Journal of Informatics Management and Information Technology
ISSN 2774-4744 (Media Online)
Vol 2, No 3, Juli 2022
Hal 78−82
https://hostjournals.com/jimat | DOI 10.47065/jimat.v2i3.149

Perancangan Aplikasi Kompresi File Teks Menggunakan Algoritma Context Tree Weighting

Maymiar Albina

Fakultas Ilmu Komputer Dan Teknologi Informasi, Program Studi Teknik Informatika, Universitas Budi Darma, Medan, Indonesia
Email: [1]mayniaralbina@gmail.com

Abstrak-Saat ini penggunaan aplikasi sebagai informasi semakin umum digunakan. tetapi ada masalah yang sering dijumpai, yaitu kebutuhan akan tempat besar sebagai media penyimpanan. Kompresi data adalah sebuah proses mengubah sekumpulan data menjadi suatu bentuk kode untuk menghemat kebutuhan tempat penyimpanan data. Algoritma Context Tree Weighting adalah salah satu algoritma kompresi data. Untuk mengetahui hasil proses kompresi dilakukan media rasio kompresi dan space saving. Hasil yang akan diketahui, rasio kompresidan space saving. Sesuai dengan hasil uji coba yang dilakukan terlihat bahwa data yang semula mempunyai ukuran dari besar dapat terkompresi dengan sangat baik diimplementasikan file teks kompresi.

Kata Kunci: Kompresi, File Teks, CTW, Teks

Abstrack-Currently the use of applications as information is increasingly being used. but there is a problem that is often encountered, namely the need for a large space as a storage medium. Data compression is the process of converting a set of data into a coded form to save the need for data storage. Context Tree Weighting algorithm is a data compression algorithm. To determine the results of the compression process, compression ratio and space saving media are used. The results will be known, the compression ratio and space saving. In accordance with the results of the trials conducted, it can be seen that the data that was originally large in size can be compressed very well with the implementation of compression text files.

Keywords: Compresion; File Text, CTW, Text

1. PENDAHULUAN

Kompresi data adalah proses mengubah aliran masukan data (sumber aliran atau data mentah asli) ke aliran data lain (keluaran, *bitstream*, atau aliran terkompresi) yang memiliki ukuran lebih ringkas. Kompresi *file* teks diproses menggunakan proses memampatkan data teks agar sizenya ringkas dan masih mempertahanakn bobot akhir dari suatu file. Jadi diperlukan kualitas kompresi yang baik yaitu ukuran yang minimum dengan tidak mengurangi kualitas tetapi mengurangi ukuran data tersebut[1].

File teks merupakan sebuah *file* yang isinya adalah teks berupa gabungan dari karakter huruf, angka, dan simbol. Ukuran dari sebuah *file* tergantung dari banyaknya isi karakter pada *file* tersebut. *File* teks berisi data *teks* yang dapat disimpan dalam format teks biasa atau format teks kaya 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[2][3].

Pada penelitian berikutnya yang dilakukan oleh Jeffrey N. Denenberg, Edward D. Weinberger, Michael L. Gordon, kompresi data dengan judul "Data Compression Method for use in a Computerized Informational and Transactional Network" menyimpulkan bahwa pada penelitiann tersebut menemuana beberapa keuntuntungan setelah mereka menggunakan metode *CTW*, berikut adalah beberapa kesimpulana dan keuntungan dari penelitannya: mengurangi transmisi data dan persyaratan penyimpanan yang interaktif jaringan layanan tanpa secara substansial, untuk digunakan dalam antar jaringan layanan aktif yang tidak mahal untuk dipasang dan beroperasi, mudah diperbarui untuk mengakomodasi perubahan dalam data yang dikirimkan dan disimpan dalam layanan interaktif, untuk menentukan frekuensi terjadinya *byte-pair* frekuensi dalam aliran data jaringan[4].

Permasalahan yang timbul adalah *file* teks format teks selalu digunakan di dunia maya karena sizenya yang sangat kecil karena ukuran *file* teks tidak lebih dari 10mb.

