Jurnal Ilmu Komputer, Teknologi Dan Informasi
Vol 2, No. 1, Januari 2024, Hal 1 - 10
ISSN: **2963-0169** (Online - Elektronik)
https://journal.grahamitra.id/index.php/jurikti

Implementasi Algoritma Burrows-Wheeler Transformation Pada Aplikasi Kumpulan Manga Berbasis Android

Dodo Hanggoro

Program Studi Teknik Informatika, Fakultas Ilmu Komputer Dan Teknologi Informasi, Universitas Budi Darma, Medan, Indonesia
Jl. Sisingamangaraja No. 338, Medan, Sumatera Utara, Indonesia
Email: hanggorododo@gmail.com

Abstrak− *Aplikasi Manga merupakan aplikasi yang digunakan untuk membaca komik Jepang secara online. Aplikasi Manga akan menampilkan isi dari suatu cerita dalam bentuk gambar-gambar yang tidak sedikit karena alur cerita yang begitu panjang. Satu gambar memiliki ukuran yang besar, untuk satu chapter dari satu manga memiliki 15 gambar, dalam satu manga memiliki gambar hingga ratusan chapter. Karena banyaknya gambar yang akan disimpan maka membutuhkan ruang penyimpanan yang besar. Keterbatasan kapasitas dari media penyimpanan online membuat gambar manga yang disimpan menjadi lebih sedikit. Untuk memperkecil ukuran suatu gambar dapat menggunakan proses pemampatan. Pemampatan atau kompresi merupakan proses yang bertujuan untuk memperkecil ukuran suatu gambar. Penelitian ini akan menggunakan Metode Burrows-Wheeler Transformation untuk mengompresi (mengecilkan) ukuran suatu orisinalitas gambar. Metode ini dapat meningkatkan efektifitas teknik kompresi. Metode BWT akan mengkompresi kumpulan gambar manga yang ada di aplikasi android, dimana sering banyaknya keluhan dari para penggunanya yang di karenakan gambar manga yang ada di aplikasi android penyedia layanan online kumpulan manga tersebut terlalu besar sehingga banyak memakan kapasitas yang ada di android bagi para penggunanya.*

Kata Kunci: Kompresi, Gambar*, Burrows, Wheeler, Transformation.*

Abstract− The Manga application is an application used to read Japanese comics online. The Manga application will display the contents of a story in the form of quite a few pictures because the story line is so long. One image has a large size, one chapter of one manga has 15 images, one manga has hundreds of images for chapters. Because there are so many images to be stored, it requires a large storage space. The limited capacity of online storage media means that fewer manga images are stored. To reduce the size of an image, you can use a compression process. Compression or compression is a process that aims to reduce the size of an image. This research will use the Burrows-Wheeler Transformation Method to compress (reduce) the size of an original image. This method can increase the effectiveness of compression techniques. The BWT method will compress the collection of manga images in the Android application, where there are often many complaints from users because the manga images in the Android application, the online service provider's collection of manga, are too large, so it takes up a lot of capacity on Android for its users.

Keywords: Compression, Image, Burrows, Wheeler, Transformation.

1. PENDAHULUAN

Perkembangan teknologi di zaman ini semakin memudahkan manusia untuk mengakses informasi kapanpun dan dimanapun. Dengan berkembangnya teknologi maka semakin banyak sumber-sumber yang menyediakan layanan-layanan *online* yang dapat diakses dari mana saja terutama dari *android* [1][2]. Jumlah pengguna layanan-layanan ini semakin bertambah dari waktu ke waktu sehingga kapasitas penyimpanan yang diberikan dibatasi. Keterbatasan kapasitas media penyimpanan dari pihak penyedia layanan *online*, membuat data yang disimpan menjadi lebih sedikit. Hal ini pula yang menjadi suatu permasalahan baru yang tidak dapat di hindari oleh pihak penyedia layanan maupun pihak penggunanya sendiri [3]. Data informasi yang utuh namun memiliki ukuran yang kecil sangat di perlukan, sehingga data tidak membebani media penyimpanan dan media penyimpanan dapat menyimpan lebih banyak data. Salah satu aplikasi yang menggunakan media penyimpanan *online* adalah aplikasi kumpulan manga. Aplikasi manga merupakan aplikasi yang digunakan untuk membaca komik Jepang [4][5][6][7].

