Informasi dan Teknologi Ilmiah (INTI)
ISSN 2301-9425 (Media Cetak), ISSN 2798-0340 (Media Online)
Volume 10, No 1, Oktober 2022
Hal: 7-12

Kombinasi Algoritma Sequitur Dan Adaptive Huffman Coding Untuk Kompresi File Teks

Mega Wati Harahap

Fakultas Ilmu Komputer dan Teknologi Informasi, Program Studi Teknik Informatika, Universitas Budi Param, Medan, Indonesia
Email: megawatiharahap18@gmail.com
Email Penulis Korespondensi: megawatiharahap18@gmail.com

Abstrak−File teks memiliki ukuran yg sangat besar sehingga mempengaruhi ruang penyimpanan dan proses pengiriman.ukuran inilah menjadi faktor berapa lama waktu yang dibutuhkan untuk melakukan proses pengiriman file. Besarnya ukuran besar akan memakan waktu transfer yang lebih lama,maka ukuran besar dapat diatasi dengan cara melakukan proses kompresi. Kompresi dilakukan untuk mengurangi ukuran dari sebuah file Algoritma kompresi yang dapat digunakan yaitu algoritma siquitur. Kelebihan algoritma siquitur adalah dengan mengganti 2 karakter berdampingan dengan 1 karakter terminal pada karakter set.pengkodean huffman yang beroperasi pada ukuran tabel kode dikurangi dan demikian secara signitifikasi mengurangi tuntutan sumber daya untuk menyimpan dan memperbarui tabel kode selama runtime. Adaptive Huffman Coding adalah teknik pengkodean adaptif berdasarkan pengkodean Huffman. Pada penelitian ini algoritma yang akan digunakan adalah sequitur dan adaptive huffman coding.

Kata Kunci: File Teks; Kompresi; Algoritma Sequitur; Adaptive Huffman Coding;

Abstract−Text files are very large in size, so they affect storage space and the sending process. This size is a factor in how long it takes to send the file. The large size will take a longer transfer time, so the large size can be overcome by doing the compression process. Compression is done to reduce the size of a file. The compression algorithm that can be used is the siquitur algorithm. The advantage of the siquitur algorithm is that it replaces 2 characters side by side with 1 terminal character in the character set. Huffman encoding that operates at the size of the code table is reduced and thus significantly reduces the resource demands of storing and updating code tables during runtime. Adaptive Huffman Coding is an adaptive coding technique based on the Huffman coding. In this study, the algorithms that will be used are sequitur and adaptive huffman coding.

Keywords: Text File; Compression; Sequitur Algorithm; Adaptive Huffman Coding;

1. PENDAHULUAN

Semakin berkembangnya teknologi maka semakin banyak pula data-data atau file-file yang ingin kita simpan atau pun kita kirimkan, tetapi kadang kala kapasitas memori yang kita miliki tidak sebanding dengan data yang akan kita simpan, oleh karena itu data-data yang akan disimpan perlu dikompres terlebih dahulu supaya ukurannya menjadi lebih kecil. Apabila ukuran data dapat dikompres menjadi lebih kecil dari ukuran aslinya, maka secara otomatis memori dapat menyimpan data lebih banyak lagi dan dari segi pengiriman pun akan semakin cepat dan menghemat waktu yang dibutuhkan.

File teks merupakan file yang berisi informasi dalam bentuk teks. Datayang berasal dari dokumen pengolah kata, angka yang digunakan dalam perhitungan, nama dan alamat dalam basis data merupakan contoh masukan data teks yang terdiri dari karakter, angka dan tanda baca, Teks adalah kumpulan karakter- karakter atau string yang menjadi satu kesatuan. Teks merupakan bentuk media paling umum yang memuat banyak karakter didalamnya selalu menimbulkan masalah pada media.

Kompresi adalah pengubahan data yang berupa kumpulan karakter menjadi bentuk kode dengan tujuan untuk menghemat kebutuhan tempat penyimpanan dan waktu transmisi data. Ada beberapa faktor yang dijadikan sebagai bahan pertimbangan dalam memilih algoritma yang akan digunakan dalam kompresi data. Kecepatan kompresi, ukuran hasil komprsi besarnya redudansi dan kompleksitas algoritma. Jika semakin besar ukuran data yang disimpan maka membutuhkan media penyimpanan yang besar. Oleh karena itu kemudian muncul metode-metode yang bertujuan untukmengkompresi data agar dapat menghemat tempat penyimpanan data salah satunya yaitu kompresi data teks agar bisa diperkecil ukurannya.

