompresi Lossless digunakan ketika sesuatu yang penting pada kondisi asli. Beberapa format gambar seperti BMP (*Bitmap*), PNG (*Portable Network Graphics*) atau GIF (*Graphics Interchange Format*) hanya menggunakan kompresi lossless, sedangkan yang lainnya seperti TIFF (*Tagged Image Format File*) dan PNG dapat menggunakan metode lossy atau lossless. Metode lossless menghasilkan data yang identik atau sama dengan data aslinya. Misalnya pada citra atau gambar dimana metode ini menghasilkan hasil yang tepat sama dengan citra semula, pixel per pixel sehingga tidak ada informasi yang hilang akibat adanya proses kompresi. Namun demikian ratio kompresi (Rasio kompresi yaitu, ukuran file yang dikompresi dibanding yang tidak terkompresi dari file) dengan metode ini masih sangat rendah. Metode ini cocok untuk kompresi citra yang mengandung informasi penting yang tidak boleh rusak akibat kompresi, misalnya gambar hasil diagnosa medis. Contoh metode lossless adalah metode run-length, Huffman, delta dan LZW (*Lempel-Ziv-Welch*).

#### 2. Metode Lossy

Lossy kompresi adalah suatu metode untuk mengkompresi data dan men-dekompresinya, data yang diperoleh mungkin berbeda dari yang aslinya tetapi cukup dekat perbedaannya. Lossy kompresi ini paling sering digunakan untuk mengkompres data multimedia (Audio dan gambar statis). Sebaliknya, kompresi lossless diperlukan untuk data teks dan file, seperti catatan bank, artikel teks dll. Format kompresi lossy mengalami generation loss, yaitu jika dilakukan secara berulang kali kompresi dan dekompresi file akan menyebabkan kehilangan kualitas citranya secara progresif. hal ini berbeda dengan kompresi data lossless. ketika pengguna yang menerima file terkompresi secara lossy (misalnya untuk mengurangi waktu download) file yang diambil dapat sedikit berbeda dari yang asli di level Bit ketika tidak dapat dibedakan oleh mata dan telinga manusia untuk tujuan paling praktis.

Proses kompresi yang dilakukan terhadap sebuah citra memberikan manfaat yang sangat besar, misalnya dalam industri multimedia saat ini. Diantara manfaat tersebut :

1. Pada proses pengiriman data (*data transmission*) pada saluran komunikasi data. Citra yang telah dikompresi membutuhkan waktu pengiriman yang lebih cepat (singkat) dibandingkan dengan citra yang tidak dikompresi. Hal ini dapat dilihat pada aplikasi pengiriman gambar lewat *fax*, *videoconferencing*, pengiriman data medis, pengiriman gambar dari satelit luar angkasa, pengiriman gambar via telepon genggam, download gambar dari internet, dan sebagainya.
2. Selain manfaat pada proses pengiriman data, kompresi citra juga bermanfaat pada saat dilakukan proses penyimpanan data (*data storing*) di dalam media sekunder (*storage*). Sebuah citra yang telah dilakukan kompresi atau pemampatan akan membutuhkan ruang memori di dalam media storage yang lebih sedikit (kecil) jika dibandingkan dengan citra yang tidak dilakukan proses kompresi (Mahesa, 2017).

### 2.2 Pengertian Dekompresi

Kebalikan dari proses kompresi data yaitu proses dekompresi. Dekompresi adalah sebuah proses untuk mengembalikan data baru yang telah dihasilkan oleh proses kompresi menjadi data awal. Dekompresi yang menghasilkan data sama persis dengan data aslinya sebelum kompresi, maka data tersebut disebut *lossless compression*. Sebaliknya, jika hasil dekompresi menghasilkan data tidak sama persis dengan data aslinya sebelum dekompresi, karena ada data yang dihilangkan karena dirasa tidak terlalu penting tetapi tidak mengubah informasi yang dikandungnya, disebut *lossy compression*. Setelah dilakukan proses kompresi terhadap sebuah gambar atau file, maka sebuah citra akan dapat dikembalikan ke dalam bentuk citra semula yaitu dengan melakukan dekompresi (*decompression*) pada data citra tersebut (Mahesa, 2017).