Aplikasi manga akan menampilkan isi dari suatu cerita dalam bentuk gambar-gambar yang tidak sedikit karena alur cerita yang begitu panjang. Satu gambar memiliki ukuran yang besar, untuk satu *chapter* dari satu manga memiliki 15 gambar bahkan lebih, Karena banyaknya gambar yang akan disimpan maka membutuhkan ruang penyimpanan yang besar. Keterbatasan kapasitas dari media penyimpanan *online* membuat gambar manga yang disimpan menjadi lebih sedikit. Salah satu cara yang digunakan untuk memecahkan masalah di atas adalah melakukan teknik kompresi. Kompresi adalah teknik yang bertujuan untuk memperkecil ukuran penyimpanan. Dengan melakukan kompresi dapat dihasilkan ukuran *size* yang lebih kecil dari ukuran asli tanpa mengurangi keaslian dari gambar tersebut. Dan memori terpakai di media penyimpanan menjadi lebih sedikit dan pada proses menggunakan aplikasi tidak lagi membutuhkan proses yang cukup lama. Salah satu metode kompresi citra yaitu metode *Burrows-Wheeler Transformation (BWT).*

Metode ini dapat meningkatkan efektifitas teknik kompresi yang sudah ada pada saat ini. Dalam penelitian ini metode BWT akan mengkompresi kumpulan gambar manga yang ada di aplikasi *android*, dimana sering banyaknya keluhan dari para penggunanya yang di karenakan gambar manga yang ada di aplikasi *android* penyedia layanan *online* kumpulan manga tersebut terlalu besar sehingga banyak memakan kapasitas yang ada di *android* bagi para penggunanya. Pada penelitian sebelumnya yang berjudul *"Analisa Perbandingan Algoritma Huffman dengan Algoritma Transformasi Burrows-Wheeler pada Kompresi Citra menggunakan metode Eksponensial"* menyimpulkan bahwa algoritma BWT menghasilkan performa waktu yang kurang namun memiliki kinerja kompresi yang tinggi terutama pada format JPEG[8][9][10]

2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Adapun tahapan penelitian yang di gunakan dalam penelitian ini adalah:

a. Studi Pustaka
Pengumpulan referensi dengan mencari jurnal, buku, tulisan ilmiah, *e-book,* maupun artikel lain yang berhubungan dengan kompresi, gambar, Algoritma *Burrows-Wheeler Transformation*.
b. Analisis Masalah
Menganalisa permasalahan dan melakukan penerapan yang diambil dalam data yang diteliti, serta menyelesaikan dengan Algoritma yang berkaitan dengan penerapan Algoritma *Burrows-Wheeler Transformation*.
c. Perancangan
Dalam tahap ini penulis melakukan perancangan aplikasi pada kompresi gambar.
d. Pengujian Sistem
Hasil penelitian pada tahap ini akan diuji sesuai dengan parameter yang telah ditentukan di batasan masalah dan memperbaiki kesalahan (*error*) yang muncul berdasarkan Algoritma *Burrows-Wheeler Transformation.*
e. Implementasi
Merupakan langkah untuk melakukan validasi dan hasil akhir (*output*) yang diperoleh dari sistem yang dirancang setelah melakukan evaluasi ketepatan terhadap kinerja sistem untuk membuat kesimpulan dari topik yang di kaji.
f. Dokumentasi
Dalam tahap ini penulis akan mendokumentasikan hasil dari analisa yang telah di lakukan.

2.2 Kompresi

Kompresi citra adalah proses penempatan citra yang bertujuan untuk mengurangi duplikasi data pada citra sehingga memori yang digunakan untuk merepresentasikan citra menjadi lebih sedikit dari pada representasi citra semula [11] [12][13]. Ada dua teknik yang dapat dilakukan dalam melakukan prposes kompresi diantaranya adalah [11]:

a. *Lossless Compression*
Lossless Compression merupakan teknik kompresi PDF dimana hasil dekompresi dari *file* yang terkompresi sama dengan *file* aslinya, tidak ada informasi yang hilang. Sayangnya, ratio kompresi *file* metode ini sangat rendah. Banyak aplikasi yang memrlukan kompresi tanpa cacat, seperti pada aplikasi radiografi. Kompresi *file* hasil diagnose medis atau gambar satelit, dimana kehilangan gambar sekecil apapun akan menyebabkan hasil yang tidak diharapkan.
b. *Lossy compression*
Lossy Compression merupakan teknik kompresi *file* yang akan menghilangkan beberapa informasi, dimana hasil dekompresi dari *file* yang terkompresi tidak sama dengan *file* aslinya, tetapi masih bisa ditolerir oleh persepsi mata. Mata tidak dapat membedakan perubahan kecil pada gambar. Metode ini menghasilkan rasio kompresi yang lebih tinggi daripada metode *lossless*.

2.3 Algoritma *Burrows-Wheeler Transform (BWT)*

Burrows dan *Wheeler* merilis sebuah laporan penelitian pada tahun 1994 berjudul "*A Block Sorting Lossless Data Compression Algorithm"* menyajikan algoritma kompresi data berdasarkan algoritma penyortiran. Algoritma *Burrows-Wheeler* menggunakan blok data dan memprosesnya menggunakan skema penyortiran. *Burrows Wheeler* merupakan teknik kompresi *Reversible* atau *Lossless,* sehingga proses *decoding* akan dapat mengembalikan data aslinya. Algoritma *Burrows Wheeler* dapat dibagi menjadi tiga tahapan, yaitu [13][14][15]:

a. Transformasi *Burrows-Wheeler* merubah sedemikian rupa letak *symbol input* data sehingga *symbol* yang sama saling berdekatan.
b. *Global Structur Transform* (GST) yang mentransformasi redundansi *local* menjadi global menggunakan *List of Updated Table (LUT).*
c. Pengkodean (*Entropy Coding*-EC) merupakan tahap terakhir dari teknik kompresi untuk memampatkan data.