Untuk mengatasi permasalahan yang ada dari *file* teks yang didapat dengan format teks maka teknik kompresi diperlukan dengan tujuan memperkecil ukuran dari *file* teks berformat teks . kompresi memiliki berbagai jenis algoritma yang dapat digunakan. Maka dari itu penulis bermaksud menggunakan algoritma *CTW* untuk mengompresi *file* audio berformat Teks tersebut. Diharapkan dengan memperkecil ukuran dari *file* yang ada dapat mengefisiensikan penyimpanan yang ada pada *harddisk.*

2. METODOLOGI PENELITIAN

2.1 Kompresi Data

David Salomon mengatakan bahwa data kompresi adalah proses mengkonversikan sebuah input data stream (*stream* sumber, atau data mentah asli) menjadi data *stream* lainnya (*bit stream* hasil, atau *stream* yang telah terkompresi) yang berukuran lebih kecil. Pemampatan merupakan salah satu cara dalam ilmu komputer yang bertujuan untuk memadatkan data sehingga hanya memerlukan ruang penyimpanan yang lebih kecil[1]. Kompresi memiliki 2 teknik, antara lain:

Journal of Informatics Management and Information Technology
ISSN 2774-4744 (Media Online)
Vol 2, No 3, Juli 2022
Hal 78−82
https://hostjournals.com/jimat | DOI 10.47065/jimat.v2i3.149

1. Kompresi *lossy* adalah kompresi data yang menghasilkan file data hasil kompresi yang tidak dapat dikembalikan menjadi file data sebelum dikompresi secara utuh. Ketika data hasil kompresi di-decode kembali, data hasil decoding tersebut tidak dapat dikembalikan menjadi sama dengan data asli tetapi ada bagian data yang hilang.
2. Kompresi *lossless* adalah data yang menghasilkan *file* data hasil kompresi yang dapat dikembalikan menjadi *file* data asli sebelum dikompresi secara utuh tanpa perubahan apapun. Kompresi jenis ini ideal untuk kompresi teks. Algoritma yang termasuk dalam metode kompresi lossless diantaranya adalah dictionary coding dan huffman coding.

A. 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:

$$\text{Rasio} = \frac{\text{ukuran file asli}}{\text{ukuran file terkompresi}} \quad (1)$$

Jika dinyatakan dalam prosentase maka dituliskan dalam persamaan berikut:

$$\text{P} = \left(\frac{1-\text{ukuran file terkompresi}}{\text{ukuran file asli}}\right) \times 100\% \quad (2)$$

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[6].

2.2 Algoritma Context Tree Weighting

Dalam Metode *Context Tree Weighting* ini merupakan metode yang memiliki sumber abjad biner. Dengan mewakili simbol x1x2. xt oleh xt. Metode ini menggunkan informasi FSMX dengan konteks pohon yang mewakili sumber. Setiap simpul pohon mewakili konteks. Ketika simbol 0 muncul sebagai kali dan simbol 1 muncul b_s kali pada konteks s, dengan menyimpan angka-angka ini diNode s, Node ini dibagi dua yaitu: 0s dan 1s. Jumlah simbol kali terjadi pada node memiliki hubungan : a0s + a1s = as b0s + b1s = bs. Dengan menghitung probabilitas $P^s(a_s, b_s)$. untuk setiap konteks yang probabilitas bahwa simbol 0 muncul setiap kali dan 0 muncul bs kali pada konteks. Selanjutnya kita menghitung rekursi probabilitas sebagai berikut

γ adalah nilai nyata yang merupakan 0 <γ <1. Kita dapat memperoleh kata-kata kode pengkodean yang sebagai nilai tertimbang Pw$^{\lambda}$ (^{x}penjumlahan ^{t}1) diakar pohon konteks. Dari pohon konteks probabilitasx^{t}1 $^{\lambda.}$ untuk ini x^{t}1 probabilitas oleh aritmatika di semua submodel di update data node. Oleh karna itu waktu dari algoritma adalah O (nD). Perhatikan bahwa kompleksitas waktu untuk pengkodean aritmatika di sini tidak dipertimbangkan. Meskipun probabilitas blok simbol mewakili probabilitas sekuens yang tepat yang dijamin memiliki batas teoretis panjang kode-kata, perlu untuk menggunakan operasi aritmatika floating-point multi-presisi, yang sulit untuk diimplementasikan dan waktu mengkonsumsi tugas

