Menebak *Password* RAR Menggunakan RAR *Password Recovery Magic* dengan Algoritma *Brute Force*

Arina Listyarini Dwiastuti (13512006)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
arinalistyarini@students.itb.ac.id

***Abstrak*—Makalah ini membahas salah satu strategi algoritma, yaitu algoritma *brute force*. Algoritma *brute force* ini digunakan pada berbagai macam *software*, mulai dari *solver* permainan macam Sudoku atau Futoshiki, sampai pada RAR *Password Recovery Magic*. Tujuan penggunaan algoritma *brute force* adalah untuk memecahkan suatu masalah dengan sangat sedeharna namun dengan cara yang jelas. Di makalah ini, akan dibahas bagaimana kerja *software* RAR *Password Recovery Magic* yang menggunakan algoritma *brute force*.**

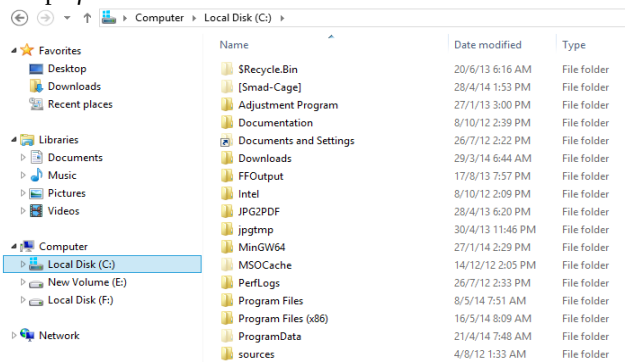
***Kata Kunci*—Algoritma *Brute Force*, RAR, WinRAR, *password*.**

I. PENDAHULUAN

Semakin majunya perkembangan dunia, semakin banyaknya penggunaan komputer. Semakin banyak penggunaan komputer, maka akan makin banyak *file-file* yang disimpan pada komputer. Mengapa? Karena apabila kita bekerja dengan komputer, kita akan menyimpan hasil pekerjaan kita tersebut pada komputer. Pekerjaan yang kita simpan tersebut disimpan di dalam komputer dalam bentuk *file* yang memiliki banyak jenis format. Beberapa jenis format yang terdapat di komputer, yaitu:

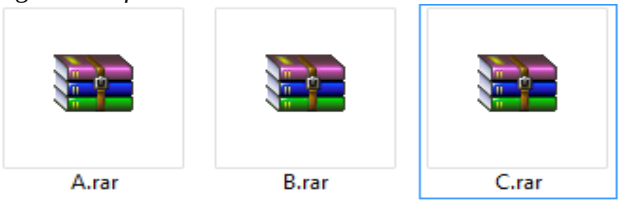
- .doc, merupakan *file* hasil dari *software* pengolah kata *Microsoft Word*
- .docx, merupakan *file* hasil dari *software* pengolah kata *Microsoft Word*
- .ppt, merupakan *file* hasil dari *software* presentasi *Microsoft PowerPoint*
- .pptx, merupakan *file* hasil dari *software* presentasi *Microsoft PowerPoint*
- .c, merupakan *file* yang berupa *source code* dengan bahasa C
- .cpp, merupakan *file* yang berupa *source code* dengan bahasa C++
- .java, merupakan *file* yang berupa *source code* dengan bahasa java
- .jpg, merupakan *file* yang berupa gambar
- .gif, merupakan *file* yang berupa gambar
- .png, merupakan *file* yang berupa gambar
- .txt, merupakan *file* yang berisi teks
- .xls, merupakan *file* hasil dari *software spreadsheet Microsoft Excel*
- .xlsx, merupakan *file* hasil dari *software spreadsheet Microsoft Excel*
- .rar, merupakan satu atau lebih *file* yang diarsipkan menggunakan *software* yang bernama *WinRAR*, bisa juga menggunakan *software* selain *WinRAR*.
- dan lain-lain

Banyaknya *file* yang terdapat dalam komputer akan sulit dicari jika tidak dikelompokkan. Biasanya, *file-file* ini akan dikelompokkan dengan menyimpan beberapa *file* ke dalam *folder*. Sehingga, dalam suatu direktori terdapat beberapa *folder*.



Gambar 1 – *Folder-folder* yang berisi *file-file*

Sekarang adalah zamannya menggunakan internet. Berbagi *file* tidak dapat hanya dilakukan dengan mengkopi *file* tersebut dengan *flash-disk* maupun *compact-disk*, namun kita dapat membagikannya di internet. Biasanya, *file* yang ingin diberikan jumlahnya tidak hanya satu, jumlahnya bisa beberapa *file*. Agar tidak repot mengungguh *file-file* tersebut dengan mengungguh *file* tersebut satu-satu, maka orang-orang akan mengarsipkan *file-file* tersebut dalam satu arsip. Biasanya arsip tersebut menggunakan WinRAR yang akan menghasilkan *file* berekstensi .rar.

