PEMAMPATAN TATA TEKS BERBAHASA INDONESIA DENGAN METODE HUFFMAN MENGGUNAKAN PANJANG SIMBOL BERVARIASI

Tri Yoga Septianto[1], Waru Djuiatno, S.T., M.T.[2], dan Adharul Muttaqin S.T. M.T.[3]
[1]Mahasisawa Teknik Elektro, [2,3]Dosen Teknik Elektro Universitas Brawijaya
Jurusan Teknik Elekto Fakultas Teknik Universitas Brawijaya
Jalan MT. Haryono 167, Malang 65145, Indonesia
E-mail : paralyze.zero_yoga00@yahoo.co.id

Abstrak - Pemampatan data merupakan salah satu upaya untuk memperbesar ruang penyimpanan agar data lebih banyak yang dapat tersimpan atau untuk mempercepat suatu proses pengiriman data. Huffman *Coding* merupakan salah satu metode pemampatan data yang menggunakan frekuensi atau probabilitas kemunculan suatu simbol atau karakter sebagai acuan pemempatan datanya, terutama berkas teks. Berkas teks memiliki kencenderungan perulangan yang sama pada bagian kosakatanya. Dari pemrosesan menggunakan metode Huffman akan menghasilkan *prefixed-code* yang merupakan representasi dari suatu simbol. *Prefixed-code* terpendek diberikan kesimbol yang memiliki kemunculan tertinggi.

Pengujian dilakukan dengan menggunakan panjang simbol 1 sampai 6 karakter dan menggunakan 5 buah berkas berbahasa Indonesia dengan ukuran masing-masing, 2 KB, 4 KB, 6 KB, 8 KB, dan 10 KB.

Hasil pengujian menggunakan rasio perbandingan antara besar berkas asli dengan besar berkas keluaran. Salah satu pengujian dilakukan dengan variasi panjang simbol pada berkas teks sebesar 10 KB akan menghasilkan rasio untuk panjang 2 sebesar 52,8% lama proses 39 detik, panjang 3 sebesar 43,5% lama proses 43 detik, panjang 4 sebesar 35,2% lama proses 30 detik, panjang 6 sebesar 28,8% lama proses 28 detik dan panjang 6 sebesar 23,4% lama proses 25 detik. Pengembangan lebih lanjut disarankan agar aplikasi dapat mengenali bentuk berkas teks yang memiliki properti-properti teks yang kompleks.

Kata kunci : ***Huffman coding, prefixed-code Command-line Interface***

I. PENDAHULUAN

Metode Huffman salah satu metode untuk memampatkan data. Kompresi Huffman mengubah simbol menjadi representasi beberapa byte tertentu yang panjang byte tersebut sesuai dengan prioritas kemunculan simbol. Semakin sering simbol tersebut muncul maka representasi simbolnya akan lebih pendek. Prioritas ini didapatkan dengan cara membuat tree diagram Huffman yang disusun dari simbol-simbol yang ada dalam suatu file yang akan diproses.

Proses parsing huffman pada awalnya hanya terdiri dari 1 karakter/simbol saja. Data yang didapat dipecah per 1 buah simbol yang ada kemudian dicari banyak kemunculan simbol-simbol yang ada pada data tersebut.

Pada saat ini metode huffman telah dikembangkan, metode ini disebut *extended huffman* atau perluasan huffman. Dimana parsing simbol metode ini menggunakan lebih dari 1 buah karakter. Banyak karakter yang digunakan bisa berjumlah 2 atau lebih. Hasil yang didapat dengan menggunakan panjang maksimum simbol yang lebih panjang didapatkan keluaran yang lebih sederhana. Semakin banyak parsing yang berbeda maka semakin panjang pula pohon huffman yang terbentuk. Hal ini dapat memperpanjang kode representasi untuk simbol-simbol yang didapatkan nanti.

Untuk mendapatkan hasil parsing yang efisien yang akan di terapkan pada data teks, dilakukan proses pencarian kemunculan seluruh panjang simbol (mulai dari panjang 1 sampai untuk skripsi ini 6) pada seluruh data teks yang akan diproses. Hal ini dilakukan agar diketahui variabel mana saja yang memiliki kemunculan terbanyak yang mana mendapatkan prioritas dalam pemilihan parsing simbol.