2.3 File Teks

File teks merupakan file 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, angka dan tanda baca. File teks merupakan file komputer yang tersusun atas rangkaian baris teks. Jenis–jenis file teks yang termasuk dalam kategori umumnya berisi rangkaian karakter tanpa informasi format visual. Konten file kategori ini biasanya merupakan catatan atau daftar personal, artikel, buku, dan lain sebagainya. File teks mirip dengan file yang dihasilkan oleh program pengolahan kata yang konten utamanya bersifat tekstual masukan dan keluaran data teks direpresentasikan [8].

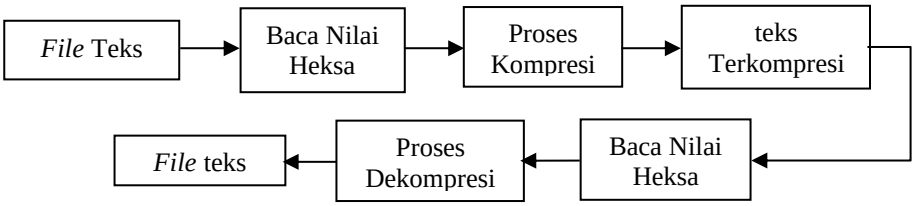
2.4 Unified Modelling Language (UML)

Unified Modelling Language (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML dapat dibuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan class dan operation dalam konsep dasarnya, maka lebih cocok untuk penulisan piranti lunak dalam bahasa berorientasi objek seperti C++, Java, atau VB. NET. Dalam UML terdapat konsep semantik, notasi, dan panduan masing-masing diagram. UML juga memiliki bagian statis, dinamis, ruang lingkup, dan organisasional. UML bertujuan menyatukan teknik-teknik pemodelan berorientasi objek menjadi terstandarisasi dengan lebih baik dan sederhana[9].

Journal of Informatics Management and Information Technology
ISSN 2774-4744 (Media Online)
Vol 2, No 3, Juli 2022
Hal 78−82
https://hostjournals.com/jimat | DOI 10.47065/jimat.v2i3.149

3. HASIL DAN PEMBAHASAN

Algoritma *Context Tree Weighting* merupakan salah satu teknik kompresi lossless yang dapat memperkecil suatu data berdasarkan dengan frekuensi karakter pada objek yang akan dilakukan proses kompresi. Penggunaan algoritma *Context Tree Weighting* akan dilakukan berdasarkan karakter yang sering muncul dan akan memiliki jumlah bit terkecil berdasarkan kode *Context Tree Weighting*, sedangkan karakter yang paling sedikit muncul akan memiliki jumlah bit terpanjang.



Gambar 1. Prosedur Kompresi Dekompresi *File* Teks.

3.1 Penerapan Algoritma Context Tree Weighting

Pada tahapan ini, akan dilakukan kompresi beberapa *string* dengan algoritma *Context Tree Weigthing*. Sebagai sample string akan dikompresi adalah "mayniar", table 1 dibawah adalah ukuran string yang belum dikompresi, dan berdasarkan referensi bahwa yang digunakan hanya 7bit**.**

Tabel 1. Ukuran string sebelum dikompresi

Karakter	Frekuensi	ASCII Biner	Bit	Bit x Frekuensi
A	2	1100001	7	14
I	1	1101001	7	7
M	1	1101101	7	7
N	1	1101110	7	7
R	1	1110010	7	7
Y	1	1111001	7	7
Total Bit				49

Setelah file teks yang berhasil dibaca menjadi *string*, maka selanjutnya akan dilakukan proses kompresi dan dekompresi dengan algoritma *Context Tree Weighting.*