### 2.3 Jenis Teknik Kompresi

Teknik kompresi secara umum dapat diklasifikasikan menjadi tiga yaitu:

1. Entropy coding adalah teknik kompresi yang menggunakan proses lossless. Tekniknya tidak berdasarkan pada media dengan spesifikasi dan karakteristik tertentu namun berdasarkan urutan data serta tidak memperhatikan semantik data.
2. Source coding adalah teknik kompresi dengan menggunakan proses lossy. Teknik ini berkaitan dengan data semantik (arti data) dan media.

- Hybrid coding adalah teknik kompresi dengan menggunakan kombinasi atau gabungan dari entropy coding dan source coding (Dzulhaq & Andayani, 2014).

Beberapa faktor yang biasa digunakan untuk menganalisa kualitas dari suatu teknik kompresi data yaitu :

- Ratio of Compression (RC)*, merupakan nilai perbandingan antara ukuran bit data sebelum dikompresi.

$$R_c = \frac{\text{Ukuran bit data sebelum dikompresi}}{\text{Ukuran bit data setelah dikompresi}}$$

- Compression Ratio (CR)*, merupakan persentase perbandingan antara data yang sudah dikompresi dengan data yang belum dikompresi.

$$C_r = \frac{\text{Ukuran bit data sebelum dikompresi}}{\text{Ukuran bit data setelah dikompresi}} \times 100\%$$

- Redudancy (Rd)*, merupakan persentasi dari hasil selisih antara ukuran data sebelum dikompresi dengan data setelah dikompresi.

$$Rd = 100\% - CR \text{ (Pratiwi \& Zebua, 2019).}$$

## 2.4 Rasio Kompresi

Proses kompresi adalah proses encoding yang menghasilkan data yang sudah dikompresi yang disebut aliran data encoded. Sebaliknya aliran data yang telah dikompresi harus dilakukan proses dekompresi untuk menghasilkan kembali aliran data yang asli. Karena proses dekompresi menghasilkan decoding dari aliran data yang sudah dikompresi maka hasilnya adalah aliran data decoded.

Tingkat pengurangan data yang dicapai sebagai hasil dari proses kompresi disebut rasio kompresi. Rasio ini merupakan perbandingan antara panjang data string asli dengan panjang data string yang sudah dikompresi, seperti dituliskan dalam persamaan berikut:

$$\text{Rasio} = \frac{\text{Ukuran File Asli}}{\text{Ukuran File Terkompresi}}$$

Jika dinyatakan dalam prosentase maka dituliskan dalam persamaan berikut:

$$P = (1 - \frac{\text{Ukuran File Terkompresi}}{\text{Ukuran File Asli}}) * 100\%$$

Yang berarti ukuran file berkurang sebesar P (dalam persentase) dari ukuran semula. Semakin tinggi rasio tingkat. suatu teknik kompresi data maka semakin efektif teknik kompresi tersebut. Pada saat dikompresi, rasio kompresi akan berselang-seling berdasarkan pengaruh data terhadap algoritma yang digunakan (Dzulhaq & Andayani, 2014).