II. TINJAUAN PUSTAKA

A. Kompresi Data

Kompresi data adalah proses pemampatan data sehingga data tersebut hanya perlu ruang penyimpanan yang lebih kecil dan juga dapat menghemat waktu pemindahan data ketempat lainnya. Teori kompresi data ini dikembangkan pertama kali oleh Claude Shannon. Dorongan pengembangan ilmu ini adalah berdasarkan beberapa pertanyaan tentang apa itu informasi, termasuk bagaimana cara penyimpanan dan pengiriman pesan (informasi). Kompresi data terbagi menjadi 2 jenis yaitu *Lossless data compression* (pemampatan data tanpa kehilangan) dan *Lossy data compression* (Pemampatan data berkehilangan). Pada skripsi ini hanya akan dibahas tentang metode *Lossless data compression.*

B. Lossless Data Compression

Teknik pemampatan data ini mampu mengembalikan data seperti aslinya dari file terkompresi tanpa kehilangan informasi apapun. Proses pemempatan data dengan teknik ini biasanya dengan mengurangi redudancy pada informasi didalam data. Salah satu metode kompresi ini yaitu Entropy Coding.

C. Entropy Coding

Dalam teori informasi *Entropy coding* adalah rancangan/skema *Lossless data compression* yang independen dari karakteristik khusus dari media. Salah satu jenis utama *entropy coding* menciptakan dan memberikan unik kode prefix bebas untuk masing-masing simbol unik yang terjadi pada input. Enkoders entropi ini kemudian memampatkan data dengan cara mengganti setiap input simbol yang panangnya telah ditentukan sesuai variable-length prefix bebas keluaran *codeword*. Panjang setiap *codeword* sebanding dengan logaritma negatif dari probabilitas simbol tersebut. Oleh karena itu, simbol yang paling sering terjadi menggunakan kode terpendek.

D. Kompresi Huffman

Kode Huffman adalah algoritma yang menggunakan frekuensi / probabilitas kemunculan dari simbol pada sebuah string sebagai input dan menghasilkan output berupa prefix kode yang mengkodekan string menggunakan bit paling sedikit dari seluruh kemungkinan binary prefix kode yang mungkin. Algoritma ini dikembangkan oleh David A. Huffman pada paper yang ditulisnya sebagai prasyarat kelulusannya di MIT. Kode Huffman salah satu algoritma dasar untuk kompresi data, yang bertujuan untuk mengurangi jumlah bit yang diperlukan untuk merepresentasikan informasi/pesan

Dalam prosesnya, metode Huffman memerlukan 2 buah hal yaitu tabel Huffman (*Huffman Table*) yang digunakan untuk penyimpan simbol dan probabilitas/frekuensi kemunculannya dan pohon kode (*Huffman Tree*) yang digunakan untuk menggambarkan jalur rangkaian pendek dari simbol yang termampatkan. Skema data terkode pada metode ini dibedakan menjadi 2 bagian yaitu kode dengan panjang tetap (*Fixed Length Encoding)* dimana pada metode ini seluruh simbol yang dikode akan mempunyai jumlah digit *bit* yang sama sedangkan untuk kode dengan panjang tidak tetap (*Variable Length Encoding*) memiliki jumlah digit *bit* yang tidak sama pada setiap simbol yang berbeda berdasarkan tingkat nilai dari frekuensi kemunculan simbol tersebut.

- **Huffman Table**

Tabel ini digunakan untuk menyimpan nilai jumlah dari tingkat frekuensi simbol yang sering muncul pada pola rangkaian data. Dalam proses Huffman, simbol harus diketahui banyak kemunculannya pada suatu rangkaian data. Hal ini diperlukan sebelum proses pembentukan *Huffman tree*.

