Penerapan Algoritma *Brute force* dalam Menentukan Kebocoran Pipa

Farid Firdaus (13511091)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]faridfirdaus17@students.itb.ac.id

***Abstrak* – Pada makalah ini akan dijelaskan tentang konsep *brute force* yang diaplikasikan pada bidang konstruksi, yaitu mendeteksi kebocoran pada pipa. Pembandingan akan dilakukan terhadap atribut debit air yang mengalir lewat pipa berdasarkan asas hokum bernoulli**

***Kata Kunci* — Algoritma *brute force*, fluida.**

I. PENDAHULUAN

Algorima *brute force* merupakan salah satu algoritma sederhana yang dapaat digunakan untuk menyelesaikan permasalahan yang dapat ditemukan dalam kehidupan sehari-hari. Algoritma ini adalah algoritma yang naïf, tidak membutuhkan strategi yang rumit, dan hampir dapat diterapkan diberbagai macam persoalan.

Dalam makalah ini, penulis akan menerapkan algoritma *brute force* untuk menyelesaikan masalah yang dapat ditemukan dalam kehidupan sehari-hari yaitu kebocoran pipa. Diharapkan dengan makalah sederhana ini, dapat membantu pembaca dalam memahami pemanfaatan algoritma *brute force* dalam kehidupan sehari-hari.

II. METODE

1 *Brute force*

Dalam pemecahan masalah diatas, penulis menggunakan algoritma *brute force*. Dalam upa-bab ini, penulis akan menjelaskan apakah yang dimaksud dengan algoritma *brute force*

Algoritma *brute force* adalah algoritma yang biasanya digunakan untuk pemecahan masalah dengan skala kecil hingga menengah. Algioritma *brute force* tidak efisien jika digunakan untuk memecahkan masalah dengan domain skala besar.

Algoritma *brute force* sendiri memiliki cara berpikir yang sederhana dalam memecahkan masalah. Inti dari pemecahan masalah adalah mencoba seluruh kemungkinan solusi yang ada. Semakin besar skala permasalahan, maka jumlah kemungkin solusi juga akan bertambah. Pertambahan ini bergantung pada jenis permasalah,, ada yang mengalamai peningkatan secara linear, logaritma, hingga eksponensial. Banyak algoritma-algoritma rumit yang sebenarnya merupakan modifikasi dan pengembangan dari algoritma *brute force*. Algoritma *brute force* sendiri seringkali digunakan sebagai pembanding saat dilakukan uji coba kualitas suatu algoritma. Terdapat juga masalah yang hanya dapat diselekasikan menggunakan algoritma *brute force* seperti permasalahan mencarai elemen maksimum di dalam senarai.

Berikut adalah kekuatan dan kelemahan dari algoritma *brute force*

Kekuatan :

1. Metode *brute force* hampir dapat digunakan untuk menyelesaikan semua permasalahan yang ada.
2. Metode *brute force* sederhana dan mudah dimengerti.
3. Metode *brute force* menghasilkan algoritma yang layak untuk beberapa masalah penting seperti permasalahan pencarian dan pengurutan.
4. Metode *brute force* menghasilkan algoritma baku untuk tugas-tugas komputasi seperti penjumlahan, pengurangan , perkalian dan pembagian, hingga masalah pengurutan dan mencari maksimum dan minimum.

Kelemahan :

1. Metode *brute force* jarang menghasilkan algoritma yang magkus.
2. Beberapa algoritma *brute force* sangat lambat sehingga tidak dapat diterima.
3. Tidak sekontruktif/sekreatif teknik pemecahan masalah lainnya.

Hal unik yang dapat ditemukan dari algoritma ini adalah karena kesederhanaannya, maka algoritma ini sering digunakan untuk aktifitas cracking (membobol) dimaan pembobol akan berusaha memasukkan kata sandi secara acak sesuai pola tertentu atau informasi yang dia dapatkan.

2 Fisika Fluida

Dalam bidang fisika, terdapat cabang ilmu mekanika fluida. Pengertian fluida sendiri adalah zat yang dapat mengalir. Fluida mencakup zat berbentuk gas dan cair karena kedua jenis zat ini dapat mengalir. Air terjun, air keran yang mengalir merupakan nukti zat cair sebagai fluida sedangkan angina yang bertiup, fenomena angina topan, membuktikan bahwa zat gas juga merupakan fluida yang dapat mengalir.

Sifat fluida ini sangat penting bagi kehidupan manusia. Dengan sifat inilah, manusia dapat menghirup oksigen untuk bernafas (udara mengalir masuk kedalam tubuh). Api dapat menyala karena udara disekitarnya mengalir dan menghantarkan oksigen. dan tentunya masih banyak lagi aspek kehidupan yang sangat bergantung dengan sifat fluida ini.