Penempatan data (data kompresssion) merupakan salah satu kajian di ilmu komputer yang bertujuan untuk mengurangi ukuran file sebelum menyimpan atau memindahkan data tersebut ke dalam media penyimpanan (Storage Device). penyimpanan seperti floppy disk, hard diks mempunyau kapasitas terbatas. Jika data yang akan disimpan pada media penyimpanan semakin bertambah dan berukuran besar, maka media penyimpana tidak dapat menyimpan data tersebut karena melebihi kapasitas, Oleh karena itu untuk mengatasi masalah ini digunakanlah penempatan data.

Menurut penelitian terdahulu yang dilakukan oleh Yulia Darmita, Khairunnisyah,Husni Mubarok pada article yang berjudul “Kompresi Data Teks Menggunakan Algoritma Siquitur”. Algoritma siquitur merupakan algoritma sebuah waktu linier yang menyimpulkan tata bahasa bebas konteks (conteks-free gramntar) kedalam suatu penempatan untuk mengurangi masukan yang berulang atau dengan kata lain melakukan pengelompkan karakter yang sama pada isi file dan mengatasi permasalahan dalam penempatan sebuah data, bersifat Lossy comoression. Berdasarkan hasil pengujian yang dilakukan, file hasil kompresi oleh algoritmasiquitur memudahkan dalam menggunakan internet sehingga waktu yang diperlukan akan menjadi lebih pendek dan kemungkinan pekerjaan Download dan Aploud gagal akan menjadi lebih kecil. Kemudian transfer file melalui jaringan akan lebih cepat, waktu

pengiriman tergantur dari provider yang cepat atau tidak serta ukuran file yang akan dikirim dan membantu dalam mengurangi ukuran dari file sehingga dapat mengurangi kapasitas penyimpanan suatu memory / RAM.[1]

Menurut penelitian yang dilakukan oleh Metariana Aditya pada article yang berjudul " Studi Banding Algoritma Kompresi Huffman Coding dan Adaptive Huffman Coding ". Adaptive huffman coding disebut juga dinamyc huffman coding adalah yeknik pengkodean adaftip berdasarkan pengkodean huffman, ini memungkinkan pembuatan kode karena simbol-simbol. Manfaat prosedur one- pass adalah bahwa sumbernya dapat dikodekan secara real time. Meskipun menjadi lebih sensitif terhadap kesalahan trasmisi, karena hanya satu kerugian yang menghancurkan keseluruhan kode[2].

2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Dalam melakukan penelitian agar mendapat hasil seperti yang diharapkan, maka diperlukan tahapan penelitian. Dimana tahapan penelitian yang akan dilakukan digambarkan.

a. Studi Literatur
   Pada tahap ini dilakukan pengumpulan referensi yang diperlukan dalam penelitian. Hal ini dilakukan untuk memperoleh informasi dan data yang diperlukan untuk penulisan skripsi ini. Referensi yang digunakan dapat berupa buku, jurnal internasional yang didapat di internet.
b. Analisa Dan Perancangan Sistem
   Pada tahap ini dilakukan untuk menganalisa dan mengkombinasikan kerja kompresi file dengan Algoritma *Adaptive Huffman Coding* dan *Seqitur*. Dan selanjutnya dilakukan pembuatan *UML* proses kompresi serta perancangan *interface.*
c. Pengujian Sistem
   Pada Tahap ini dilakukan pengujian terhadap sistem yang telah dibangun.
d. Implementasi Sistem
   Pada tahap ini dilakukan implementasi terhadap hasil analisa dan perancangan kedalam bahasa program.
e. Dokumentasi
   Membuat dokumentasi sistem dari tahap awal sampai pengujian sistem,kemudian untuk disusun dalam format penelitian.

