**MENGIMPLEMENTASI ALGORITMA KOMPRESI LZW PADA DATABASE SERVER**

**Abdillah Mufki Auzan Mubin**

Teknik Informatika - Universitas Widyatama

*Email:* [*abdillah.mufki123@widyatama.ac.id*](mailto:abdillah.mufki123@widyatama.ac.idom)

# ABSTRAK

Paper ini membahas tentang implementasi algoritma kompresi LZW untuk mengkompresi data yang tersimpan dalam *web server*. Hal ini dilakukan untuk memperkecil ukuran data yang akan disimpan di *web server*. Algoritma *LZW* yang dirancang menggunakan bahasa pemrograman *PHP* dan sistem manajemen basis data *MySQL*. Proses kompresi dilakukan pada saat data (artikel) yang disimpan pada *web server* dan didekompresi setiap artikel tersebut akan dibaca.

Algoritma *LZW* yang merupakan *dictionary based compression* sangat efektif mengkompresi data teks yang memiliki banyak pola huruf, kata ataupun kalimat yang berulang, semakin banyak pembendaharaan gabungan string dalam *dictionary* tambahan pada suatu *plaintext* maka semakin baik hasil kompresi *plaintext* tersebut. Hasil implementasi algoritma *LZW* pada *web server* penyimpanan artikel menjadi bertambah banyak, terlihat dari nilai persentase penghematan penyimpanan setiap artikel berkisar antara 11,6656% sampai dengan 16,6656% untuk perhitungan nilai rata-rata persentase penghematan

200 sampai dengan 1.000 karakter pertama dari sepuluh buah artikel acak.

***Kata Kunci: informasi, internet, kompresi, LZW, dictionary***

# PENDAHULUAN

Banyaknya informasi yang disimpan dalam *database server* membutuhkan ukuran media penyimpanan yang besar. Keterbatasan kapasitas media penyimpanan data menyebabkan tidak semua jumlah informasi dapat disimpan pada media penyimpanan data tersebut. Solusi yang dapat dilakukan untuk mengatasi masalah tersebut diantaranya: menghapus beberapa data yang dianggap tidak penting, menambahkan ukuran media penyimpanan data, atau mengkompresi setiap informasi yang disimpan.

Dengan pertimbangan efisiensi biaya dan pentingnya dokumentasi data, maka penggunaan kompresi data dapat dijadikan solusi untuk mengatasi masalah di atas. Implementasi kompresi data memiliki kelebihan yaitu dapat mengurangi

konsumsi ruang media penyimpanan data dengan memadatkan ukuran setiap data yang disimpan. Algoritma kompresi yang akan digunakan dalam penelitian ini adaah algoritma kompresi *Lempel-Ziv- Welch* (*LZW*) menggunakan bahasa pemrograman PHP.

# KAJIAN TEORI

**Kompresi Data**

Kompresi DataKompresi data telah dimanfaatkan dalam berbagai aspek multimedia. gambar, audio maupun video yang kita dapatkan dari web merupakan file yang telah terkompresi. TV-HD juga merupakan hasil kompresi MPEG-2, kebanyakan modem juga melakukan proses kompresi data, dan beberapa sistem berkas otomatis mengkompresi data ketika data tersimpan (Blelloch, 2013). Sehingga alasan kita membutuhkan kompresi data yaitu karena tegnologi ini memudahkan kita mendapat informasi atau data yang berkualitas dan tidak menghabiskan ruang penyimpanan kita.

Kompresi data merupakan seni atau ilmu yang memberikan informasi dalam bentuk yang padat menggunakan struktur yang ada pada data asli dengan tujuan untuk mengurangi jumlah bit pada data yang terkompresi (Sayood, 2016). Data dapat berupa karakter dalam file text, maupun nomor yang merupakan representasi dari filegambar. Kompresi data terdiri dari 2 komponen yaitu sebuah algoritma encoding yaitu algoritma yang membuat pesan atau textmenjadi terkompresi sehingga jumlah bit yang ada dalam file tersebut menjadi lebih sedikit, dan algoritma decoding yaitu algoritma yang merekonstruksi file terkompresi menjadi file asli.