1. Kompresi file teks *Context Tree Weighting*
   Pada *context tree weighting* string akan dibaca biner nya lalu dihitung probabilitas dari masing-masing biner string, dengan menggunakan rumus:

   $P_e(b_{t+1} = 1|b_1^t) = \frac{b+1/2}{a+b+1}$

   $atau$

   $P_e(a,b) = \frac{b+1/2}{a+b+1}$

dimana a adalah jumlah bit 0 dan b adalah jumlah bit 1, sehingga untuk perhitungan sample kita akan menggunakan bit dari biner huruf "a" yaitu 1100001, maka perhitungan dengan rumus diatas menjadi:

$P_e(3,4) = 5/2048$

maka estimasi probabilitas untuk huruf a adalah 0,0024, untuk selanjutnya dapat dilihat pada tabel berikut:

Tabel 2. Estimasi Probabilitas Kompresi

Karakter	Frekuensi	Biner	Estimasi Probabilitas $P_e(a,b)$	Ep *x* Frekuensi
a	2	1100001	$P_e(4,3)$ = 0,0024	0,0048
i	1	1101001	$P_e(3,4)$ = 0,0024	0,0024
m	1	1101101	$P_e(2,5)$ = 0,0044	0,0044
n	1	1101110	$P_e(2,5)$ = 0,0044	0,0044
r	1	1110010	$P_e(3,4)$ = 0,0024	0,0024
y	1	1111001	$P_e(2,5)$ = 0,0044	0,0044
Hasil				0,0228

Journal of Informatics Management and Information Technology
ISSN 2774-4744 (Media Online)
Vol 2, No 3, Juli 2022
Hal 78−82
https://hostjournals.com/jimat | DOI 10.47065/jimat.v2i3.149

Berdasarkan pada gambar dan tabel diatas, didapat nilai bilangan heksadesimal *file* audio WAV. Untuk keperluan hitungan manual penulis hanya mengambil sampel 20 bilangan heksadesimal *file* WAV. Adapun bilangan heksadesimal *file* WAV sampel tersebut adalah sebagai berikut : 52 49 46 46 4A CB 9E 00 57 41 10 00 00 00 01 00 02 00 44 AC.

Lalu dari hasil perhitungan estimasi probabilitas setiap karakter dihitung persentase rasio kompresi yaitu 0,0228 x 49 bit (ukuran sebelum kompresi) adalah 1,1172 ukuran setelah kompresi.

Rasio Kompresi= $\frac{ukuran\ setelah\ kompresi}{ukuran\ sebelum\ kompresi}\ x\ 100\%$

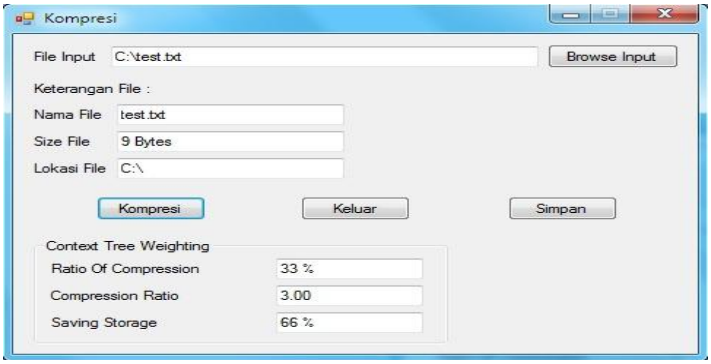
$= \frac{1,1172}{49}\ x\ 100\%$

$= 2,28\%$

maka 2,28% adalah hasil kompresi dari perhitungan estimasi probabilitas dari string “mayniar”, yang berarti sekitar 2,28% ukuran telah dikompresi.