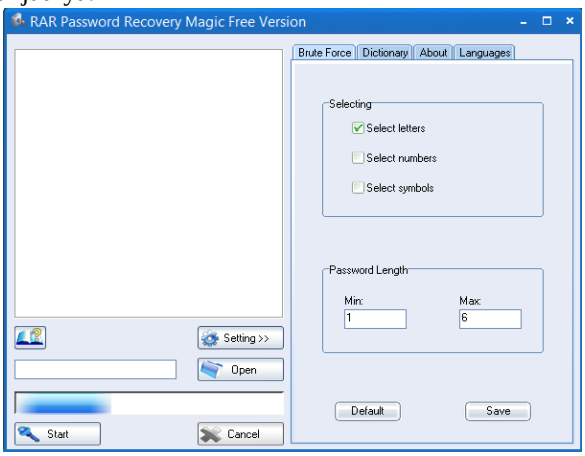


Gambar 2 – *File* dengan format .rar yang dapat berisi

satu atau lebih *file*

Untuk menjaga dan menjamin keamanan *file* tersebut agar tidak jatuh ke tangan orang yang salah, maka orang yang memiliki arsip *file* .rar tersebut biasanya akan memberi *password* pada .rar yang ia miliki. *Password* atau kata sandi adalah kumpulan karakter yang digunakan untuk memverifikasi sesuatu, dapat digunakan juga sebagai pengenal.

Ada kalanya kita ingin membuka *file* .rar yang diberi *password* tetapi kita tidak mengetahui *password* tersebut. Ada salah satu cara untuk membobol *password* .rar, yaitu dengan menggunakan *software* RAR *Password Recovery*. *Software* ini mengimplementasikan algoritma *brute force*. Penjelasan lebih lanjut akan dibahas pada sub-bab selanjutnya.



Gambar 3 – Tampilan *Software* RAR *Password Recovery Magic* (Sumber: dhafintutorial.blogspot.com)

II. DASAR TEORI

A. *Algoritma Brute Force*

Untuk mencari solusi dari suatu masalah, kita dapat menggunakan algoritma *brute force*. Algoritma *brute force* adalah sebuah pendekatan untuk memecahkan suatu masalah yang didasarkan pada pernyataan masalah dan definisi konsep yang dilibatkan. Kesederhanaan algoritma *brute force* terkadang membuat algoritma ini terlihat lebih mangkus dari algoritma yang lebih canggih. Biarpun membutuhkan waktu yang cukup banyak dan kurang mangkus, algoritma *brute force* dapat diterapkan pada sebagian besar masalah, terkadang ada masalah yang hanya bisa dipecahkan oleh algoritma *brute force*.

Terdapat beberapa kelebihan algoritma *brute force*, yaitu:

- Algoritma *brute force* dapat digunakan untuk memecahkan hampir sebagian besar masalah
- Algoritma *brute force* adalah algoritma sederhana yang mudah dimengerti
- Algoritma *brute force* merupakan algoritma yang layak untuk memecahkan beberapa masalah penting seperti pencarian, pengurutan, pencocokan *string*, perkalian matriks, dan lain-lain
- Algoritma *brute force* menghasilkan algoritma baku untuk tugas-tugas komputasi seperti penjumlahan ataupun perkalian beberapa bilangan, menentukan elemen minimum atau maksimum dalam suatu tabel.

Selain kelebihan, terdapat juga beberapa kelemehan yang dimiliki oleh algoritma *brute force*. Berikut ini adalah beberapa kelemahan yang dimiliki oleh algoritma *brute force*, yaitu:

- Algoritma *brute force* jarang menghasilkan algoritma yang mangkus
- Beberapa algoritma *brute force* lambat sehingga tidak dapat diterima
- Tidak se"canggih" teknik pemecahan masalah lainnya

Jika sudah bingung dan ragu dengan algoritma yang dapat memecahkan suatu masalah, algoritma *brute force* adalah jalan keluarnya.

Teknik heuristik adalah suatu teknik untuk mempercepat pencarian solusi, yaitu dengan cara mengeliminasi beberapa kemungkinan solusi tanpa harus mengeksplorasinya secara penuh. Teknik heuristik ini dapat juga membantu kita memutuskan solusi mana yang harus dievaluasi pertama kali.