Sebelum data dikode, terlebih dahulu harus diketahui berapa kali jumlah dari simbol tersebut sering muncul dengan cara memeriksa seluruh simbol data, sehingga dapat diperoleh nilai statistiknya, sebagai contoh terdapat statistik frekuensi dari rangkaian data (*Data Stream*) berikut:

Tabel 1 Tabel Huffman

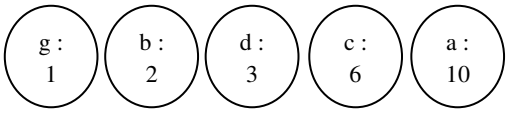
Simbol	Frekuensi
a	10
b	2
c	6
d	3
g	1

- **Huffman Tree**

Kode Huffman adalah string biner yang digunakan untuk mengkodekan setiap karakter di dalam data. Kode Huffman pada dasarnya merupakan kode prefiks (prefiks kode). Kode prefiks biasanya direpresentasikan sebagai pohon biner yang berlabel 0 (cabang kiri) atau 1 (cabang kanan). Rangkaian bit yang terbentuk pada setiap lintasan dari akar sampai tree merupakan kode prefiks untuk karakter yang berpadanan.

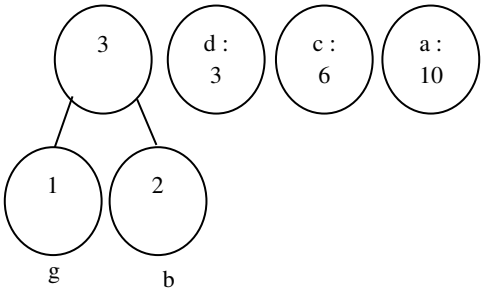
Berikut ini merupakan contoh proses pembentukan pohon Huffman :

Langkah : Urutkan simbol dari yang terkecil



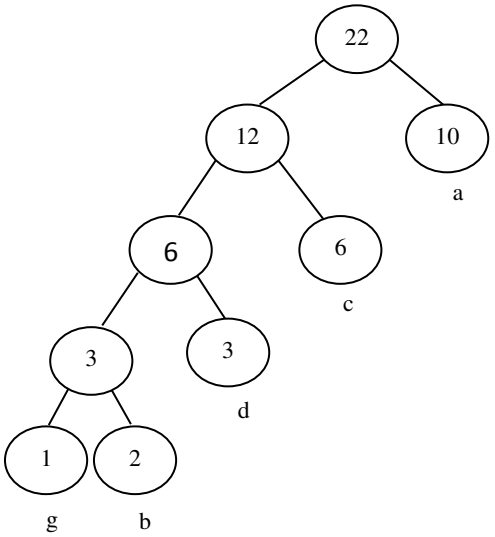
Gambar 1 Karakter/Simbol diurutkan berdasarkan nilai frekuensi

Langkah 2 : Gabungkan 2 buah karakter/simbol terkecil kemudian sisipkan total nilai dari karakter/simbol tersebut.



Gambar 2 Gabungan 2 buah Simbol terkecil

Langkah 3 : Ulangi langkah 2 sampai tersisa 1 pohon Huffman



Gambar 3 Hasil akhir pohon Huffman

- **Metode Huffman Dengan Panjang Simbol Tetap**

Metode ini merupakan pengembangan dari metode Huffman yang telah dikenal sebelumnya. Jika metode Huffman sebelumnya menggunakan representasi 1 buah simbol untuk direpresentasikan dalam *codeword*, Huffman n-based Variable Length simbol menggunakan simbol sebanyak n buah simbol untuk direpresentasikan kedalam codeword. Tujuan metode ini agar hasil kompresi data dapat lebih kecil. Semakin panjang simbol yang direpresentasikan dalam *codeword* maka bit yang digunakan dapat semakin kecil juga. Hal ini dapat menyebabkan data yang akan dimampatkan menjadi lebih kecil dari pada menggunakan 1 buah simbol sabagaimana metode Huffman pada umumnya. Sebagai contoh pada aliran data "aaaabbbbccccdddd" sebagai berikut :