Kompresi data adalah proses mengkonversi atau mengubah data input atau data asli menjadi data yang memiliki ukuran yang lebih kecil. Kompresi data menjadi popular dikarenakan dua hal yakni perilaku manusia yang suka menumpuk data dan tidak suka membuangnya dan ketidak senangan manusia dalam menunggu transfer data yang lama (Salomon, 2007). Dalam melakukan kompresi dilakukan 2 metode yaitu Kompresi dan Decompresi. Kompresi merupakan metode atau algoritma untuk mencapai tujuan kompresi pada data asli. Dibalik setiap algoritma terdapat ide, model matematika atau teknik implementasi untuk mencapai tujuan dalam mengkompresi data dengan mempertimbangkan aspek efisiensi dan keefektifan kompresi. Dekompresi merupakan teknik untuk mengembalikan data yang telah dikompresi supaya data dapat kembali ke dalam bentuk yang semula atau data awal (Pu, 2016)



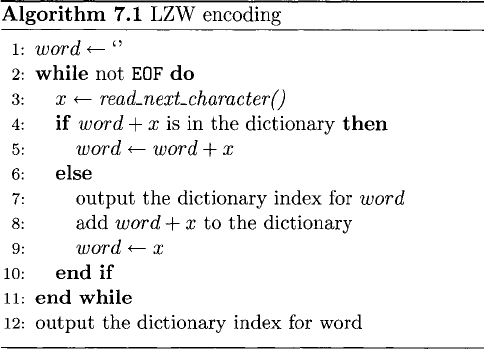




## **Algoritma LZW**

*LZW* merupakan algoritma *dictionary based compression*, yaitu algoritma kompresi yang berbasiskan kamus. Pendekatan yang digunakan adalah mengidentifikasi adanya pola perulangan karakter[1]. Kelebihan teknik kompresi *LZW* adalah kecepatan waktu kompresi yang sangat singkat dengan tingkat kompresi yang

cukup baik, yaitu persentase penghematan mencapai sekitar 60.2% ± 28.9[3]. Berikut ini adalah *pseudocode* proses kompresi dan dekompresi pada *LZW* secara umum[1]:

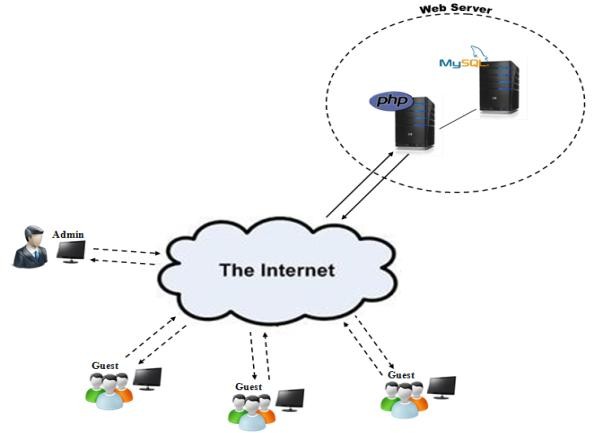


**Sisstem manajemen basis data MySQL**

*Data Base Management System* (DBMS) adalah sekumpulan program yang digunakan untuk mendefinisikan, mengatur administrasi, dan memproses basis data dan aplikasi-aplikasi yang terkait dengan basis data. Sebuah DBMS adalah perangkat yang digunakan untuk membuat struktur dan beroperasi pada data yang berada di antara basis data[4]. MySQL adalah sebuah sistem manajemen basis data relasional yang digunakan pada arsitektur *client*/*server*. MySQL menjadi sistem basis data *open source* yang paling populer dan paling sukses di dunia. Popularitas ini sebagian besar dikarenakan karena kehandalan, kinerja, dan kemudahan penggunaannya [5]. Sebuah *RDBS* (Sistem basis data relasional) adalah penyimpanan data dan layanan pengambilan data berdasarkan model relasional data, seperti yang diusulkan oleh E.F. Codd pada tahun 1970. Sistem ini adalah mekanisme penyimpanan standar untuk data terstruktur[5].

# PEMBAHASAN

Pada penelitian ini dirancang implementasi teknik kompresi data LZW yang diterapkan pada *webserver*. Pada Gambar 2. ditunjukkan arsitektur implementasi kompresi pada *webserver*.



Gambar 2. Arsitektur implementasi.

Artikel yang diunggahkan oleh *User admin* akan dikompresi pada *webserver* untuk kemudian hasil kompresi tersebut disimpan dalam *database server*. Pada saat tejadi pembacaan artikel, maka artikel dalam bentuk hasil kompresi akan didekompresi hingga muncul dalam bentuk teks aslinya.