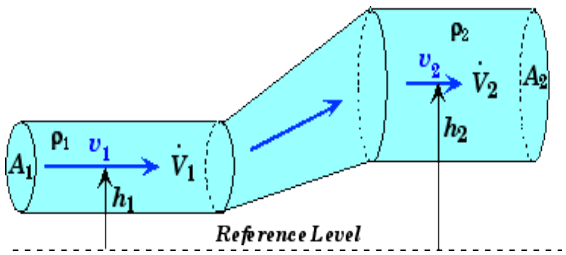
Mekanika fluida sendiri terbagi kedalam 2 jenis, yaitu fluida statis dan fluida dinamis. Fluida statis adalah ilmu yang mempelajari karakteristik dari fluida yang dalam keadaan tidak sedang mengalir, sedangkan fluida dinamis adalah ilmu yang mempelajari karakteristik fluida yang sedang mengalir.

Dalam fluida statis, contoh permasalahan yang biasa ditemui adalah seperti menghitung tekanan fluida didalam suatu wadah yang diam. Salah satu teknologi yang memanfaatkan sifat fluida statis adalah dongkrak hidrolik, dimana dongkrak ini bekerja dengan memanfaatkan hokum bahwa tekanan fluida dan luas permukaan pada dua buah titik berbanding terbalik apabila titik tersebut saling terhubung. Barometer yang digunakan sebagai pengukur tekanan di udara juga memanfaatkan hukum dari fluida statis.

Dalam fluida dinamis sendiri, juga banyak teknologi yang memanfaatkan hukum-hukum dalam fluida dinamis. Salah satunya yang terkenal adalah hukum Bernoulli yang dijabarkan dalam bentuk persamaan dibawah ini

$$P_1 + \frac{1}{2}\rho_1 v_1^2 + \rho_1 g h_1 = P_2 + \frac{1}{2}\rho_2 v_2^2 + \rho_2 g h_2$$

Gambar 2.1 Persamaan Bernoulli



Gambar 2.2 Ilustrasi Hukum Bernoulli
(sumber :
http://cahyarhamadani19.blogspot.com/2013/02/fluida-dinamis-persamaan-kontinutas.html)

P = Tekanan
v = Kecepatan aliran fluida
ρ = Massa Jenis Cairan
g = Gaya Gravitasi
h = Ketinggian
A = luas pipa

Dalam persamaan Bernoulli , disebutkan bahwa jumlah tekanan, energi kinetik per satuan volume, dan energi potensial per satuan volume memiliki nilai yang sama di setiap titik sepanjang aliran fluida ideal. Fluida ideal adalah fluida tak kental, tunak, dan memiliki garis arus yang ideal.

Pemanfaatan hukum Bernoulli yang paling terkenal adalah dalam perancangan sayap pesawat terbang, Sayap pesawat terbang dirancang sedemikian rupa agar saat terbang, gaya yang diberikan dari bagian bawah sayap lebih tinggi dibandingkan gaya yang diberika dari bagian atas sayap pesawat, sehingga pesawat pun dapat terangkat.

Aliran fluida sendiri terbagi menjadi 3 jenis yaitu :

1. Aliran Laminar
   Pada aliran laminar, fluida mengalir dalam bentuk lapisan-lapisan dimana fluida meluncur dengan mulus. Dalam aliran laminar ini, viskositas berfungsi untuk meredam adanya pergerakan aliran fluida secara liar

2. Aliran turbulen
   Aliran turbulen adalah dimana aliran terjadi tidak secara konsisten. Partikel-partikel fluida bergerak sangat tidak menentu akibat percampuran momentum dan energi kinetic dari setiap partikel. Sangat sulit menentukan aliran debit fluida untuk fluida yang mengalir dengan jenis aliran turbulensi.

3. Aliran Transisi
   Aliran transisi merupakan aliran peralihan dari aliran laminar menuju aliran transisi.

Untuk mempermudah penyaluran zat fluida, maka digunakanlah pipa dan pompa, Dengan pompa, maka fluida akan dialirkan menuju pipa tujuan. Saat fluida tersebut mengalir, maka persamaan Bernoulli diatas akan berlaku.

Karakteristik fluida dinamis inilah yang dimanfaatkan dalam jaringan pipa. Pada jalur pipa yang mengalami kebocoran, tekanan fluida di dalam pipa akan mengakibatkan cairan di dalam pipa akan keluar dari pipa. Dengan asumsi bahwa luas pipa disetiap titik adalah sama, maka kebocoran pada pipa dapat dideteksi menggunakan

hukum ini.