Sebagian besar metode kompresi beroperasi dalam modus *streaming*, di mana kode *input a byte* atau beberapa *byte*, proses mereka, dan berlanjut sampai akhir-*of -file* dirasakan. Metode *Burrows-Wheeler* (BW) bekerja dalam mode blok, di mana input stream dibaca blok demi blok dan setiap blok adalah dikodekan secara terpisah sebagai satu string. Metode ini disebut sebagai blok penyortiran. Metode *Burrows-Wheeler* bekerja dengan baik pada gambar, suara, dan teks, dan dapat mencapai rasio kompresi yang sangat tinggi (1 bit per *byte* atau bahkan lebih baik). Metode *Burrows-Wheeler* di mulai dengan *string* S dari n simbol dan berlanjut ke *string* L dan string lain yang memenuhi dua kondisi:

a. Setiap wilayah L akan cenderung memiliki konsentrasi hanya beberapa simbol. Cara lain mengatakan ini, jika simbol s ditemukan pada posisi tertentu di L, maka kejadian lainnya s yang mungkin ditemukan di dekatnya. Properti ini berarti bahwa L dapat dengan mudah dan efisien dikompresi dengan metode bergerak-ke-depan mungkin dalam kombinasi dengan *Adaptive Huffman*. Ini juga berarti bahwa metode BWT akan bekerja dengan baik hanya jika n besar (di setidaknya beberapa ribu simbol per string).

b. Hal ini dimungkinkan untuk merekonstruksi *string* S asli dari L (data yang lebih sedikit mungkin dibutuhkan untuk rekonstruksi, selain L, tapi tidak banyak). Istilah matematika untuk simbol berebut adalah permutasi, dan mudah untuk menunjukkan bahwa string n simbol memiliki n! (Diucapkan "n faktorial") permutasi. Ini adalah sejumlah besar bahkan untuk nilai yang relatif kecil dari n, sehingga permutasi tertentu yang digunakan oleh BW harus dipilih dengan hati-hati. BWT hasil kode di langkah-langkah berikut:
   1. *String* L diciptakan, oleh *encoder*, sebagai permutasi dari S. Beberapa informasi dilambangkan dengan I2.
   2. *Decoder* itu berbunyi output stream dan *decode*. Hasilnya adalah sebuah *string* L dan variabel I.
   3. Kedua L dan saya digunakan oleh *decoder* untuk merekonstruksi *string* asli S.

Sebelum melakukan proses kompresi dengan metode *Burrows-Wheeler Transform* (BWT), Terdapat proses yang harus dilewati terlebih dahulu. Proses tersebut adalah data citra yang akan melalui proses kompresi harus diubah dari bentuk piksel dua dimensi ke bentuk piksel satu dimensi berurutan dengan cara pemindaian *zig-zag*. Setelah bentuk dari data citra diubah lalu dilanjutkan dengan metode BWT. Misalkan sebuah vektor 1 dimensi berurutan *p* telah didapat seperti berikut [14]:

$$p = [\ 3\ 2\ 5\ 3\ 1\ 4\ 2\ 6\ ] \quad (1)$$

Vektor *p* disalin ke baris pertama, atau disebut sebagai indeks 0. Urutan selanjutnya adalah mengurutkan dengan cara merubah susunan perputaran ke kiri untuk setiap baris selanjutnya. Tahap pertama dari BWT ditampilkan melalui Tabel 1.

Tabel 1. Tahap Pertama *Forward Transform* dari BWT

Index	**Step 1**							
0	3	2	5	3	1	4	2	6
1	2	5	3	1	4	2	6	3
2	5	3	1	4	2	6	3	2
3	3	1	4	2	6	3	2	5
4	1	4	2	6	3	2	5	3
5	4	2	6	3	2	5	3	1
6	2	6	3	2	5	3	1	4
7	6	3	2	5	3	1	4	2

Tahap selanjutnya adalah setiap baris disusun secara leksikografi (berdasarkan kamus). Tahap terakhir dari proses BWT adalah *output* yang terdiri dari *index* terakhir pada langkah sebelumnya.

Tabel 2. Tahap Kedua dan Ketiga *Forward Transform* dari BWT

index	**Step 2**							
0	3	4	2	6	3	2	5	3
1	2	5	3	1	4	2	6	3
2	2	6	3	2	5	3	1	4
3	3	1	4	2	6	3	2	5
4	3	3	5	3	1	4	2	6
5	4	2	6	3	2	5	2	1
6	5	3	1	4	2	6	3	2
7	6	3	2	5	3	1	4	2

Vektor *p* asli muncul di baris kelima pada Tabel 2 dan *output* dari BWT berada pada kolom terakhir, dinyatakan dengan:

$$L = [3\ 3\ 4\ 5\ 6\ 1\ 2\ 2] \quad (2)$$

Dengan *index* = 4. Hasil dapat ditulis sebagai BWT = *[index,]*, dimana *L* adalah output dari *Burrows Wheeler Transform* dan index menggambarkan lokasi asli dari urutan leksikografi.