Tabel 2 Panjang 1 Simbol

Simbol	Prob	Kode
a	4	00
b	4	01
c	4	10
d	4	11

Tabel 3 Panjang 2 Simbol

Simbol	Prob	Kode
aa	2	00
bb	2	01
cc	2	10
dd	2	11

Tabel 4 Panjang 4 Simbol

Simbol	Prob	Kode
aaaa	1	00
bbbb	1	01
cccc	1	10
dddd	1	11

Maka hasil dari keluarannya adalah :
1 simbol :
00000000010101011010101011111111
2 simbol : 0000010110101111
4 simbol : 00011011

- **Metode Huffman Dengan Panjang Simbol Bervariasi**

Metode ini adalah pengembangan dari metode panjang simbol tetap dimana ada proses pencarian simbol terbaik yang terprioritaskan untuk terbentuk dari pada simbol yang memiliki nilai kemunculan yang lebih kecil. Proses ini bertujuan agar simbol yang memiliki kemunculan yang lebih tinggi dapat terbentuk sehingga penggunaan *prefixed-code* dapat lebih optimal. Jika kode tersebut banyak digunakan dikarenakan simbol dari kode tersebut banyak terbentuk dapat dikatakan bahwa rasio pemampatan data dapat lebih diperkecil lagi. Sebagai contoh untuk kumpulan karakter "bbbxbbbyaaaxaaay" dengan menggunakan panjang simbol bervariasi 3. Setelah menyeleksi simbol-simbol mana saja yang digunakan dalam parsing maka akan terbentuk tabel sebagai berikut :

Tabel 5 Daftar Simbol Pajang Bervariasi Dengan 3 Variabel

Simbol	Prob	Kode
bbb	2	00
aaa	2	01
x	2	10
y	2	11

Maka hasil dari keluaranya adalah :
3 Simbol dengan variasi : 0010001101100111
16 bit dengan 4 record data

Jika menggunakan metode panjang tetap maka simbol-simbol yang akan digunakan dalam parsing yaitu :

Tabel 6 Daftar Simbol Panjang Tetap Dengan 3 Variabel

Simbol	Prob	Kode
bbb	1	00
xbb	1	01
bya	1	100
aax	1	101
aaa	1	110
y	1	111

Maka hasil keluarannya adalah :
3 Simbol Panjang Tetap : 0001100101110111
16 bit dengan 6 record data

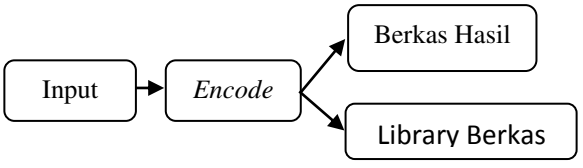
Dengan demikian hasil pemampatan data lebih dapat diperkecil lagi dari pengguaan metode panjang tetap.

III. PERANCANGAN

A. Blok Diagram Sistem

Pada pemampatan data dan pengembalian data ke data asli terdiri dari beberapa langkah yang dapat digambarkan menjadi blok diagram dengan modal seperti pada gambar :

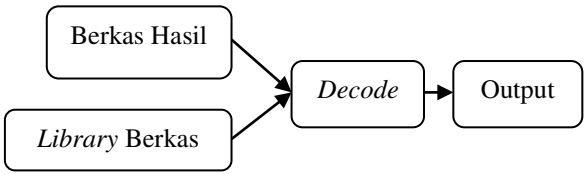
- **Pemampatan Data**



Gambar 4 Diagram Blok Sistem Secara Keseluruhan

1. *Input* : sebagai masukan berkas teks
2. *Encode :* sebagai proses pemampatan data
3. *Output :* sebagai hasil pemampatan yang terdiri dari 2 buah berkas, yaitu Berkas hasil dan *Library* berkas tersebut.