2.2 Kompresi

Kompresi data di dalam konteks ilmukomputer adalah merupakan ilmu atau seni dalam merepresentasikan informasi yang terdapat pada data ke dalam suatu bentuk yang lebih padat (kecil)[3]. Walaupun mengalami pengecilan ukuran pada media penyimpanan, data atau informasi yang terdapat di dalamnya tetap dapat dimengerti dan digunakan sebagaimana mestinya, meskipun ada juga yang mengalami sedikit perubahan pada kompresi dengan jenisLossy Compression. Pada teks, kompresi bekerja dengan cara mengubah atau menyusun ulang susunan karakter sehingga menghasilkan suatu pola dan karakter baru yang berjumlah lebih sedikit dari jumlah karakter awal dan pada akhirnya teks memiliki ukuran penyimpanan yang lebih kecil. Biasanya teks dikompresi dengan menggunakan jenisLossless Compression[4].

2.3 File Teks

File teks juga merupakan file teks yang berisi informasi-informasi dalam bentuk teks. Data yang berasal dari dokumen pengolah kata, angka yang digunakan dalam perhitungan, nama dan alamat dalam basis data merupakan contoh masukan data teks yang terdiri dari karakter dan tanda baca. File teks merupakan file komputer yang tersusun atas rangkaian baris teks. Jenis-jenis file teks yang termasuk dalam kategori umumnya bersisi rangkaian karakter tanpa informasi format visual. File teks mirip dengan file yang dihasilkan oleh program pengolahan kata yang konten utamanya bersifat tekstual masukan dan keluaran data teks direpresentasikan.jenis-jenis file seperti Docx, Doc, Txt, Xml, Pdf, Ppt [5].

2.4 Algoritma Sequitur

Algoritma sequitur merupakan algoritma linier yang menyimpulkan tata bahasa bebas konteks (conteks-free grammar) kedalam suatu penempatan untuk mengurangi masukkan yang berulang. Operasi sequitur terdiri dari pemastian dua sifat yang berlaku. Ketika menjelaskan algoritma , sifa-sifatnya bertindak sebagai batasan . Algoritma beroperasi menjalankan batasan didalam sebuah tata bahasa yang ketika diagram keunikan (diagram uniqueness) dianggar, sebuah aturan baru dibentuk, dan ketika batasan aturan digunakan (rule utility) dianggap aturan yang sia-sia dihapus[6]

Diagram keunikan mempunyai arti bahwa tidak ada pasangan dari simbol atau diagram muncul lebih dari sekali dalam sebuah tata bahasa. Jika hal ini terjadi maka akan melanggar aturan batasan pertama (diangram uniqueness) sehingga akan membentuk aturan baru (simbol non-terminal) yang akan menggantikan simbol atau diagram yang muncul lebih dari sekali.[6]

Rule utility mempunyai arti bahwa setiap aturan produksi digunakan lebih dari sekali. Dan jika ada aturan yang hanya digunakan sekali maka akan terjadi pelanggaran pada batasan kedua (Rule utility) sehinggaaturan yang hanya yang dipakai sekali akan dihapus atau dihilangkan.

2.5 Algoritma Adaptive Huffman Coding

Algoritma Adaptive Huffman Coding diperkenalkan oleh Faller dan Gallager (Faller 1973, Gallaber 1978). Dan kemudian Knuth (1985) memberikan kontribusi dengan peningkatan pada algoritmanya dan menghasilkan algoritma yang dikenal dengan algoritma FGK. Versi terbaru dari Adaptive Huffman Coding diperkenalkan oleh Vitter (1987). Semua metode yang ditemukan merupakan skema define-word menentukan pemetaan dari pesan sumber menjadi codeword yang berdasarkan pada perkiraan probabilitas pesan sumber. Kode bersifat adaptif atau menyesuaikan, berganti sesuai dengan perkiraan optimalnya pada saat itu juga. Dalam hal ini, Adaptive Huffman Coding merespon lokalitas. Dalam pengertian, encoder mempelajari karakteristik dari sumber, sedangkandecoder harus mempelajari kesamaan dengan encoderyaitu dengan cara memperbaharui pohon Huffman sehingga terjadi sinkronisasi antara decoder dengan encoder.