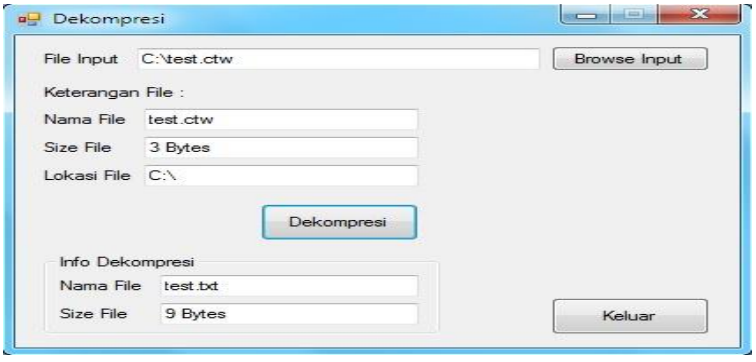
3.2 Pengujian Sistem

1. Pengujian Proses Kompresi
   Proses kompresi pada peujian akan dimulai dengan mencari *file* teks berformat teks yang akan dikompresi, kemudian klik tombol kompresi untuk memulai proses kompresi.



Gambar 1. Hasil Pengujian Kompresi

2. Pengujian Proses Dekompresi
   Proses dekompresi disini akan berfungsi untuk mengolah kembali *file* yang sudah dikompresi agar kembali menjadi ukuran dan format *file* diawal. Berikut adalah tampilan *form* dekompresi.



Gambar 2. Hasil Pengujian Dekompresi

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

Tabel 3. 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

4. KESIMPULAN

Berdasarkan dari penelitian yang telah dilakukan, maka hasil akhir pada penelitian tersebut dapat diambil beberapa kesimpulan. Adapun kesimpulan tersebut Proses kompresi file teks dengan algoritma *Context Tree Weighting* dilakukan

ISSN 2774-4744 (Media Online)
Vol 2, No 3, Juli 2022
Hal 78−82
https://hostjournals.com/jimat | DOI 10.47065/jimat.v2i3.149

dengan membaca teks pada file teks lalu dilakukan proses kompresi dengan ctw. Kompresi file teks dapat dirancang dan dibangun dengan menggunakan aplikasi *Microsoft Visual Studio 2008* dengan menerapkan algoritma *Context Tree Weighting* sehingga diharapkan dapat mempermudah penulis dalam mengompresi ukuran file teks...

REFERENCE

[1] D. Salomon, *Data Compression The Complete Reference FourthEdition*, vol. 53, no. 9. 2007.
[2] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens, "The Context-Tree Weighting Method: Basic Properties," *IEEE Transactions on Information Theory*, vol. 41, no. 3. pp. 653–664, 1995, doi: 10.1109/18.382012.
[3] A. Puspabhuana, S. Kharisma, J. Pangkal, P. Km, and B. Pass, "Perbandingan Tiga Langkah Teknik-Teknik Kompresi Teks," *IT Soc.*, pp. 23–29, 2016, [Online]. Available: http://e-journal.president.ac.id/presunivojs/index.php/Itforsociety/article/view/298.
[4] R. N. Ibrahim, "PERBANDINGAN KOMPRESI FILE MENGGUNAKAN ALGORITMA RUN LENGTH DENGAN TWO LEVEL HOSHING," vol. 1, no. 2, pp. 90–104, 2007.
[5] J. N. Denenberg, E. D. Weinberger, and M. L. Gordon, "DATA COMPRESSION METHOD FOR USE IN A COMPUTERIZED INFORMATIONAL AND TRANSACTIONAL NETWORK," vol. 32, N, 1996.
[6] M. I. Dzulhaq and A. A. Andayani, "Aplikasi Kompresi File Dengan metode Lempel-Ziv-Welchof," *J. Sisfotek Glob.*, vol. 4, no. 1, pp. 1–4, 2014.
[7] D. Salomon and G. Motta, *Handbook of Data Compression*, Fifth Edit. New York: Springer, 2010.
[8] D. A. Yansyah and I. Pendahuluan, "PERBANDINGAN METODE PUNCTURED ELIAS CODE DAN HUFFMAN PADA KOMPRESI FILE TEXT," vol. 2, no. 6, pp. 33–36, 2015.
[9] P. Sulistyorini, "Pemodelan Visual dengan Menggunakan UML dan Rational Rose," vol. XIV, no. 1, pp. 23–29, 2009.