## **Dictionary LZW**

*Dictionary LZW* merupakan sekumpulan *record* kamus yang dibutuhkan pada perancangan sistem kompresi menggunakan algoritma *LZW*. Penentuan urutan *record* kamus sangatlah penting, karena urutan *record* kamus mempengaruhi hasil keluaran kompresi. Pengurutan didasarkan dari urutan karakter dasar yang paling umum hingga karakter yang jarang digunakan. Adapun tabel *dictionary LZW* ditunjukkan pada Tabel 1 (Lampiran). Jumlah *record* yang dibuat berjumlah

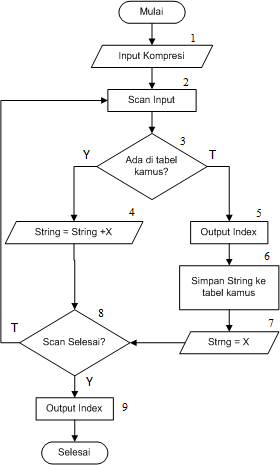
94 indeks. Data *Record* ini memuat karakter- karakter yang sering digunakan.

## **Pengkodean UTF-8 dan Character Set**

Pengkodean *UTF-8* merupakan suatu proses mengkodekan hasil keluaran kompresi yang berbentuk bilangan heksadesimal, menjadi suatu urutan biner. Selanjutnya urutan biner tersebut ditranslasikan dalam bentuk karakter. Fungsi pengkodean *UTF-8* ini adalah agar sistem dapat melakukan *encoding* suatu nilai menjadi urutan biner dan *decoding* urutan biner tersebut menjadi nilai tertentu. Adapun tabel translasi karakter pengkodean UTF-8 ini ditunjukkan pada Tabel 2 (Lampiran).

## **Diagram Alir Kompresi LZW**

Gambar 3. menunjukkan diagram alir kompresi LZW yang diterapkan pada sistem.



Gambar 3. Diagram alir proses kompresi *LZW*

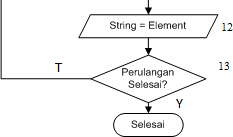
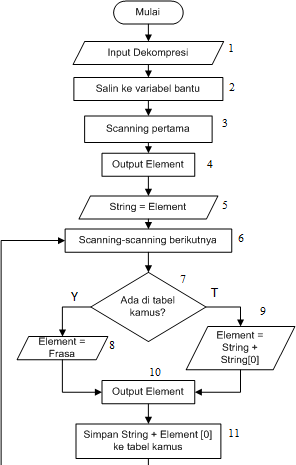
Alur pada diagram alir di atas adalah sebagai berikut:

1. Data masukan berupa *plaintext*.
2. Pembacaan masukan perkarakter.
3. pembacaan tabel kamus pada *field* yang bersesuaian dengan *field* nilai *String*.
4. Jika terdapat dalam kamus maka *String* digabung dengan X (karakter yang sedang terbaca).
5. Menambahkan *Output* indeks *String* baru.
6. Menyimpan indeks *String* ke tabel kamus sebagai kamus tambahan yang baru.
7. Memasukkan X sebagai data *String*.
8. Memeriksa apakah semua karakter telah dibaca.
9. Mengeluarkan *Output* indeks *String* ke variabel

*compressed*.

## **Diagram Alir Kompresi LZW**

Gambar 4 menunjukkan diagram alir kompresi LZW yang diterapkan pada sistem.



Gambar 4. Diagram alir proses dekompresi LZS

Alur pada diagram alir di atas adalah sebagai berikut:

1. Data masukan berupa compressedtext).
2. Menyalin data masukan ke variabel array.
3. Scanning isi variabel array indeks pertama.
4. Mengeluarkan Output Element ke variabel ‘plain’
5. Menyalin variabel Element ke variabel String
6. Membaca isi variabel array indeks-indeks selanjutnya.
7. Memeriksa Adakah record dengan yang nilai dari field indeks yang merupakan nilai desimal dari kode isi variabel array indeks
8. Jika ya, Element = Frase.
9. Jika tidak, Element = String sebelumnya digabung dengan karakter pertama dari String.
10. Mengeluarkan Output Element ke variabel ‘plain’.
11. Menggabungkan dan menyimpan String sebelumnya dengan karakter pertama dari Element ke tabel kamus sebagai kamus tambahan yang baru.

# PENGUJIAN DAN ANALISA

## **Pengujian**

Pengujian proses kompresi dan dekompresi LZW pada sistem ini dilakukan pada sejumlah teks artikel yang dengan jumlah karakter 200, 400, 600,

800 dan 1000. Masing-masing berjumlah sepuluh buah artikel.