Sistem ini juga memberikan keuntungan lain dari sisi kebutuhan transmisi data, data akan lewat hanya sekali (tanpa tabel statistik). Metode one-pass tidak akan menarik apabila jumlah bit yang ditransmisikan lebih besar dari metode two- pass.Namun, performa dari metode ini, dalam ruang lingkup jumlah bit yang ditransmisikan, dapat lebih baik daripada algoritma Static Huffman Coding. Masalahan ini tidak kontradiktif dengan optimalisasi pada metode statik, karena metode ini optimal berdasarkan time-variant. Tetapi tidak tertutup kemungkinan kinerja dari metode adaptif dapat lebih buruk dari pada metode statik. Metode adaptif Faller, Gallager dan Knuth adalah dasar dari UNIX Utility Compact. Kinerja compact ini tergolong baik, karena memiliki faktor kompresi mencapai 30% - 40%.
Kompresi dimulai dengan pohon huffman yang kosong. Tidak ada simbol- simbol yang telah ada sebelumnya. Simbol pertama dimasukkan pada aliran dalam bentuk terkompresi. Simbol ini kemudian ditambahkan ke dalam pohon, jika simbol ini ditemukan lagi didalam aliran (pohon) maka frekuensi bertambah satu, dan akan memodifikasi pohon, pohon diperiksa lagi untuk melihat apakah itu masih pohon huffman (kode terbaik), jika tidak maka pohon akan disusun kembali, dan mengakibatkan perubahan pada kode.

Berikut adalah ringkasan operasi yang dibutuhkan untuk memperbarui pohon. perulangan dimulai pada saat simbol dimasukan. Simbolnya adalah daun yang kita nyatakan dengan X, dengan frekuensi terjadinya F. Setiap iterasi dari perulangan melibatkan tiga tahap sebagai berikut:

a. Kombinasi X dengan penerusnya di pohon (dari kiri ke kanan dan bawah ke atas). Jika penggantinya segera memiliki frekuensi F + 1 atau lebih, simbolnya masih dalam urutan yang diurutkan dan tidak perlu mengubah apapun. Jika tidak, beberapa penerus X memiliki frekuensi F yang identik atau lebih kecil. Dalam kasus ini, X harus ditukar dengan simbol terakhir dalam kelompok ini (kecuali X yang tidak boleh ditukar dengan induknya).
b. Kenaikan frekuensi X dari F ke F + 1 adalah kenaikan frekuensi semua induknya.
c. Jika X adalah akar, perulangan berhenti, jika tidak, perulangan akan berulang dengan induk dari node X.

3. HASIL DAN PEMBAHASAN

3.1 Analisa Masalah

Semakin besarya kebutuhan manusia akan informasi maka semakin banyak pula manusia mengumpulkan data. Kecenderungan manusia dalam mengumpulkan data mengakibatkan penggunaan penyimpanan data yang semakin besar. Semakin besar media penyimpanan yang diperlukan untuk menyimpan data maka dibutuhkan kompresi data. Kompresi data adalah proses meredupsi ukuran data untuk menghasilkan reperesentasi digital yang padat namun tetap dapat mewakili kuantitas informasi data tersebut. Dalam penelitian ini penulis mengkombinasikan algoritma siquitur dan adaptive Huffman coding dalam mengkompresi file teks.

Analisis sistem merupakan tahap awal dalam sebuah penelitian yang bertujuan mengetahui masalah terkait dalam pembuatan sebuah sistem dan menggambarkan proses-proses yang ada di dalam sistem untuk menghasilkan keluaran yang sesuai dengan kebutuhan pemakai.

Pada tahap ini akan dijelasakan secara umum cara kerja algoritma *Siquitur* dan algoritma *Adaptive Huffman Coding* dalam kompresi file teks. Algoritma *adaptive Huffmancoding* merupakan teknik kompresi dengan cara melakukan pengkodean dalam bentuk bit untuk mewakili data karakter. Algoritma *sequitur*merupakan teknik kompresi yang bekerja dengan cara merubah atau mengganti karakter yang sama yang memiliki jumlah lebih dari satu.

Hasil kompresi dari algoritma *Siquitur* dan algoritma *Adaptive Huffman Coding* bertujuan untuk mengatasi kelemahan dalam ruang penyimpanan dan mempercepat proses pengiriman file. Karena siquitur bekerja dan berperan sebagai penyusunan (sorting) ulang karakter dan adaptive huffman coding yang akan mengkodekan (encoding) karakter dari hasil susunan sebelumnya sehingga menghasilkan tingkat kompresi yang lebih efektif.