## 2.5 Pengertian Algoritma Elias Gamma Code

Algoritma *elias gamma code* dibuat oleh Peter *Elias*. Tabel kode *elias gamma*, *elias* dibuat dengan menambah panjang kode dalam *Unary* (u). Kode berikutnya,  $E\gamma$  ditambahkan pada panjang kode (M) dalam biner ( $\beta$ ). Dengan demikian, *elias gamma code*, yang juga untuk bilangan bulat positif, sedikit lebih kompleks untuk dibangun. Aturan untuk mengkodekan sebuah bilangan dengan menggunakan *Elias Gamma* adalah sebagai berikut :

- Rubah bilangan kode dalam bentuk biner.
- Kurangkan 1 dari jumlah bit yang dipilih pada langkah pertama dan tambahkan sesuai dengan banyaknya bilangan nol. Proses yang ekivalen untuk menyatakan proses yang pada *point* nomor dua adalah sebagai berikut :
  - Pisahkan *integer* menjadi pangkat 2 tertinggi ( $2^N$ ) yang dapat dan ditampungnya sisakan digit biner N dari *integer* tersebut.
  - Kodekan N dalam bentuk *unary*, jika N adalah nol maka diikuti oleh satu.
  - Tambahkan sisa digit biner N yang telah dihasilkan.

Proses kompresi atau *encoding* suatu *integer* dengan berdasarkan *elias gamma* dilakukan dengan cara :

- Tentukan nilai N untuk pangkat yang paling mendekati nilai n yang dituliskan sebagai  $\beta(n)$ . Nilai ini disebut sebagai *unary code*, dimana jumlah nilai N ditulis menjadi angka 0 dan diakhiri dengan angka 1.
- Dapatkan nilai L dengan mengurangi nilai n dengan nilai  $2N$ , nilai yang didapati diubah menjadi bilangan *biner* Pembentukan kode *elias gamma code* dapat diambil sebuah contoh  $n = 9$  Lalu temukan bilangan bulat N terbesar sehingga  $2N \leq n < 2N+1 = 23 \leq 9 < 23+1$  setelah ditemukan kode N terbesar lalu rubah nilai n menjadi biner lalu hilangkan 1 bit paling kiri  $9 = 1001 \rightarrow 001$ , kodekan dalam bentuk

unary N sebagai 0 diikuti oleh 1 sehingga dihasilkan unary 3 → 0001, lalu tambahkan sisa digit biner n dibelakang kode unary yang telah dihasilkan 0001001.

Langkah berikutnya adalah mengkonversi karakter sesuai dengan kode *Elias Gamma*, sebelum dikonversi terlebih dahulu dilakukan pemeriksaan terhadap panjang *string bit* dengan langkah sebagai berikut :

1. Jika sisa bagi panjang *string bit* terhadap 8 adalah 0 maka tambahkan 00000001. Nyatakan dengan *bit* akhir.
2. Jika sisa bagi panjang *string bit* terhadap 8 adalah n (1, 2, 3, 4, 5, 6, 7) maka tambahkan 0 sebanyak 7 – n + “1” di akhir *string bit*. Nyatakan dengan L, lalu tambahkan bilangan biner dari 9 – n. Nyatakan dengan *bit* akhir.

Proses Dekompresi/*decoding* dalam kode *elias gamma code* dapat dilakukan langkah-langkah adalah:

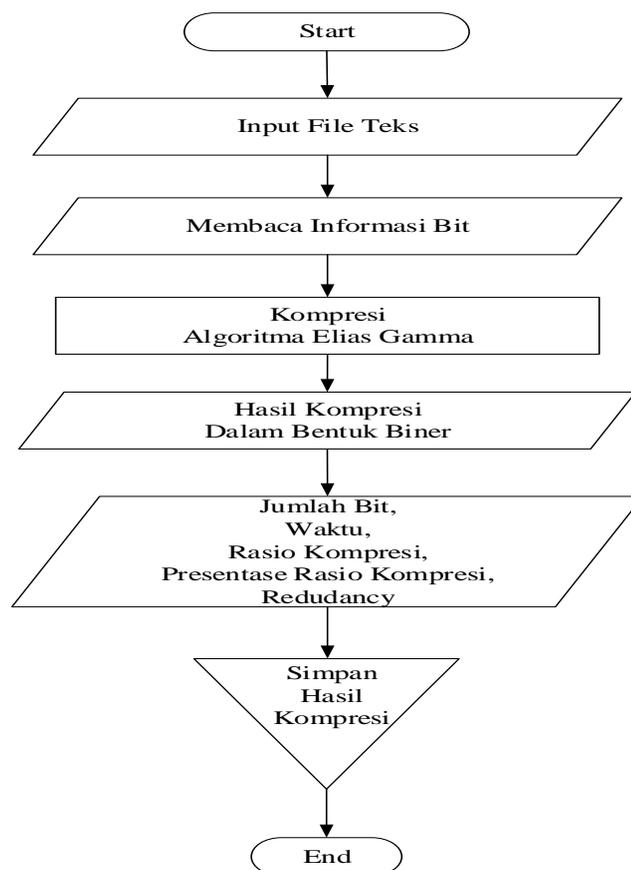
1. Lakukan pembacaan pada 8 *bit* terakhir, hasil pembacaan berupa bilangan decimal. Nyatakan hasil pembacaan dengan n.
2. Hilangkan *bit* pada bagian akhir sebanyak 7 + n.