2.3.1 Proses Dekompresi *Burrows-Wheeler Transform (BWT)*

Metode BWT adalah metode transformasi yang dapat dibalik (*reverse*) sehingga vektor asli dapat dikembalikan dari *output* yang dihasilkan oleh BWT tersebut. Di bentuk *reverse* transform hanya output BWT *L* dan *index* yang dibutuhkan untuk mengembalikan vektor urutan aslinya. Cara membentuk tabel *reverse*: *For i* = 1, …, *n* – 1,

a. Langkah (3 *i* – 2): Letakkan kolom n di depan kolom 1, …, *n* – 1,.
b. Langkah (3 *i* – 2): Urutkan panjang string *i* yang dihasilkan secara leksikografi.
c. Langkah (3 *i* – 2): Letakkan list yang terurut di kolom pertama pada tabel.

Output dari *reverse transform* juga berada di *index* 4, itulah mengapa *index* dimanfaatkan untuk merepresentasikan *output* dari *forward transform* dan urutannya adalah, yang mana memiliki urutan asli dari vektor *p*.

Setelah proses dekompresi dengan metode BWT selesai, data citra diubah kembali dari bentuk piksel satu dimensi berurutan ke bentuk piksel dua dimensi dengan cara pemindaian *unzig-zag* sehingga citra hasil dekompresi dapat ditampilkan kembali.

3. HASIL DAN PEMBAHASAN

Pada tahap ini akan dijelasakan secara umum cara kerja algoritma *Burrows-Wheeler Transformation* dalam kompresi gambar. Algoritma *Burrows-Wheeler Transformation* merupakan teknik kompresi dengan cara menggunakan blok data dan memprosesnya menggunakan skema penyortiran.

3.1 Implementasi Algoritma *Burrows-Wheeler Transform*

Dalam penelitian ini, peneliti akan membahas 2 proses yaitu proses kompresi dan proses dekompresi metode *Burrows-Wheeler Transform* dan peneliti akan mengkompresi sebuah gambar alur cerita manga yang berbentuk *grayscale*. Analisa perbandingan pada algoritma BWT dilakukan dengan tujuan agar mengetahui kualitas dari algoritma BWT. Jenis gambar yang akan dikompresi adalah berformat *jpeg* dengan ekstensi .jpg, Dengan resolusi piksel 733 x 1125 seperti yang ditunjukkan pada Gambar 1 berikut.

Gambar 1. Citra *Grayscale*

3.1.1 Kompresi *Algoritma Burrows-Wheeler Transfor*

Pada penelitian ini akan dilakukan analisis dan perancangan perangkat lunak untuk kompresi gambar dengan menggunakan metode *Burrows-Wheeler Transform* (BWT). Metode BWT memiliki beberapa tahapan untuk proses kompresi. Berikut ini adalah langkah-langkah proses kompresi dengan metode BWT:

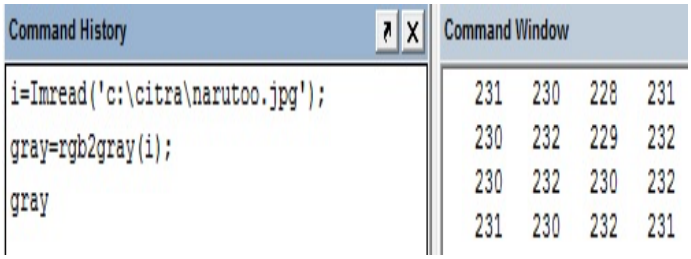
a. Data citra terlebih dahulu diubah dari bentuk dua dimensi menjadi satu dimensi.
b. Kemudian bentuk sebuah matriks dimana nilai baris matriks sama dengan panjang data citra yang berukuran satu dimensi.
c. Baris pertama matriks diisi dengan data citra asli yang berukuran satu dimensi, lalu untuk baris selanjutnya diisi dengan data citra sebelumnya yang dilakukan perubahan, yaitu perputaran susunan data citra ke arah kiri.
d. Setelah semua baris terisi, urutkan setiap baris pada matriks secara leksikografi atau berdasarkan kamus.
e. Ambil kolom terakhir sebagai *output* dari BWT. Simpan baris dengan nilai data citra awal dengan simbol *I (index)* agar dapat melakukan proses *inverse* jika ingin dikembalikan ke bentuk semula. Hitung rasio dan waktu kompresi pada *output* BWT.