Gambar 2.3 Aliran pipa yang mengalami kebocoran (sumber : google.com)

III. PENJELASAN ALGORITMA DAN IMPLEMENTASI

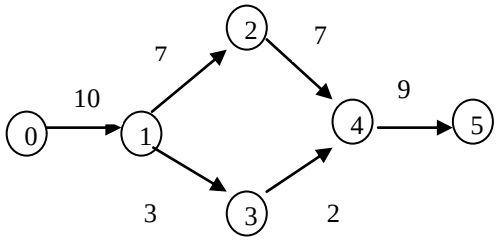
1. Algoritma dan Implementasi

Pada permasalahan kebocoran pipa ini, dibutuhkan sebuah pemodelan sehingga masalah dapat diselesaikan dengan lebih mudah. Penulis memilih pemodelan dalam bentuk graf karena kemiripan masalah dengan graf. Jika dianalogikan, maka titik yang akan diperiksa dapat direpresentasikan sebagai node pada graf, sedangkan aliran air pada pipa dapat dianalogikan sebagai graf berbobot dan memiliki arah (aliran air pada pipa hanya memiliki satu arah).

Dalam implementasinya pun, model permasalahan akan dimasukkan melalui file eksternal, dalam bentuk matriks. Matriks ini menyatakan graf dalam permasalahan tersebut. Berikut adalah contoh matriks yang akan digunakan sebagai bahan uji coba nantinya.

0	10	0	0	0	0
0	0	7	3	0	0
0	0	0	0	7	0
0	0	0	0	2	0
0	0	0	0	0	9
0	0	0	0	0	0

Jika digambarkan dalam bentuk graf berbobot, maka bentuk grafnya adalah sebagai berikut



Terdapat 3 buah metode utama yang digunakan dalam penyelesaian masalah ini. Berikut adalah pseudocode dari metode pertama

```
function getSizeState(ArrayList<Double> state) -
> integer

{Fungsi ini akan mengembalikan sebuah bilangan
integer. Bilangan integer yang dikembalikan
adalah jumlah elemen yang tidak sama dengan
0,0 pada array list state}

KAMUS
integer   : sum

ALGORITMA
{Menghitung jumlah elemen tak 0 pada
ArrayList<Double> state}

sum ← 0

x traversal [0…state.size()] {
    if state.get(x) != 0 then
        sum ← sum + 1
}

→ sum
```

Pada fungsi diatas, akan dikembalikan sebuah nilai integer, yaitu jumlah bilangan tak 0.0 dari sebuah array list of double. Nilai ini digunakan untuk menentukan apa jalur pipa tersebut adalah jalur tunggal, atau sebuah jalur pipa yang akan bercabang /menyatu.

Pada metode kedua, akan dikembalikan sebuah nilai boolean. Fungsi akan mengecek satu persatu yaitu dengan ketentuan menjumlahkan seluruh elemen pada kolom ke-index. Apabila jumlah dari elemen pada kolom tersebut sama dengan nilai dari parameter masukan value, maka fungsi akan mengembalikan nilai true, namun jika tidak, maka fungsi akan mengembalikan nilai false. Berikut adalah pseudocode dari metode kedua.

```
function check (ArrayList<ArrayList<Double>> global, ArrayList<Double> nextState, double value, integer index) -> boolean

{Fungsi ini akan mengembalikan nilai true apabila jumlah bilangan pada kolom ke-index sama dengan nilai value, sebalikanya mengembalikan nilai false apabila tidak sama}

KAMUS
double  : sum
boolean : result

ALGORITMA
{Menghitung jumlah elemen tak 0 pada ArrayList<Double> state}

result ← false
sum ← 0

x traversal [0…global.size()] {
    y traversal [0…global.get(x)..size()] {
        if (y=index) then
            sum ← sum + global.get(x)..get(y)
    }
}

if   (sum = value)
    result ← true

→ sum
```

Pada metode ketiga, adalah metode utama yang sangat penting, metode ini meanfaatkan kedua metode sebelumnya untuk menentukan apakah terdapat kebocoran berdasarkan matriks yang dimasukkan. Di metode ini jugalah, metode *brute force*c diterapkan yaitu mencoba satu-persatu seluruh elemen matriks tak 0.

Metode ketiga bisa dikatakan adalah metode utama dimana metode *brute force* diterapkan pada metode ini. *Brute force* dilakukan saat akan dilakukan pengecekan pada elemen di dalam matriks. Seluruh elemen tak 0 akan diperiksa dengan ketentuan seperti yang telah dijelaskan pada metode check.