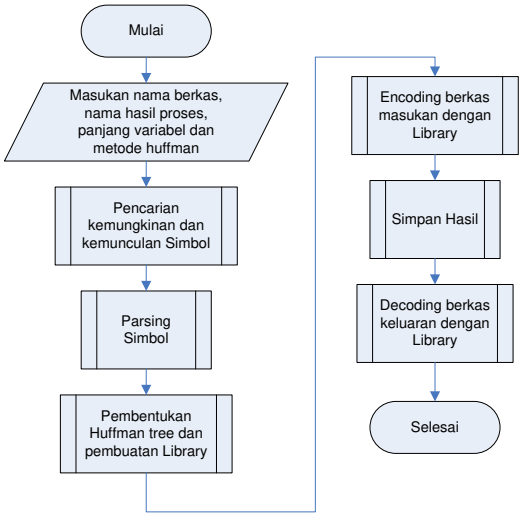
➢ **Pengembalian Data**



1. *Input :* sebagai masukan yang terdiri dari 2 buah berkas, yaitu berkas yang akan diproses dan *library* dari berkas tersebut
2. *Decode :* sebagai proses pengembalian data kebentuk aslinya
3. *Output :* sebagai hasil pengembalian data yang berupa data asli dari berkas yang diproses.

B. Perancangan Perangkat Lunak

Perangkat lunak akan dirancang untuk melakukan beberapa proses dalam pemampatan data yang akan dilakukan. Proses tersebut adalah pencarian.



Gambar 5 Flowchart Proses *encode* Kerja Program

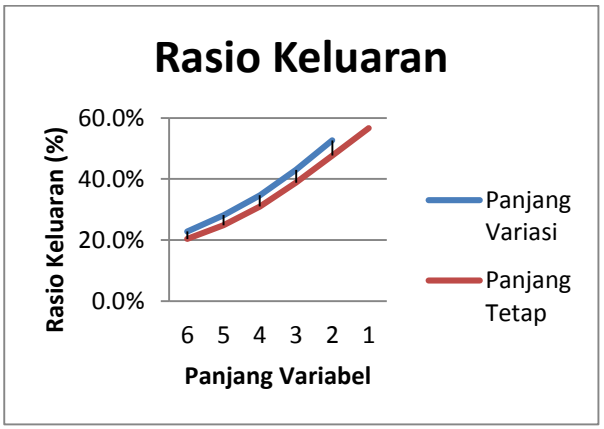
Untuk proses *decoding* teks akan dilakukan proses berupa pembacaan ulang berkas keluaran dan *library*. Kemudian akan dicocokkan *stream* data binary yang dihasilkan dengan *library* yang sesuai dengan berkas tersebut.

IV. PENGUJIAN DAN ANALISIS

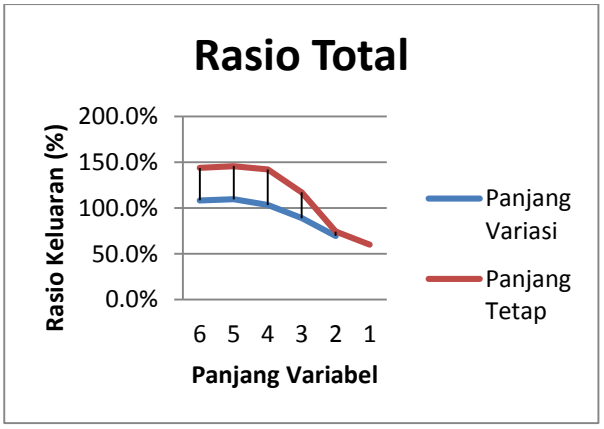
Tabel 6 Hasil Pengujian Dengan Teks Bahasa Indonesia

Nama File	Ukuran (KB)	Metode	Panjang	Size (KB)		Rasio (%)	
				Keluaran	Library	Keluaran	Total
berita3.txt	6	Variasi	6	1,37	5,13	22,8%	108,3%
			5	1,69	4,88	28,2%	109,5%
			4	2,08	4,12	34,7%	103,3%
			3	2,58	2,75	43,0%	88,8%
			2	3,16	1,01	52,7%	69,5%
		Tetap	6	1,22	7,41	20,3%	143,8%
			5	1,49	7,25	24,8%	145,7%
			4	1,86	6,66	31,0%	142,0%
			3	2,33	4,69	38,8%	117,0%
			2	2,86	1,59	47,7%	74,2%
			1	3,4	0,19	56,7%	59,8%