Analisa kebuthan sistem membahas secara garis besar apa saja yang dibutuhkan dalam membangun sistem tersebut. Ada beberapa kebutuhan yang harus dimiliki sistem yaitu membaca file *string* dan file hasil kompresi, melakukan proses kompresi dan dekompresi file teks berdasarkan algoritma *Siquitur* atau *Adaptive Huffman Coding* dan mampu membandingkan hasil kompresi *Siquitur* dan *Adaptive Huffman Coding* untuk kompresi file teks.

3.1.1 Kompresi dan Dekompresi File Teks Dengan Algoritma *Sequitur*

Setelah file teks yang berhasil dibaca menjadi *String,* maka selanjutnya akan dilakukan proses kompresi dengan algoritma *Siquitur* dan *Adaptive Huffman Coding.*

Pada tahap ini akan dilakukan kompresi dan dan dekompresi beberapa string dengan algoritma siquitur dan algoritma Adaptive Huffman Coding. Hasil dari kompresi kedua algoritma tersebut akan dikombinasikan.Misalkan ada file teks yang berisi string "mega mega". Untuk mengetahui ukuran string tersebut dapat dilihat pada tabel 1.

Tabel 1. Ukuran String Sebelum Dikompresi

Karakter	Frekiuensi	Biner	Bit	Frekuensi x bit
M	2	01101101	8	16
E	2	01100101	8	16
G	2	01100111	8	16
A	2	01100001	8	16
Sp	1		8	8
	Jumlah Frekuensi x Bit			72

a. Kompresi File Teks dengan *Sequitur*

Misalkan kita ingin mengkompresi string " mang maman makan kacang " dengan algoritma siquitur. Untuk mengetahui ukuran string tersebut lihat tabel 2.

Tabel 2. Kompresi dengan algoritma *sequitur*

Proses	string	Diagram	Nonterminal	Rule	New String	DeleteRule
1	mega	me	A	A=me	Ga	
2	mega	Ag	B	B=Ag	a	
3	mega	Ba	C	C=Ba	ABC	

Pada tabel 2. karakter mega mega akan diperiksa untuk melihat pasangan karakter lebih dari satu kali, pasangan karakter yang muncul lebih dari 1 kali adalah "me,ag,ba", maka karater "me,ag.ba" akan diproses, karater "me,ag.ba" akan dimasukkan ke dalam diagram yang menghasilkan nonterminal A,B,C dan menghasilkan karakter baru atau tidak ,jika tidak maka proses selesai.

Tabel 3. Total Bit Algoritma *Sequitur*

Karakter	Urutan	Frekiuensi	Biner	Bit	Frekuensi x bit
M	M	2	01100001	8	16
E	E	2	01101101	8	16
G	G	2	01101110	8	16
A	A	2	01100111	8	16
Spasi	Sp	1		8	8
		Jumlah Frekuensi x Bit			72

b. Dekompresi algoritma siquitur

Untuk dekompresi algoritma sequitur dapat dilihat pada Tabel 4. berikut :

Tabel 4. Dekompresi Algoritma *Sequitur*

Proses	Delete Rule	New String	Rule	Nonterminal	Diagram	String
1		Ga	A=me	A	me	mega
2		a	B=Ag	B	Ag	mega
3		ABC	C=Ba	C	Ba	mega

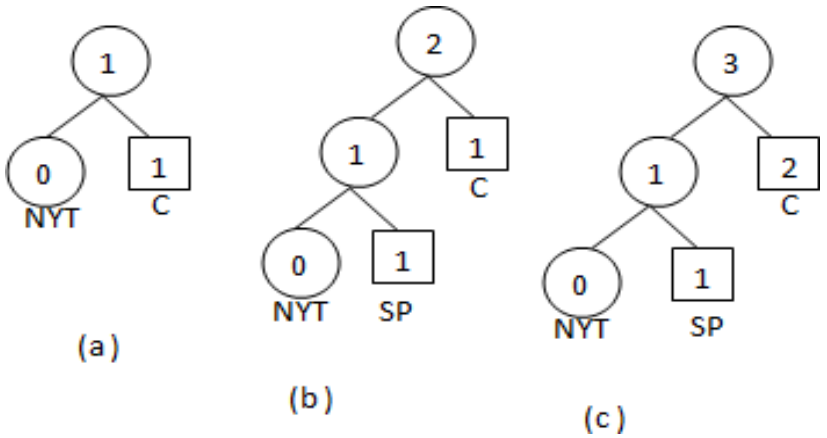
a. Mega mega ma=A
   Aga Aga
b. Ag=B
   Ba Ba
c. Ba=C
   C C