Berikut ini adalah algoritma dekomposisi *Elias Gamma Code* adalah :

1. Lakukan pembacaan *string bit* dari awal sampai angka 1 ditemukan. Catat posisi angka 1 dan nyatakan sebagai p Nyatakan jumlah angka 0 dengan n.
2. Lakukan pembacaan *string bit* setelah angka 1 sebanyak n.
3. Ganti kode hasil pembacaan dengan karakter sesuai berdasarkan tabel *Elias Gamma Code* (Sartika & Zebua, 2019).

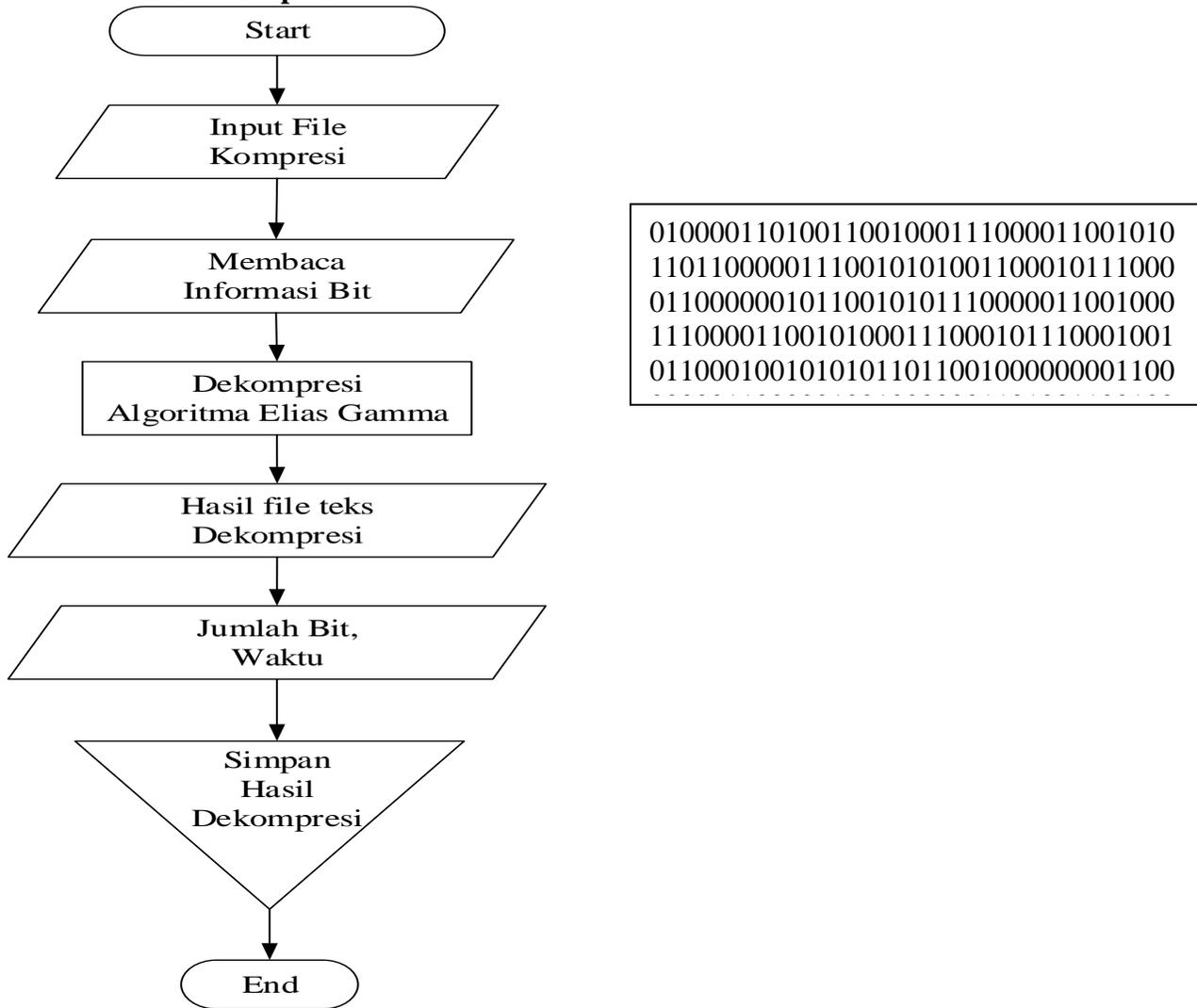
### 3.3 ANALISA DAN PEMBAHASAN

#### 3.1 Flowchart Kompresi



Gambar 1 Flowchart Kompresi

### 3.2 Flowchart Dekompresi



Gambar 2 Flowchart Dekompresi

### 3.3 Perhitungan Kompresi File Teks Algoritma Elias Gamma

Input File Teks :

Nama Saya Dina Cahayati Dan Saya Mahasiswi Kaputama Binjai

Untuk selanjutnya melakukan perhitungan jumlah bit :

$$\begin{aligned}
 \text{Jumlah bit} &= \text{panjang pesan} \times \text{bit} \\
 &= 58 \times 8 \\
 &= 468 \text{ bit}
 \end{aligned}$$

Tabel 1 Pergantian Teks Dengan Kode *Elias Gamma Code*

Setelah itu munculah hasil kompresi menjadi biner :

Saat dilakukan kompresi dengan algoritma *Elias Gamma* sehingga menghasilkan jumlah bit hasil kompresi yaitu 155 bit. Lalu dilakukan perhitungan rasio kompresi dan presentase rasio kompresi dimana perhitungannya dituliskan dalam persamaan berikut:

$$\text{Rasio} = (\text{Ukuran File Asli}) / (\text{Ukuran File Terkompresi})$$

Jika dinyatakan dalam Presentase maka dituliskan dalam persamaan berikut:

$$P = (1 - (\text{Ukuran File Terkompresi}) / (\text{Ukuran File Asli})) * 100\%$$

Maka proses perhitungan rasio kompresi dan presentase rasio kompresi adalah sebagai berikut:

Jumlah bit file asli = 464 bit

Jumlah bit file hasil kompresi = 155 bit

Rasio kompresi = (Ukuran File Asli) / (Ukuran File Terkompresi) = 464 / 155 = 2.99

Persentase rasio kompresi =  $(1 - (\text{Ukuran File Terkompresi}) / (\text{Ukuran File Asli})) * 100\% = 1 - (155 / 464) * 100\% = 66.59\%$