Sebelum melakukan proses kompresi dengan metode *Burrows-Wheeler Transform* (BWT), terdapat proses yang harus dilewati terlebih dahulu. Proses tersebut ialah data citra yang akan melalui proses kompresi harus diubah dari bentuk piksel dua dimensi ke bentuk piksel satu dimensi berurutan dengan cara pemindaian *zig-zag*. Setelah bentuk dari data citra diubah lalu dilanjutkan dengan metode BWT seperti berikut:

Gambar 2. Citra *Grayscale Resolusi* 4x4

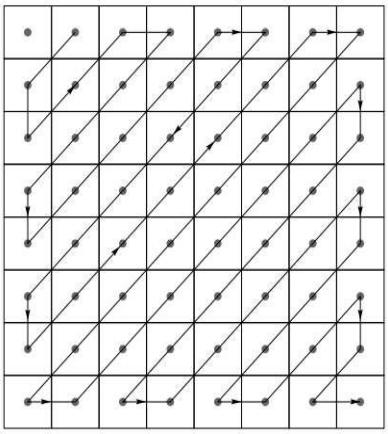
Jurnal Ilmu Komputer, Teknologi Dan Informasi
Vol 2, No. 1, Januari 2024, Hal 1 - 10
ISSN: **2963-0169** (Online - Elektronik)
https://journal.grahamitra.id/index.php/jurikti

Gambar 2 merupakan hasil citra *grayscale* dari gambar 1 yang diambil sebagai contoh kasus dengan resolusi 4x4 dengan size 87KB dan bit *depth* = 24. Matriks dari citra di atas yang diperoleh dari aplikasi Matlab yang menghasilkan nilai kuantisasi sebagai berikut:



Gambar 3. Hasil Nilai Kuantisasi dari Aplikasi Matlab

Nilai kuantisasi yang telah didapat dari Matlab selanjutnya akan dilewatkan dalam bentuk *zig-zag* seperti pada gambar 4 berikut:



Gambar 4. Proses *Zig - zag*

Setelah bentuk dari data citra diubah lalu dilanjutkan dengan metode BWT. Misalkan sebuah vektor 1 dimensi berurutan *p* telah didapat hasilnya seperti berikut:
p^0 = [231 230 230 230 232 228 231 229 231 231 230 230 232 232 232 231]

Vektor p^0 disalin ke baris pertama, atau disebut sebagai *index* 0. Urutan selanjutnya adalah mengurutkan dengan cara merubah susunan perputaran dari kiri ke kanan atau susunan angka di depan di pindahkan ke belakang untuk setiap baris selanjutnya. Tahap pertama BWT ditampilkan melalui Tabel 3.

Tabel 3. Tahap Pertama *Forward Transform* dari BWT

Index	Step I
p^0	231 230 230 230 232 228 231 229 231 231 230 230 232 232 232 231
p^1	230 230 230 232 228 231 229 231 231 230 230 232 232 232 231 231
p^2	230 230 232 228 231 229 231 231 230 230 232 232 232 231 231 230
p^3	230 232 228 231 229 231 231 230 230 232 232 232 231 231 230 230
p^4	232 228 231 229 231 231 230 230 232 232 232 231 231 230 230 230
p^5	228 231 229 231 231 230 230 232 232 232 231 231 230 230 230 232
p^6	231 229 231 231 230 230 232 232 232 231 231 230 230 230 232 228
p^7	229 231 231 230 230 232 232 232 231 231 230 230 230 232 228 231
p^8	231 231 230 230 232 232 232 231 231 230 230 230 232 228 231 229
p^9	231 230 230 232 232 232 231 231 230 230 230 232 228 231 229 231
p^{10}	230 230 232 232 232 231 231 230 230 230 232 228 231 229 231 231
p^{11}	230 232 232 232 231 231 230 230 230 232 228 231 229 231 231 230
p^{12}	232 232 232 231 231 230 230 230 232 228 231 229 231 231 230 230
p^{13}	232 232 231 231 230 230 230 232 228 231 229 231 231 230 230 232
p^{14}	232 231 231 230 230 230 232 228 231 229 231 231 230 230 232 232
p^{15}	231 231 230 230 230 232 228 231 229 231 231 230 230 232 232 232

Tahap selanjutnya adalah setiap baris disusun secara leksikografi (Kecil ke Besar) berdasarkan kamus. Tahap terakhir dari proses BWT adalah *output* yang terdiri dari *index* terakhir pada langkah sebelumnya. Berikut tahap Kedua dan Ketiga dari BWT ditampilkan melalui Tabel 4.