Ketentuan tambahan yaitu nilai tersebut hanya diperiksa jika memiliki bilangan tak 0 sebanyak 1 buah untuk per baris dalam matriks. Misalkan untuk baris dengan nilai (0 0 1 0 2 0), maka baris tersebut tidak akan diperiksa menggunakan metode check, namun jika baris tersebut berisi elemen (0 0 1 0 0 0), maka baris tersebut akan diperiksa, karena merupakan arus pada titik ini hanya ada 1 buah, sehingga pengukuran debit dapat dilakukan dengan lebih mudah.

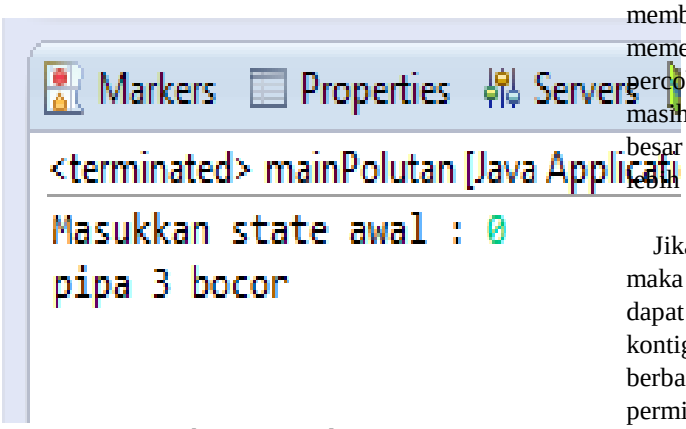
```
procedure hasLeak
(ArrayList<ArrayList<Double>> allState)

{Procedure ini akan mencari tahu stupersatu seluruh elemen, apakah nilai asal elemen tersebut, memiliki nilai yang sama dengan nilai elemen yang sedang diperiksa, apabila ditemukan nilai yang tidak sama, maka diindikasikan bahwa terjadi kebocoran pada titik tersebut. Prosedur akan mencetak pada titik mana saja terjadi kebocoran}

KAMUS
double  : value
integer: index

ALGORITMA
{Menghitung apakah terdapat kebocoran dengan mengecek satu persatu seluruh elemen tak 0 pada matriks allState}

value ← 0
index ← 0
x traversal [0…allState.size()] {
   if (stateAwal = x) then
      x ← x + 1 {diabaikan jika yang akan
   diperiksa adalah state awal}

   if (getSizeState(allState.get(x)) = 1) then
      n traversal [0…allState.get(x).size()] {
         if (allState.get(x).get(n) != 0) then {
            value ←allState.get(x).get(n)
            index ← n
            break
         }
      }

      if (!(check(allState, allState.get(index),
value, x))) then
      write("pipa " + x + " bocor")
}
```

Jika ditemukan perbedaan antara debit air masuk dengan debit air keluar pada satu titik (nilai debit air masuk terletak pada jumlah nilai elemen matriks pada kolom yang sama, sedangkan nilai debit air keluar adalah elemen matriks yang sedang diperiksa), maka diindikasikan pada titik tersebut terjadi kebocoran pipa. Asumsi inilah yang digunakan untuk menentukan apakah dititik tersebut mengalami kebocoran pipa atau tidak.

Nilai value pertama-tama diinisialisasi dari nol, lalu akan di cek satu-persatu pada kolom yang telah ditentukan. Apabila terdapat nilai bukan 0, maka nilai tersebut akan ditambahkan kedalam nilai value. Pada akhirnya, nilai value ini,merepresentasikan nilai debit air yang masuk pada titik yang bersangkutan. Nilai value ini akan

dibandingkan dengan nilai awal yang dijadikan acuan pemeriksaan (nilai ini merepresentasikan debit air yang keluar dari titik tersebut).

Dari hasil pengujian pada matriks yang telah ditunjukkan sebelumnya, didapatkan hasil pada program adalah sebagai berikut.



Gambar 3.1. Hasil Pengujian

Pada pengujian diatas, pengguna akan diminta untuk memasukkan state awal aliran fluida dimulai. Input ini berguna untuk mengecek nantinya, apakah titik tersebut akan diperiksa atau tidak. Jika titik yang akan diperiksa adalah state awal, maka pemeriksaan akan dihentikan , dan titik yang akan diperiksa, akan ditambahkan (state awal akan dilewati begitu saja). Hal ini dilakukan dengan asumsi bahwa fluida mengalir dari state awal, tanpa adanya masukkan debit air menuju state awal tersebut (state awal dinggap sebagai sumber aliran air).