B. *Penerapan Algoritma Brute Force*

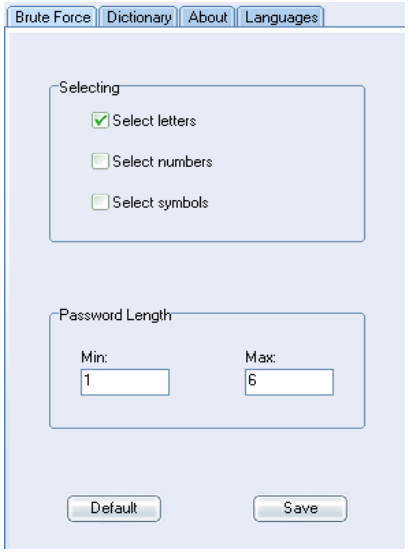
Algoritma *brute force* dapat diterapkan untuk memecahkan berbagai macam persoalan. Berikut ini adalah beberapa contoh masalah yang dapat dipecahkan dengan algoritma *brute force*:

- Menghitung perpangkatan suatu bilangan, a^n dengan a > 0 dan n bilangan bulat > 0
- Menghitung faktorial dari suatu bilangan, n! dengan n bilangan bulat > 0
- Mengalikan dua buah matriks yang berukuran n x n
- Menentukan semua factor bilangan bulat
- Mencari elemen terbesar atau terkecil
- *Sequential search*
- *Bubble sort*
- Menentukan apakah suatu bilangan merupakan bilangan prima atau bukan
- *String matching*
- Mencari pasangan titik yang jaraknya terdekat
- dan lain-lain.

C. *Software RAR Password Recovery Magic*

Software RAR Password Recovery Magic yang dikembangkan oleh Password Recovery Magic Studio Ltd adalah *software* yang dapat membantu penggunanya untuk mencari *password* yang cocok untuk membuka *file* .rar. *Software* ini memiliki beberapa fitur yang dapat menjadi keunggulan:

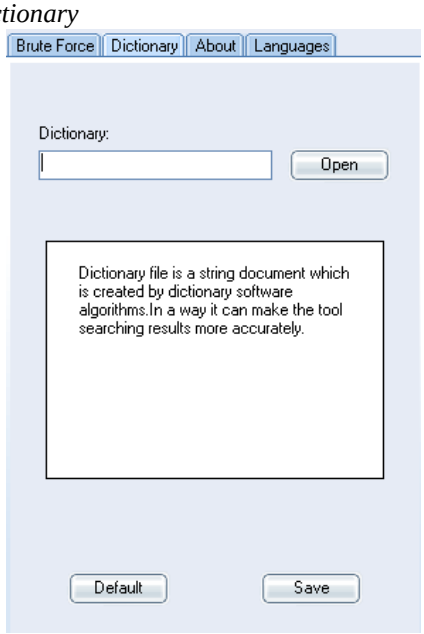
- *Brute Force*



Gambar 4 – Fitur *Brute Force* pada *Software* RAR *Password Recovery Magic*

Fitur *Brute Force* ini adalah bagian dari peraturan yang dapat diubah untuk mencari *password* tersebut menggunakan algoritma *brute force*. *Software* ini dapat melakukan pencarian yang berisi huruf saja, angka saja, simbol saja, atau perpaduan dari huruf, angka, dan simbol. Panjang karakter *password* yang akan dicari juga dapat ditentukan jumlah minimal ataupun jumlah maksimalnya.

- *Dictionary*



Gambar 5 – Fitur *Dictionary* pada *Software* RAR *Password Recovery Magic*

Fitur *Dictionary* ini berfungsi untuk menentukan heuristik (mengeleminasi beberapa kemungkinan solusi) yang terdapat pada *password* tersebut sehingga dapat mempercepat pencarian dan tidak perlu mengevaluasi semua kemungkinan yang ada.

D. File .rar

File yang berekstensi *.rar* adalah hasil pengarsipan satu atau lebih *file* yang menggunakan suatu *software*. *File* .rar ini dapat diberi *password* yang panjang karakternya minimal 1 dan maksimal 127. Karakter pada *password file* .rar ini dapat berupa angka, huruf, maupun simbol.