Tabel 4. Tahap Kedua dan Ketiga *Forward Transform* dari BWT

Index	Step II	Step III
p^0	228 231 229 231 231 230 230 232 232 232 231 231 230 230 230 **232**	**232**
p^1	229 231 231 230 230 232 232 232 231 231 230 230 230 232 228 **231**	**231**
p^2	230 230 230 232 228 231 229 231 231 230 230 232 232 232 231 **231**	**231**
p^3	230 230 232 228 231 229 231 231 230 230 232 232 232 231 231 **230**	**230**
p^4	230 230 232 232 232 231 231 230 230 230 232 228 231 229 231 **231**	**231**
p^5	230 232 228 231 229 231 231 230 230 232 232 232 231 231 230 **230**	**230**
p^6	230 232 232 232 231 231 230 230 230 232 228 231 229 231 231 **230**	**230**
p^7	231 229 231 231 230 230 232 232 232 231 231 230 230 230 232 **228**	**228**
p^8	231 230 230 230 232 228 231 229 231 231 230 230 232 232 232 **231**	**231**
p^9	231 230 230 232 232 232 231 231 230 230 230 232 228 231 229 **231**	**231**
p^{10}	231 231 230 230 230 232 228 231 229 231 231 230 230 232 232 **232**	**232**
p^{11}	231 231 230 230 232 232 232 231 231 230 230 230 232 228 231 **229**	**229**
p^{12}	232 228 231 229 231 231 230 230 232 232 232 231 231 230 230 **230**	**230**
p^{13}	232 231 231 230 230 230 232 228 231 229 231 231 230 230 232 **232**	**232**
p^{14}	232 232 231 231 230 230 230 232 228 231 229 231 231 230 230 **232**	**232**
p^{15}	232 232 232 231 231 230 230 230 232 228 231 229 231 231 230 **230**	**230**

Vektor *p* asli muncul di *index* ke-8 pada Tabel 3.3, dan *output* dari BWT berada pada *step* III, dinyatakan dengan:

p = **[232 231 231 230 231 230 230 228 231 231 232 229 230 232 232 230]**

Dengan *index* = 8, Hasil dapat ditulis sebagai $BWT = [Index, p]$, dimana *p* adalah *output* dari *Burrows-Wheeler Transform* dan *index* pada step II menggambarkan lokasi asli dari urutan Leksikografi.

Step 1:
Input: [228 229 230 231 232]
Y = [228 229 230 231 232]
Step 2:
Y_1 = [228 229 230 231 **232**]
Posisi *index* = **4**
Y_1 = [232 228 229 230 231]
Y_2 = [**231** 228 229 230 232]
Posisi *index* = **0**
Y_2 = [231 228 229 230 232]
Lakukan dengan Langkah yang sama seperti Y_1 dan Y_2 sampai dengan Y_{16}
Step 3:
Untuk mendapatkan rasio kompresi, metode *Burrows Wheller Transform* memanfaatkan metode lain. Perlu diketahui metode BWT tidak mengkompresi data, akan tetapi mentransformasikan data *input* menggunakan teknik penyortiran karakter, sehingga karakter yang sama akan saling berdekatan. Maka dari itu di penelitian ini metode BWT ini memanfaatkan metode *Huffman.* Berikut ini adalah langkah-langkah mendapatkan nilai rasio kompresinya:

1. Data matriks tersebut di ubah menjadi vektor, sehingga didapat vektor
   = **[4 0 0 3 1 3 0 1 3 0 4 2 2 4 0 3]**.
2. Tentukan nilai frekuensi kemunculan. Hasilnya adalah:

Tabel 5. Tabel Frekuensi

Nilai Keabuan	Frekuensi
0	5
1	2
2	2
3	4
4	3

3. Urutkan frekuensi dari yang terkecil ke frekuensi yang terbesar. Data setelah diurutkan: [1 2 4 3 0].
4. Membuat pohon biner berdasarkan frekuensi.
5. Membuat kode *Huffman* dari pohon *Huffman*.

Tabel 6. Kode *Huffman*

Nilai	Kode Huffman	Panjang Kode
0	0	1
1	1000	4
2	1001	4

Nilai	Kode Huffman	Panjang Kode
3	101	3
4	11	2

Proses penghitungan hasil kompresi dengan cara mengalihkan panjang kode dengan jumlah frekuensi pada setiap data, kemudian hasil perkalian dijumlahkan seperti tabel di bawah.

Tabel 7. Perhitungan Kompresi BWT

Nilai	Frekuensi	Panjang Kode	Frekuensi x Panjang Kode
0	5	1	5
1	2	4	8
2	2	4	8
3	4	3	12
4	3	2	6
Jumlah			**39**

Dari tabel 7 di atas dapat dilihat bahwa hasil kompresi adalah sebesar 39 bit, maka akan dilakukan perhitungan rasio kompresi seperti di bawah ini:

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

$= \frac{39}{87}$ x 100% = 0.4482 x 100%

= 44,82%

Gambar 5. Gambar 4x4 Setelah Dikompresi

3.1.2 Dekompresi *Algoritma Burrows-Wheeler Transform*

Metode BWT adalah metode transformasi yang dapat didekompresi (*reverse*) sehingga vektor asli dapat dikembalikan dari *output* yang dihasilkan oleh BWT tersebut. Di bentuk *reverse* transform hanya *output* BWT *L* dan *index* yang dibutuhkan untuk mengembalikan vektor urutan aslinya. Tahap *reverse* BWT ditampilkan pada Tabel 8.

