PENCOCOKAN EKSPRESI REGULER DENGAN ALGORITMA RUNUT BALIK

Ernestasia Siahaan

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung
Jl. Ganesha 10 Bandung
e-mail: if16026@students.if.itb.ac.id

ABSTRAK

Dalam penggunaan aplikasi komputer, sering dilakukan pencocokan string untuk mencari bagian tertentu dari sebuah teks atau dokumen. Terkadang pemeriksaan hanya diperlukan untuk mencocokan string masukan dengan sebuah pola tertentu (tidak diperlukan ketepatan kesesuaian semua karakter, hanya beberapa karakter saja). Untuk itu, disusun beberapa aturan untuk mendefinisikan kelompok string tertentu yang disebut dengan ekspresi reguler.

Pencocokan sebuah ekspresi reguler dengan string masukan dilakukan dengan mengimplementasikan algoritma tertentu. Pada pencocokan ekspresi reguler tertentu, digunakan algoritma runut balik. Meskipun kompleksitas waktu algoritma ini besar dibandingkan dengan algoritma lainnya, algoritma runut balik dapat menangani ekspresi reguler yang cukup rumit.

Algoritma runut balik merupakan perbaikan dari algoritma *bruteforce* yang menggunakan pohon DFS dalam pencarian solusinya. Pada algoritma runut balik, apabila sebuah lintasan tidak menuju solusi persoalan, lintasan tersebut tidak dilanjutkan dan dilakukan langkah mundur kepada simpul pohon yang masih dapat diperluas untuk membuat lintasan pencarian solusi yang baru.

Algoritma runut balik terutama digunakan dalam mencocokkan ekspresi reguler yang memiliki simbol-simbol khusus seperti '*', '?', '+', dan sebagainya.

Kata kunci: Ekspresi Reguler, Runut Balik.

1. PENDAHULUAN

Dalam penggunaan aplikasi komputer, sering dilakukan pencocokan string untuk mencari bagian tertentu dari sebuah teks atau dokumen. Karena teks yang akan diperiksa bersifat tetap, dapat dilakukan pencarian karakter per karakter yang membentuk string yang sama dengan masukan yang diberikan.

Terkadang pemeriksaan karakter per karakter tidak diperlukan, namun diperlukan pencocokan string masukan dengan sebuah pola yang terdefinisi. Misalnya, pada sebuah program tekstual, diperlukan pencocokan untuk memastikan string yang dimasukkan oleh pengguna merupakan sebuah command atau bukan. Jika command yang ada sedikit, dapat dilakukan enumerasi setiap command yang ada untuk dicocokkan satu per satu dengan masukan pengguna. Namun jika terdapat banyak command, enumerasi tidaklah menjadi pilihan yang mangkus.

Untuk itu, didefinisikan sejumlah aturan untuk membedakan kumpulan string tertentu. Aturan tersebut dinamakan ekspresi reguler. Dengan ekspresi reguler, string tertentu dapat diidentifikasi dengan simbol tertentu. Misalnya, untuk mengetahui apakah sebuah string merupakan email atau bukan, dibuat ekspresi reguler sebagai berikut:

\b[A-Z0-9._%-]+@[A-Z0-9.-]+\.[A-Z]{2,4}\b

Berdasarkan aturan tersebut, ketika dalam sebuah string ditemukan simbol '@' dan '.' yang diikuti dua sampai empat alfabet, dapat dipastikan string tersebut merupakan sebuah alamat email.

Salah satu algoritma yang digunakan untuk mencocokkan ekspresi reguler dengan sebuah string masukan adalah algoritma runut balik.

Pada makalah ini akan dijelaskan tentang ekspresi reguler dan penggunaan algoritma runut balik untuk mencocokkan ekspresi reguler tertentu dan string masukan.

2. EKSPRESI REGULER

2.1 Pengertian

Ekspresi reguler merupakan susunan karakter yang mendefinisikan pola tertentu sebuah string. Jika pada pencarian string biasa harus dilakukan pencocokan

karakter satu per satu, maka pada penggunaan ekspresi reguler hanya dilakukan pencocokan pola string.

Untuk memeriksa apakah sebuah ekspresi reguler cocok dengan sebuah string masukan, terdapat dua jenis algoritma yang berbeda. Algoritma yang pertama adalah pembentukan sebuah DFA (Deterministic Finite Automata). DFA yang terbentuk dari ekspresi reguler dijalankan pada string masukan. Proses ini memerlukan waktu yang berbanding lurus dengan panjang string masukan.

Algoritma yang kedua adalah algoritma runut balik. Penggunaan algoritma ini untuk memeriksa sebuah ekspresi reguler akan dijelaskan pada bab berikutnya. Akan terlihat bahwa algoritma ini mempunyai kompleksitas waktu yang lebih besar daripada algoritma pertama. Namun, algoritma ini dapat menangani kasus-kasus yang lebih kompleks daripada yang dapat ditangani oleh algoritma pertama.