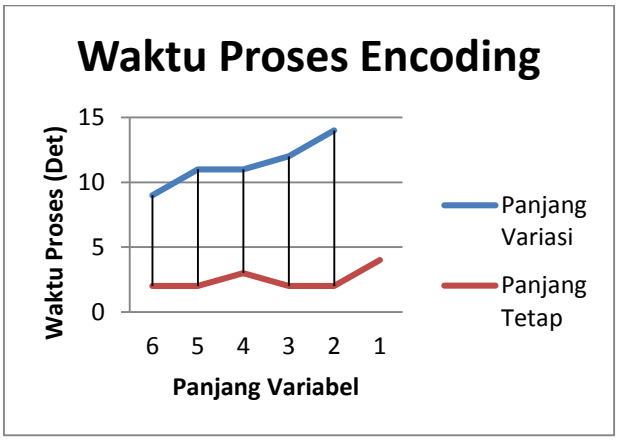
Semakin tinggi panjang simbol yang digunakan maka rasio kompresi yang dihasilkan akan lebih kecil.



Gambar 6 Perbandingan Antara Panjang Tetap Dengan Variasi



Gambar 7 Perbandingan Total Antara Berkas Keluaran Dengan Library terhadap Panjang Tetap Dan Bervariasi



Gambar 7 Perbandingan Waktu Proses

V. Kesimpulan Dan Saran

1. Kesimpulan

Berdasarkan hasil pengujian dan analisis maka didapatkan beberapa kesimpulan sebagai berikut :

1. Penambahan panjang variabel akan memperkecil rasio keluaran kompresi senilai 50-53% (Panjang 2), 40-44% (Panjang 3), 29-35% (Panjang 4), 23-29% (Panjang 5), dan 19-23% (Panjang 6) daripada 57-58% (Panjang 1).
2. Panjang variabel yang bervariasi dapat memperkecil total ukuran berkas keluaran senilai 63-117% (Rata-rata panjang 2 sampai panjang 6) daripada panjang tetap yang bernilai 66-141% (Rata-rata panjang 2 sampai panjang 6)

2. Saran

Saran yang dapat diberikan untuk pengembangan lebih lanjut :

1. Menggunakan pemrosesan paralel untuk proses pencarian kemungkinan dan kemunculan simbol.
2. Proses *decoding* terkendala masalah penggunaan memori, dengan menggunakan fungsi / algoritma yang lebih komplek dapat memperlancar masalah ini.
3. Proses penulisan *library* masih menggunakan operasi 1-byte sehingga belum dapat mengoptimalkan metode Huffman yang mengakibatkan berkas *library* menjadi besar. Dengan menggunakan operasi bit akan lebih dapat memperkecil berkas *library*.

DAFTAR PUSTAKA

[1] Ata Amrullah, Isbat Uzzin N, Rizky Yuniar H. 2007. *Kompresi Dan Enkripsi SMS Dengan Metode Huffman Kode Dan Algoritma Enigma*. http://digilib.its.ac.id/ITS-Undergraduate-3100010041069/14636 (diakses 24 Juli 2014)

[2] Tri, Arya Prabawa. 2008. *Kode Huffman*. Bandung : Teknik Informatika ITB

[3] Permana, Raditya. 2008. *Implementasi Huffman Coding untuk Kompresi SMS Menggunakan J2ME*. Malang : Depdiknas Joint

[4] Prihantanto, Muchamad Yoga. 2010. *ENCODING* HUFFMAN DENGAN PANJANG SIMBOL BERVARIASI. Malang : Teknik Elektro Universitas Brawijaya.

[5] I Made Joni, Budi Raharjo 2008, *Pemrograman C dan Implementasinya : Edisi Kedua*. Bandung : INFORMATIKA

[6] Sianipar, I Ketut Wiryajati, Herry S. Mangiri. 2013, *Pemrograman & Strucktur Data C*. Bandung : INFORMATIKA

[7] Huffman Compression, Wikipedia http://en.wikipedia.org/wiki/Huffman_coding (diakses pada tanggal 24 Juli 2014)