Tabel 8. Tahap Pertama *Reverse*/Dekompresi *Transform* dari BWT

Index	*(i = 1)*			*(i = 2)*			*(i = 3)*		
	a	***b***	***c***	***a***	***b***	***c***	***a***	***b***	***c***
p^o	232	228	228..232	232228	228231	228231	232228231	228231229	228231229
p^1	231	229	229..231	231229	229231	229231	231229231	229231231	229231231
p^2	231	230	230..231	231230	230230	230230	231230230	230230230	230230230
p^3	230	230	230..230	230230	230230	230230	230230230	230230232	230230232
p^4	231	230	230.231	231230	230230	230230	231230230	230230232	230230232
p^5	230	230	230..230	230230	230232	230232	230230232	230232228	230232228
p^6	230	230	230..230	230230	230232	230232	230230232	230232232	230232232
p^7	228	231	231..228	228231	231229	231229	228231229	231229231	231229231
p^8	231	231	231..231	231231	231230	**231230**	231231230	231230230	**231230230**
p^9	231	231	231..231	231231	231230	231230	231231230	231230230	231230230
p^{10}	232	231	231..232	232231	231231	231231	232231231	231231230	231231230
p^{11}	229	231	232..229	229231	231231	231231	229231231	231231230	231231230
p^{12}	230	232	232..230	230232	232228	232228	230232228	232228231	232228231

Index	***(i = 1)***			***(i = 2)***			***(i = 3)***		
	a	***b***	***c***	***a***	***b***	***c***	***a***	***b***	***c***
p^{13}	232	232	232..232	232232	232231	232231	232232231	232231231	232231231
p^{14}	232	232	232..232	232232	232232	232232	232232232	232232231	232232231
p^{15}	230	232	232.230	230232	232232	232232	230232232	232232232	232232232

Lakukan proses *Reverse*/Dekompresi *Transform* dari BWT sampai ke tahap 16.

Tabel 9. Tahap Keempatbelas *Reverse*/Dekompresi *Transform* dari BWT

Index	***(i = 16)***		
	a	***b***	***c***
p^{o}	23222823122923123123 02302322322322312312 30230231	22823122923123123023 02322322322312312302 30231231	22823122923123123023 02322322322312312302 30231231
p^{1}	23122923123123023023 22322322312312302302 31231228	22923123123023023223 22322312312302302312 31228231	22923123123023023223 22322312312302302312 31228231
p^{2}	23123023023023222823 12292312312302302322 32232231	23023023023222823122 92312312302302322322 32230231	23023023023222823122 92312312302302322322 32230231
p^{3}	23023023023222823122 92312312302302322322 32230231	23023023222823122923 12312302302322322322 30231230	23023023222823122923 12312302302322322322 30231230
p^{4}	23123023023223223223 12312302302302322282 31231232	23023023223223223123 12302302302322282312 31232228	23023023223223223123 12302302302322282312 31232228
p^{5}	23023023223223223123 12312302302322322322 30231230	23023223223223223123 12302302322322322302 31230230	23023223223223223123 12302302322322322302 31230230
p^{6}	23023023223223223123 12302302302322282312 31232228	23023223223223223123 02302302322282312312 32228230	23023223223223223123 02302302322282312312 32228230
p^{7}	22823122923123123023 02322322322312312302 30231231	23122923123123023023 22322322312312302302 31231228	23122923123123023023 22322322312312302302 31231228
p^{8}	23123123023023023222 82312292312312302302 32232232	23123023023023222823 12292312312302302322 32232231	**23123023023023222823 12292312312302302322 32232231**
p^{9}	23123123023023223223 22312312302302302312 28231231	23123023023223223223 12312302302302322282 31231232	23123023023223223223 12312302302302322282 31231232
p^{10}	23223123123023023023 22282312292312312302 30232232	23123123023023023222 82312292312312302302 32232232	23123123023023023222 82312292312312302302 32232232
p^{11}	22923123123023023223 22322312312302302312 31228231	23123123023023223223 22312312302302302312 28231231	23123123023023223223 22312312302302302312 28231231
p^{12}	23023222823122923123 12302302322322322302 31230230	23223123123023023023 22282312292312312302 30232232	23223123123023023023 22282312292312312302 30232232
p^{13}	23223223123123023023 02322282312292312282 30230232	23223223123123023023 02322282312292312282 30230232	23223223123123023023 02322282312292312282 30230232
p^{14}	23223223223123123023 02302322282312202322 28230230	23223223223123123023 02302322282312202322 28230230	23223223223123123023 02302322282312202322 28230230
p^{15}	23023223223223123123 02302302322282312312 32228230	22923123123023023223 22322312312302302312 31228231	22923123123023023223 22322312312302302312 31228231

Setelah proses Dekompresi dengan metode BWT selesai, Data citra diubah kembali dengan cara pemindaian *unzig-zag* sehingga citra hasil Dekompresi dapat ditampilkan kembali ke nilai awal. Maka setelah di Dekompresi nilai awal berada di *(i = 16) index ke-8* **2312302302302322282312292312312302302322322322231** sesuai dengan nilai kuantisasi awal. Berdasarkan hasil dekompresi diatas dapat dilihat nilai awal pixel citra sample 4x4 menghilangkan string semula. Matriks nilai pixel dekompresi dapat dilihat pada gambar berikut.