III. IMPLEMENTASI ALGORITMA *BRUTE FORCE*

Dalam melakukan pencarian password, pertama-tama. Berikut ini adalah *pseudo-code* untuk melakukan pencarian *password* tersebut:

```
function RARPassword (char_min: integer, char_maks: integer, jenis: integer) → array of character

Kamus:
i, j: integer
huruf: array[0..jumlah banyaknya huruf a sampai z termasuk capital - 1] of character
angka: array[0..jumlah angka 0 sampai 9 - 1] of character
simbol: array[0..jumlah simbol yang terdapat di komputer dan valid untuk password RAR - 1] of character
try: array[0..char_maks - 1] of character

Algoritma:
for i ← 0 to jumlah banyaknya huruf a sampai z termasuk capital - 1 do
    huruf[i] ← …{huruf dari a sampai z}
endfor

for i ← 0 to jumlah banyaknya angka 0 sampai 9 - 1 do
    angka[i] ← …{angka 0 sampai 9}
endfor

for i ← 0 to jumlah simbol yang terdapat di komputer dan valid untuk password RAR - 1 do
    angka[i] ← …{simbol yang terdapat di komputer dan valid untuk password RAR}
endfor

if jenis = 1 then {Hanya mencari kombinasi huruf}
    {inisiaslisasi try: array[0..char_min – 1] of character}
    {isi try[0] sampai try[char_min-1] dengan "a", jika tidak match, isi try[char_min-1] dengan "b", jika
```

```
masih tidak match, isi try[char_min-1] dengan "c", dan terus ganti try[char_min-1] sampai dengan "z" jika masih belum match. Jika belum match, ganti try[char_min-2] dengan "b" sampai "z" hingga menemukan yang match. JIka belum match, ganti try[char_min-3] dengan "b" sampai "z" dan seterusnya hingga match.}
    {jika tidak terdapat yang match, inisialisasi try: array[0..char_min] of character dan lakukan pengisian array seperti langkah di atas}
    {jika tidak terdapat yang match, inisialisasi array sampai jumlahnya mencapai char_maks dan coba terus isi array dengan "a" sampai "z" hingga ketemu yang match)
else
    if jenis = 2 then {Hanya mencari kombinasi angka}
    {sama seperti langkah sebelumnya, namun array diisi dengan angka}
    else
        if jenis = 3 then {hanya mencari kombinasi simbol}
            {sama dengan langkah sebelumnya, namun array diisi dengan simbol}
        else
            if jenis = 4 then {mencari kombinasi angka, simbol, dan huruf}
                {sama dengan langkah sebelumnya, namun array disi dengan huruf, angka, dan simbol}
            else
                if jenis = 5 then {mencari kombinasi angka dan simbol}
                    {…}
            endif
        endif
    endif
endif
➔ try
```

Pertama-pertama, pengguna *software* menentukan jenis karakter apa saja yang akan dicari, apakah kombinasi simbol saja, kombinasi angka saja, kombinasi huruf saja, kombinasi antara simbol dan huruf, kombinasi antara simbol dan angka, kombinasi antarahuruf dan angka, atau kombinasi antara simbol, huruf, dan angka.Setelah itu, pengguna *software* akan menentukan panjang *password* yang akan dicari. Semakin besar panjang *password* dan semakin beragam kemungkin yang dicari, maka waktu pencarian akan semakin lama.

Misalkan *password* yang akan dicari adalah "whoa", pengguna *software* hanya melakukan pencarian antara kombinasi angka saja atau pencarian antara kombinasi simbol saja, maka pengguna tidak akan menemukan *password* tersebut. Sehingga penggunaan *software* ini menjadi tidak maksimal jika pengguna tidak "beruntung" untuk menemukan *password* tersebut. Namun jika pengguna menentukan untuk mencari kombinasi di antara ketiganya, maka waktu yang dibutuhkan untuk pencarian akan sangat lama.

Misalkan *password* yang benar adalah "waaaaw" dan pengguna hanya mencari kombinasi huruf. Jumlah karakter yang pengguna tentukan adalah antara 4 sampai 7, maka akan dicari terlebih dahulu perpaduan antara 4 karakter, dimulai dari huruf "a".

a a a a

Namun, "aaaa" adalah bukan *password* yang benar, sehingga *software* akan mencoba kemungkinan berikut ini:

a a a b
a a a c
a a a d
a a a e
a a a f
a a a g
a a a h
a a a i
a a a j
a a a k
a a a l
a a a m
a a a n
a a a o
a a a p
a a a q
a a a r
a a a s
a a a t
a a a u
a a a v
a a a w
a a a x
a a a y
a a a z

Karena tidak ditemukan, huruf ketiga akan diganti dengan huruf "b" dan akan mencoba kemungkinan berikut ini:

a a b a
a a b b
a a b c
a a b d
a a b e
a a b f
a a b g

a a b h
a a b i
a a b j
a a b k
a a b l
a a b m
a a b n
a a b o
a a b p
a a b q
a a b r
a a b s
a a b t
a a b u
a a b v
a a b w
a a b x
a a b y
a a b z