Redudancy = 100% - 66.59% = 33.41%

No	Karakter	Frekuensi	Code Elias Gamma	Bit	Bit x Frekuensi
1	a	17	1	1	17
2		8	001	3	24
3	i	6	011	3	18
4	y	3	00001	5	15
5	S	2	00011	5	10
6	m	2	01001	5	10
7	D	2	01011	5	10
8	t	2	0000001	7	14
9	n	3	0000011	7	21
10	s	2	0001001	7	14
11	h	2	0001011	7	14
12	N	1	0100001	7	7
13	M	1	0100011	7	7
14	C	1	0101001	7	7
15	w	1	0101011	7	7
16	K	1	00000001	9	9
17	p	1	000000011	9	9
18	u	1	00000101	9	9
19	B	1	000001011	9	9
20	j	1	00010001	9	9
Total Char		58	Jumlah Bit x Frekuensi	155	

## 4. PEMBAHASAN DAN IMPLEMENTASI

### 4.1 Pembahasan

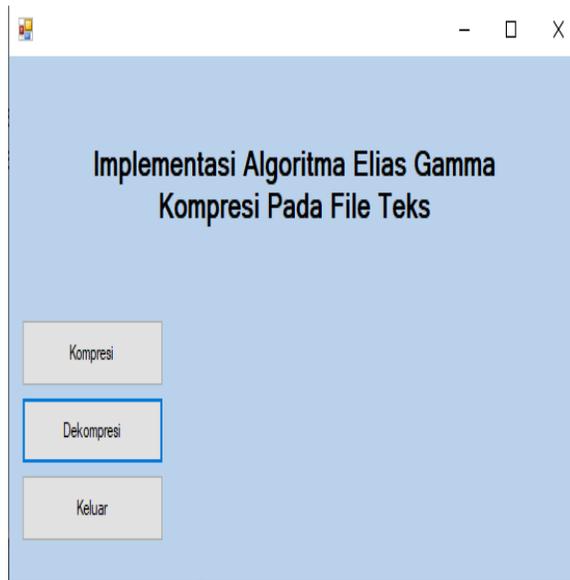
Sistem ini dibangun menggunakan bahasa pemrograman Microsoft Visual 2010 yang memiliki konsep program berorientasi objek, sehingga aplikasi yang dirancang lebih sederhana dan memiliki bagian-bagian yang terpisah sehingga dapat di modifikasi, dikembangkan dan dirawat dengan lebih mudah. Aplikasi ini dirancang menggunakan Microsoft Visual 2010 yang menyediakan berbagai fasilitas dalam mengembangkan aplikasi.

### 4.2 Pembahasan Langkah Kerja Aplikasi

Tampilan dari sistem Kompresi dan dekompresi file teks yang telah dirancang menggunakan aplikasi pemrograman *Visual Basic 2010*, yaitu sebagai berikut:

#### a. Tampilan Halaman Utama

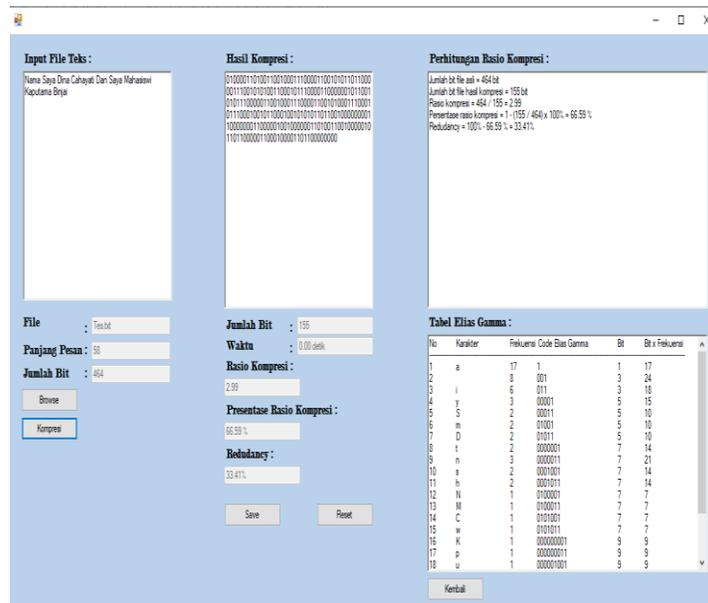
Tampilan halaman utama sistem tersebut adalah sebagai berikut :