Hasil dari sequitur : C C

3.1.2 Kompresi dan Dekompresi File Teks Dengan *Adaptive Huffman Coding.*

Setelah file teks yang berhasil dibaca menjadi *string*, maka selanjutnya akan dilakukan proses kompresi dan dekompresi dengan algoritma *adaptive Huffman coding* dan *sequitur*.

a. Kompresi file teks *adaptive Huffman coding*
Pada *adaptive Huffman codingstring* yang dibaca akan dimasukkan kedalam pohon dan jumlah frekuensi yang lebih tinggi akan ada diatas. Misalkan kita ingin mengkompresi file txt yang memiliki string "gadjahmada", dengan cara sebagai berikut.



Gambar 1. Kompresi *Adaptive Huffman Coding*

Pada tahap pengujian program kompresi *file* teks ini dengan menggunakan algoritma sequitur dan Adaptive Huffman Coding, penulis melakukan dekompresi terhadap beberapa data teks. Adapun untuk melihat data hasil dari kompresi tersebut dapat dilihat pada tabel berikut:

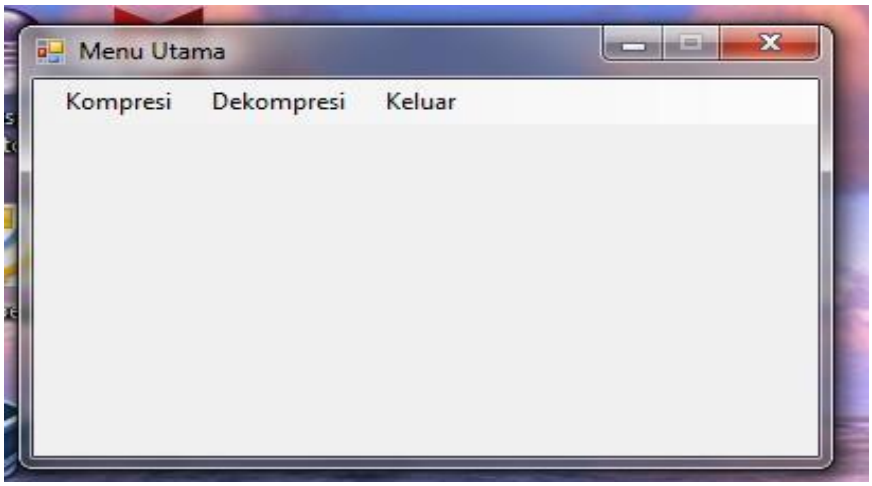
Tabel 5. Hasil Pengujian Program

Data Sebelum Dikompresi		Data Sesudah Dikompresi	
Nama file	Ukuran size	Nama file	Ukuran size
Test.txt	9 bytes	Text.sqt	3 bytes