Karena tidak ditemukan, huruf kedua akan diganti dengan huruf "b" dan akan mencoba kemungkinan-kemungkinan berikut ini:

a b a a
a b a b
a b a c
a b a d
a b a e
a b a f
a b a g
a b a h
a b a i
a b a j
a b a k
a b a l
a b a m
a b a n
a b a o
a b a p
a b a q
a b a r
a b a s
a b a t
a b a u
a b a v
a b a w
a b a x
a b a y
a b a z

Karena tidak ditemukan, huruf pertama akan diganti dengan "b" dan akan dicoba kemungkinan berikut ini:

b a a a
b a a b
b a a c
b a a d
b a a e
b a a f
b a a g
b a a h
b a a i
b a a j
b a a k
b a a l
b a a m
b a a n
b a a o
b a a p
b a a q
b a a r
b a a s
b a a t
b a a u
b a a v
b a a w
b a a x
b a a y
b a a z

Karena tidak ditemukan, huruf ketiga akan diganti dengan huruf "b" dan akan mencoba kemungkinan-kemungkinan berikut ini:

b a b a
b a b b
b a b c
b a b d
b a b e
b a b f
b a b g
b a b h
b a b i
b a b j
b a b k
b a b l
b a b m
b a b n
b a b o
b a b p
b a b q
b a b r
b a b s
b a b t
b a b u
b a b v
b a b w
b a b x
b a b y
b a b z

Karena tidak ditemukan, huruf kedua akan diganti dengan "b" dan akan dicoba kemungkinan berikut ini:

b b a a
b b a b
b b a c
b b a d
b b a e

b b a f
b b a g
b b a h
b b a i
b b a j
b b a k
b b a l
b b a m
b b a n
b b a o
b b a p
b b a q
b b a r
b b a s
b b a t
b b a u
b b a v
b b a w
b b a x
b b a y
b b a z

Karena tidak ditemukan, huruf pertama akan diganti dengan huruf c, huruf kedua dan ketiga akan diganti dengan huruf a, dan huruf keempat akan dicoba satu-satu kemungkinannya dari a sampai z. Jika masih belum ditemukan, huruf ketiga akan diganti dengan huruf b dan huruf keempat akan dicoba satu-satu kemungkinannya dari huruf a sampai z. Jika masih belum ditemukan, huruf ketiga akan diganti dengan huruf b dan seterusnya. Jika semua kemungkinan sudah dicoba pada jumlah 4 karakter, dicoba kemungkinan pada jumlah 5 karakter. Begitu seterusnya sampai 6 karakter sehingga *password* "waaaaw" akan ditemukan.

IV. KESIMPULAN

Algoritma *brute force* memang tidak secerdas dan tidak semangkus algoritma canggih yang ada. Namun, algoritma *brute force* sangatlah membantu untuk memecahkan persoalan yang sepertinya tidak mungkin, seperti menebak *password* RAR yang jumlah karakternya antara 1 sampai 127 dengan jenis karakter yang berbagai macam (mulai dari huruf, angka, sampai simbol yang jumlahnya sangat banyak). Karena algoritma *brute force* mencoba satu-satu kemungkinan yang jumlahnya sangat banyak itu, maka waktu yang dibutuhkan sangatlah lama sehingga dibutuhkan kesabaran.

Untuk memecahkan masalah kecil, tidak terlalu rumit, dan tidak menjadikan waktu sebagai *constraint*, maka kita dapat menggunakan algoritma *brute force* sebagai pilihan untuk memecahkan masalah tersebut. Namun jika kita membutuhkan waktu yang singkat dan algoritma yang sangat mangkus, kita tidak dapat menjadikan algoritma *brute force* sebagai alternatif.

V. TERIMA KASIH

Penulis mengucapkan terima kasih kepada Dr. Ir. Rinaldi Munir, M.T. dan Dr. Masayu Leyla Khodra, S.T., M.T., dosen pengajar IF2211 Strategi Algoritma Program Studi Teknik Informatika Institut Teknologi Bandung serta pihak-pihak lainnya yang membantu dan memudahkan penulis untuk menyelesaikan makalah ini.

DAFTAR PUSTAKA

[1] Munir, Rinaldi. 2009. Diktat Kuliah IF2211 Strategi Algoritma. Bandung: Penerbit Institut Teknologi Bandung.
[2] Gambar-gambar merupakan hasil dokumentasi penulis. Jika diambil dari referensi lain, sumbernya akan dicantumkan pada gambar tersebut.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 19 Mei 2014

Arina Listyarini Dwiastuti (13512006)