Gambar 3 Tampilan Halaman Utama

**b. Tampilan Halaman Proses Kompresi**

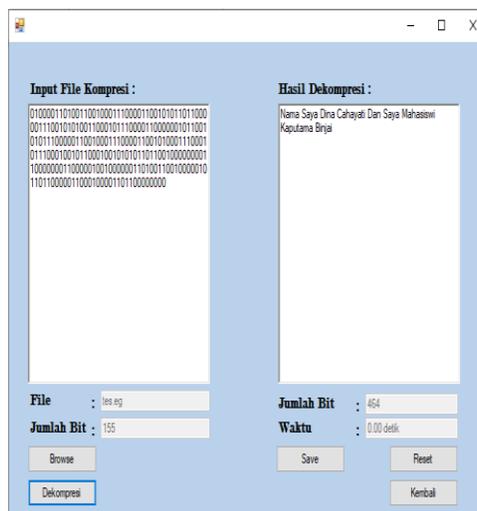
Tampilan halaman Proses Kompresi sistem tersebut adalah sebagai berikut :



Gambar 4 Tampilan Halaman Proses Kompresi

**Tampilan Halaman Proses Dekompresi**

Tampilan halaman Proses Dekompresi sistem tersebut adalah sebagai berikut :



Gambar 5 Tampilan Halaman Proses Dekompresi

## 5. KESIMPULAN DAN SARAN

### 5.1 Kesimpulan

Berdasarkan hasil perancangan dan pembuatan program aplikasi kompresi *file* teks menggunakan algoritma Elias Gamma ini dapat di ambil kesimpulan sebagai berikut :

1. Aplikasi ini dibuat untuk mengkompresi *file* teks dan dapat mengembalikan *file* teks yang terkompresi ke dalam bentuk semula.
2. Berdasarkan grafik hubungan antara waktu proses kompresi dan dekompresi dengan ukuran *file* tersebut menunjukkan bahwa ukuran *file* berbanding lurus dengan waktu, Semakin besar ukuran *file* tersebut maka semakin besar waktu kompresi dan dekompresinya.

### 5.2 Saran

Adapun saran penulis usulkan dalam pengembangan aplikasi ini adalah sebagai berikut:

1. Bahan implementasi dapat diganti dengan *file* lain berupa citra, audio maupun video.
2. Untuk penelitian lebih lanjut di harapkan dapat mengembangkan sistem yang telah ada untuk dapat melakukan kompresi dan dekompresi.

## DAFTAR PUSTAKA

- [1]. Dzulhaq, M. I., & Andayani, A. A. (2014). Aplikasi Kompresi File Dengan metode Lempel-Ziv-Welchhof. *Jurnal Sisfotek Global*, 4(1), 1–4.
- [2]. Mahesa, K. (2017). Rancang Bangun Aplikasi Kompresi Dan Dekompresi Pada Citra Digital Menggunakan Metode Huffman. *Jurnal Processor*, 12(1), 948–963.
- [3]. Pratiwi, D., & Zebua, T. (2019). Analisis Perbandingan Kinerja Algoritma Fixed Length Binary Encoding Dan Algoritma Elias Gamma Code Dalam Kompresi File Teks. *KOMIK (Konferensi Nasional Teknologi Informasi Dan Komputer)*, 3(1), 424–430. <https://doi.org/10.30865/komik.v3i1.1623>
- [4]. Sartika, H., & Zebua, T. (2019). Perancangan Dan Implementasi Algoritma Elias Gamma Code Untuk Mengkompresi Record Database Pada Aplikasi Rangkuman Pengetahuan Umum Lengkap. *KOMIK (Konferensi Nasional Teknologi Informasi Dan Komputer)*, 3(1), 259–265. <https://doi.org/10.30865/komik.v3i1.1600>