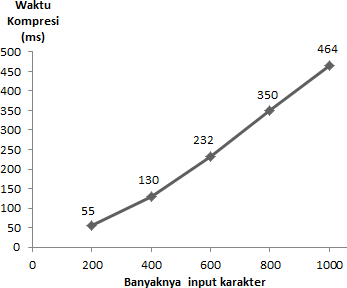
Tabel 3. menunjukkan rata-rata hasil pengujian kompresi dan dekompresi LZW dengan parameter pengujian *Compression Time* (CT), *Compression Ratio* (CR), *Compression Factor* (CF), *Saving Percentage* (SP), dan *Decompression Time* (DT).

Tabel 3. Rata-rata Hasil Pengujian

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **NO** | **INPUT** | **CT**  **(ms)** | **CR**  **(kali)** | **CF**  **(kali)** | **SP**  **(%)** | **DT**  **(ms)** |
| 1 | 200 | 55 | 0,883 | 1,132 | 11,67 | 33 |
| 2 | 400 | 130 | 0,876 | 1,143 | 12,43 | 63 |
| 3 | 600 | 232 | 0,860 | 1,164 | 14,02 | 86 |
| 4 | 800 | 350 | 0,848 | 1,179 | 15,17 | 121 |
| 5 | 1000 | 464 | 0,833 | 1,201 | 16,70 | 169 |

## **Analisa**

Grafik pada Gambar 5. menunjukkan hasil waktu kompresi.

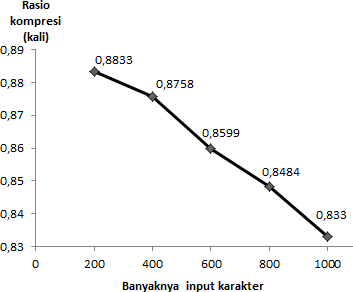


Gambar 5. Grafik rata-rata waktu kompresi

Pada grafik di atas bahwa semakin

bertambahnya jumlah karakter maka waktu kompresi semakin lama.

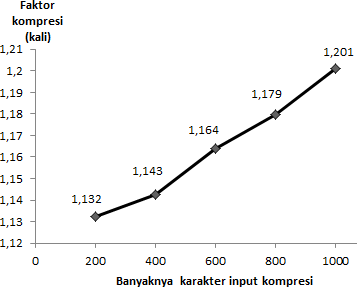
Grafik pada Gambar 6. menunjukkan hasil rasio kompresi.



Gambar 6. Grafik rata-rata rasio kompresi

Pada grafik di atas ditunjukkan bahwa rasio kompresi semakin bertambahnya jumlah karakter maka rasio kompresi semakin kecil.

Grafik pada Gambar 7. menunjukkan hasil faktor kompresi.



Gambar 7. Grafik rata-rata faktor kompresi

# SIMPULAN

## **Simpulan**

Simpulan dari hasil penelitian yang telah dilakukan adalah sebagai berikut:

1. Teknik kompresi LZW dapat diimplementasikan untuk mengkompresi teks artikel yang akan disimpan dalam *web server*.
2. Hasil pengujian kompresi yang dilakukan pada artikel dengan jumlah karakter antara 200 sampai dengan 1000 adalah:
   1. waktu kompresi yang dibutuhkan untuk mengkompresi mencapai kisaran waktu antara 55 ms sampai dengan 464 ms.
   2. Rasio kompresi yang diperoleh adalah 0,883 kali sampai dengan 0,833 kali.
   3. Faktor kompresi yang mencapai kisaran 1,132 kali sampai dengan 1,201 kali.
   4. Penghematan hasil kompresi mencapai kisaran 11,67% sampai dengan 16,70%.
   5. Waktu dekompresi yang mencapai kisaran waktu antara 33 ms sampai dengan 169 ms. Data ini menunjukkan bahwa waktu dekompresi lebih cepat dibanding waktu kompresi.

# DAFTAR PUSTAKA

1. Pu, I. M. (2006). *Fundamental Data Compression.* Burlington: Elsevier.
2. Sayood, K. (2006). *Introduction to Data Compression.* San Francisco: Morgan Kaufmann.
3. Linawati, & Panggabean, H. P. (2004). Perbandingan Kinerja Algoritma Kompresi, LZW, dan DMC pada Berbagai Tipe File. *INTEGRAL*, 9.
4. Taylor, A. G. (2003). *SQL for Dummies 5th Edition.* Indianapolis: Wiley Publishing.