Dari hasil pengujian program diatas, menunjukkan bahwa terjadi kebocoran pada pipa 3. Jika diperhatikan pada pemodelan graf sebelumnya, terlihat dengan jelas, pada titik 3, jumlah debit air yang masuk adalah sebanyak 3 unit, namun jumlah debit air yang keluar hanya sebesar 2 unit. Terjadi perbedaan sebesar 1 unit yang dimana ini melanggar hokum asas Bernoulli. Dengan demikian, diindikasikan bahwa pada pipa disekitar titik 3 mengalami kebocoran.

IV. PARAMETER LANJUTAN

Dalam pengembangan program ini, dapat dilakukan riset lebih lanjut. Terdapat banyak parameter yang menentukan ketepatan hasil penghitungan, mengingat hokum Bernoulli menggunakan 6 buah parameter. Pada percobaan yang penulis lakukan, diasumsikan bahwa pipa yang digunakan berada pada ketinggian yang sama, luas permukaan yang sama, dan benar-benar memiliki karakteristik fluida ideal.

Jika diambil contoh kasus, dalam pengujian yang dilakukan adalah menggunakan fluida yang tidak ideal, maka fluida pada bagian yang bersentuhan dengan pipa akan bergerak lebih lambat (diakibatkan gaya gesek cairan dengan pipa), dibandingkan fluida yang berada di tengah-tengah pipa. Karena perbedaan kecepatan fluida ini, maka aliran bisa menjadi tidak ideal, dan debit air pun akan berubah-ubah. Dengan parameter ukur yang berubah-ubah, maka data hasil yang didaptkan pun, keakuratannya sulit untuk dipertanggungjawabkan.

Meskipun demikian, dengan pemecahan masalah ini, membuktikan bahwa *brute force* dapat digunakan untuk memecahkan solusi di kehidupan sehari-hari. Untuk percobaan kali ini, skala permasalahan yang dipergunakan masihlah kecil. Untuk skala permasalahan yang lebih besar lagi, mungkin akan membutuhkan algoritma yang lebih cerdas dibandingkan algoritma *brute force*.

Jika diaplikasikan langsung kedalam kehidupan nyata, maka dibutuhkan perangkat keras berupa sensor yang dapat mengukur debit air didalam suatu pipa secara kontigu. Pemanfaatan aplikasi ini dapat diterapkan pada berbagai bidang, baik di bidang konstruksi sipil, perminyakan, lingkungan, kelautan, pertambangan, dan masih banyak lagi. Dalam bidang industry, banyaknya digunakan pipa-pipa untuk mengalirkan berbagai macam fluida, mulai dari bahan bakar, gas, bahan-bahan kimia, hingga zat-zat yang mudah terbakar, menjadikan pentingnya meningkatkan keamanan dalam hali ini. Dengan semakin berkembangnya aplikasi yang serupa, dapat meningkatkan tingkat pendeteksian dini apabila terjadi kebocoran pada pipa-pipa tersebut ehingga kerugian yang lebih besar dapat terhindarkan

V. KESIMPULAN

Algoritma *brute force* dapat diterapkan sebagai solusi pemecahan masalah pencarian titik pipa bocor pada sistem jaringan pipa. Algoritma ini dapat diterapkan, dengan mengikuti ketentuan bahwa, fluida yang digunakan adalah fluida ideal, dan pipa fluida memiliki ketinggian dan luas permukaan,yang sama untuk setiap titik.

Sistem pendeteksian dini ini juga dapat diterapkan dalam berbagai aspek, mulai dari lingkup rumah tangga hingga lingkup iindustri berskala besar seperti pabrik baja, dan sejenisnya.

Diperlukan riset lebih lanjut akan parameter-parameter lanjutan yang seperti telah dijelaskan diatas sehingga pengukuran debit air menjadi lebih akurat, sehingga kebocoran dengan skala kecol pun dapat dideteksi sedini mungkin

REFERENCES

[1] http://fisikadedek.blogspot.com/2013/05/fluida-statik-dan-dinamis.html
[2] http://cahyarhamadani19.blogspot.com/2013/02/fluida-dinamis-persamaan-kontinutas.html
[3] http://www-igm.univ-mlv.fr/~lecroq/string/node3.html
[4] http://www.cse.ohio-state.edu/~gurari/course/cis680/cis680Ch16.html
[5] Kanginan, Marthen (2006). *Fisika 2 untuk SMA Kelas XI*. Jakarta: Erlangga.

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, 20 Desember 2013

Farid Firdaus
13511091