231	230	228	231
230	232	229	232
230	231	230	232
231	230	232	231

Gambar 6. Matriks Nilai *Pixel* Dekompresi

Kemudian nilai *pixel* dekompresi berikut diubah ke citra hasil dekompresi. Citra hasil dekompresi dapat dilihat pada gambar 8.

Gambar 8. Gambar Hasil Dekompresi

4. KESIMPULAN

Kesimpulan yang dapat diambil setelah melakukan penerapan algoritma *Burrows-Wheeler Transformation* pada aplikasi kumpulan manga berbasis *Android* untuk kompresi gambar manga adalah yaitu proses kompresi dan dekompresi pada kumpulan gambar manga berbasis *Android* dilakukan dengan menggunakan blok data dan memprosesnya menggunakan skema penyortiran pada algoritma *Burrows-Wheeler Transformation*. Dengan menerapkan algoritma *Burrows-Wheeler Transformation* pada perancangan aplikasi kumpulan manga berbasis *Android* untuk kompresi gambar manga, maka besarnya *size* gambar pada setiap *chapter* manga yang dikompresi dan dekompresi dapat dimanfaatkan sehingga besarnya *size* gambar dapat diperkecil.

REFERENCES

[1] S. M. B. Pandia, "Kompresi File Dokumen Menggunakan Algoritma Boldi-Vigna Codes," *KOMIK (Konferensi Nas. Teknol. Inf. dan Komputer)*, vol. 6, no. 1, pp. 596–602, 2023.
[2] S. Simanjuntak, "Implementasi Metode Taboo Code Untuk Kompresi File Video," *J. Sains dan Teknol. Inf.*, vol. 1, no. 3, pp. 79–84, 2022.
[3] R. Krasmala, A. Budimansyah, and U. T. Lenggana, "Kompresi Citra Dengan Menggabungkan Metode Discrete Cosine Transform (DCT) dan Algoritma Huffman," *J. Online Inform.*, vol. 2, no. 1, p. 1, Jul. 2017, doi: 10.15575/join.v2i1.79.
[4] Y. T. HARAHAP, "PERBANDINGAN KOMPRESI DATA TEKS DENGAN METODE LZY, LZAP DAN LZW."
[5] M. R. Ramadhan, "Analisa Perbandingan Algoritma Run Length Encoding Dengan Burrows-Wheeler Transform Dalam Kompresi File Video," *KOMIK (Konferensi Nas. Teknol. Inf. dan Komputer)*, vol. 6, no. 1, pp. 322–332, 2023.
[6] Y. Sihura, T. Zebua, and H. Hutabarat, "Kinerja Algoritma Yamamoto's Recursive Code dan Algoritma Fixed Length Binary Encoding pada Kompresi File PDF," *J. Comput. Syst. Informatics*, vol. 3, no. 4, pp. 303–312, 2022.
[7] R. Syahputra, "Peningkatan Rasio Kompresi Algoritma RLE Menggunakan Transormasi bwt," *J. Sains dan Teknol.*, vol. 1, no. 2, pp. 116–119, 2021.
[8] I. R. Lubis, "Menggunakan Metode Eksponensial," vol. 16, pp. 382–384, 2017.
[9] G. Hasibuan, "Analisa Kombinasi Algoritma Burrows Wheeler Transform dan Adaptive Huffman Coding untuk Kompresi Citra," *Bull. Multi-Disciplinary Sci. Appl. Technol.*, vol. 1, no. 2, pp. 34–40, 2022.
[10] A. S. Harahap, "Analisis Dan Implementasi Kompresi File Citra Menggunakan Algoritma Burrows Wheeler Transform," *J. Sains dan Teknol. Inf.*, vol. 1, no. 1, pp. 6–12, 2021.
[11] C. T. Utari, P. Studi, M. Teknik, U. S. Utara, and K. Citra, "IMPLEMENTASI ALGORITMA RUN LENGTH ENCODING UNTUK PERANCANGANAPLIKASI KOMPRESI DAN DEKOMPRESI," vol. V, no. 2, pp. 24–31, 2016.
[12] S. B. Ginting, "Perbandingan Algoritma Yamamoto's Recursive Code Dan Additive Code Dalam Kompresi File Video," *KOMIK (Konferensi Nas. Teknol. Inf. dan Komputer)*, vol. 5, no. 1, 2021.
[13] A. N. Latief, L. Haryanto, A. Lawi, A. N. Latief, L. Haryanto, and A. Lawi, "Analisis Transformasi Burrows Wheeler untuk Kompresi Data Analysis of Burrows-Wheeler Transform for Data Compression."

[14] I. Lubis, P. Tarigan, and N. Sitompul, "Analisa Perbandingan Kompresi Citra Menggunakan Metode Discrete Cosine Transform (Dct) Dan Burrows Wheeler Transform (Bwt)," *Pelita Inform.*, vol. 16, pp. 285–287, 2017.
[15] Hendri, "Kompresi Citra dari Format BMP ke Format PNG," *Time*, vol. III, no. 1, pp. 27–31, 2014.