3.2 Tampilan Sistem

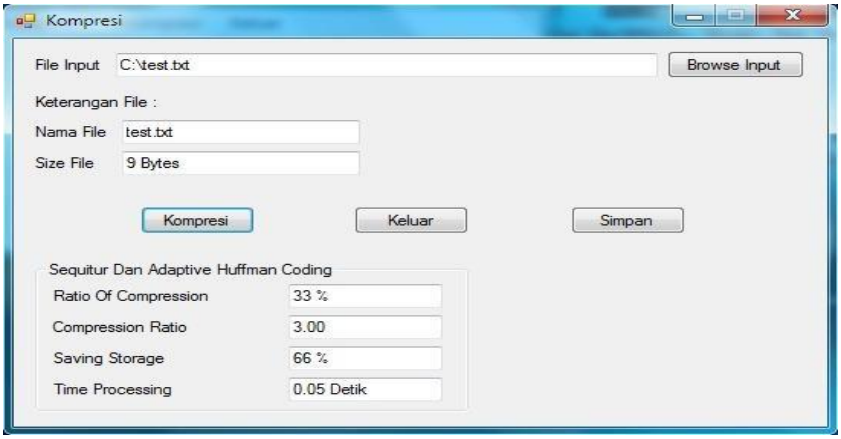
Tampilan program merupaka tampilan dari apalikasi yang akan dibangun, berupa *screenshot* dari tampilan apalikasi tersebut. tampilan program aplikasi tersebut terdiri dari jendela *menu utama*, jendela *Kompresi* dan jendela *dekompresi*. Berikut adalah tampilan menu utama pada aplikasi yang di rancang sebagai berikut :

a. Tampilan menu utama
Menu utama dapat menampilkan jendela paling awal ketika aplikasi pertama kali dijalankan, berisi tombol-tombol, seperti tombol kompresi, dekompresi, dan tombol keluar. Berikut tampilan menu utama yaitu:



Gambar 3. Form Menu Utama

b. Tampilan Jendela Kompresi
Tampilan jendela kompresi merupakan halaman yang akan di gunakan dalam melakukan proses kompresi. Berikut tampilan jendela kompresi yaitu:



Gambar 4. Tampilan Input Kompresi

c. Tampilan Jendela Dekompresi

Tampilan input proses dekompresi merupakan halaman yang di gunakan untuk melakukan proses dekompresi. Pada *form* proses dekompresi ada bagian- bagian menu di dalamnya yang meliputi sebagai berikut :

Gambar 5. Tampilan Input Dekompresi

4. KESIMPULAN

Adapun kesimpulan yang diperoleh dari penelitian ini dimana Proses kompresi file teks dengan kombinasi algoritma sequitur dan Adaptive Huffman coding dilakukan dengan membaca teks lalu dilakukan proses kompresi dengan sequitur, hasil dari sequitur dilanjutkan ke kompresi dengan Adaptive Huffman Coding. Kombinasi dilakukan dengan algoritma Sequitur yang berperan sebagai penyusun ulang karakter dan Adaptive Huffman Coding yang akan mengkodekan karakter dari hasil susunan sebelumnya sehingga menghasilkan tingkat kompresi yang lebih efektif. Kompresi file teks dapat dirancang dan dibangun dengan menggunakan aplikasi Microsoft Visual Basic 2010 dengan menerapkan algoritma Sequitur dan Adaptive Huffman Coding sehingga diharapkan dapat memudahkan penulis dalam mengkompresi ukuran file teks..

REFERENCES

[1] A. Ahmadi and W. Supriyono, *Psikologi Belajar*. Jakarta: Rineka Cipta, 2004.
[2] H. Umar, *Metode Penelitian*. Jakarta: Raja Grafindo Persada, 2005.
[3] A. Wedianto, H. L. Sari, and Y. S. H., "Analisa Perbandingan Metode Filter Gaussian, Mean Dan Median Terhadap Reduksi Noise," *J. Media Infotama*, vol. 12, no. 1, pp. 21–30, 2016, doi: 10.37676/jmi.v12i1.269.
[4] S. Suprijanto, *Matematika*. Jakarta: Yudisthira, 2009.
[5] I. M. Pu, *Fundamental Data Compression*. London: Butterworth- Heinemann, 2005.
[6] T. Sutoyo, *Teori Pengolahan Citra Digital*. Yogyakarta: Andi, 2009.
[7] G. G. Maulana, "Pembelajaran Dasar Algoritma Dan Pemrograman Menggunakan El-Goritma Berbasis Web," *J. Tek. Mesin*, vol. 6, no. 2, pp. 69–73, 2017, doi: 10.22441/jtm.v6i2.1183.
[8] M. E. Frayoga, P. Studi, and T. Informatika, "Perancangan Aplikasi Kompresi File Teks dengan Menerapkan Algoritma Sequitur," vol. 6, no. 1, pp. 45–50, 2019.
[9] D. Salomon and G. Motta, *Handbook of Data Compression*. London: Springer Science & Business Media, 2010.