Beberapa contoh penggunaan ekspresi reguler adalah sebagai berikut:

- Validasi masukan string sesuai pola tertentu, misalnya memeriksa apakah string yang dimasukkan berupa sebuah alamat *email*, nomor telepon, dan sebagainya.
- Menemukan substring dari sebuah string berdasarkan kecocokan pola yang telah didefinisikan.
- dan lain-lain.

Penggunaan ekspresi reguler ini ditemui pada *text editor*, kakas dan bahasa pemrograman seperti Perl, Tcl dan PHP.

2.2 Sintaks Ekspresi Reguler

Berikut ini adalah beberapa aturan pembentukan sebuah ekspresi reguler:

- Z* menyatakan Z dapat muncul sejumlah nol atau lebih kali.
- Z+ menyatakan Z dapat muncul sejumlah satu atau lebih kali.
- Z? menyatakan Z dapat muncul sejumlah nol atau satu kali saja.
- Z{3} menyatakan Z dapat muncul tepat tiga kali.
- Z{3,} menyatakan Z dapat muncul paling tidak tiga kali.
- Z{3,6} menyatakan Z dapat muncul antara tiga sampai enam kali.
- XY | YZ menyatakan bahwa string dapat berupa salah satu dari XY atau YZ.
- ^XY menyatakan sebuah baris dimulai dengan XY.
- XY$ menyatakan sebuah baris berakhir dengan XY.
- [Xx]Y menyatakan string dapat berupa XY maupun xY.
- [0-9] menyatakan sebuah karakter antara 0 sampai dengan 9.
- [A-Z] menyatakan sebuah karakter antara A sampai Z (hanya menerima huruf kapital).
- [a-z] menyatakan sebuah karakter antara a sampai z (hanya menerima huruf kecil).

3. ALGORITMA RUNUT BALIK

3.1 Pengertian

Algoritma runut balik merupakan metode pencarian solusi persoalan dengan membangun pohon dinamis secara DFS (Depth First Search). Adapun langkah-langkah pencarian solusi dengan algoritma runut balik adalah sebagai berikut:

- Dibentuk sebuah lintasan untuk mencari solusi dari akar pohon ke daun, sesuai metode pencarian DFS. Setiap simpul yang sudah dibangkitkan dinomori dari atas ke bawah sesuai dengan urutan kebangkitannya.
- Apabila lintasan yang dibentuk oleh sebuah simpul yang sedang diperluas tidak menuju ke arah solusi, simpul tersebut dimatikan.
- Setelah sebuah simpul dimatikan, pencarian dilanjutkan dengan membangkitkan simpul anak lainnya. Apabila tidak ada lagi simpul anak yang dapat dibangkitkan, pencarian dilanjutkan dengan melakukan runut balik ke simpul terdekat yang masih hidup.

Demikian dilakukan hingga ditemukan solusi atau jika tidak terdapat lagi simpul hidup untuk melakukan runut balik.

3.2 Pseudo Code

Algoritma runut balik dapat diimplementasikan secara rekursif maupun secara iteratif. Pada makalah ini, akan diberikan skema rekursif algoritma runut balik.

```
procedure RunutBalik (input k : integer)
{Mencari semua solusi persoalan dengan metode runut balik; skema rekursif
Masukan: k, yaitu indeks komponen vector solusi, x[k]
Keluaran: solusi x = x[1], x[2], …, x[n]
}

Algoritma:
   for tiap x[k] yang belum dicoba sedemikian sehingga (x[k] ← T(k)) and B(x[1], x[2], …, x[k]) = true do
       if (x[1], x[2], …, x[k]) adalah lintasan dari akar ke daun then
          SimpanSolusi(x)
       endif
       RunutBalik(x+1)
   endfor
```

3.3 Kompleksitas Waktu

Kompleksitas waktu algoritma runut balik bersifat eksponensial. Karena setiap simpul yang dibangkitkan pada pohon ruang statusnya berkaitan dengan sebuah pemanggilan rekursif, jika jumlah simpul dalam pohon ruang status sebanyak 2^n, maka algoritma runut balik dapat membutuhkan waktu hingga $O(p(n)2^n)$ dengan p(n) menyatakan polinom derajat n yang merupakan waktu perhitungan pada setiap simpul.

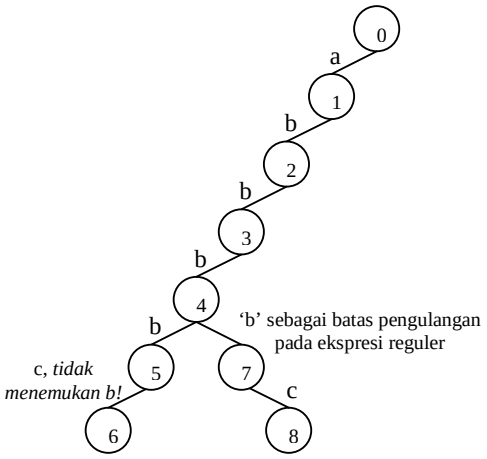
4. IMPLEMENTASI ALGORITMA RUNUT BALIK PADA PENCOCOKAN EKSPRESI REGULER

Telah disebutkan sebelumnya bahwa kelebihan algoritma runut balik dibandingkan dengan algoritma lainnya untuk mencocokkan ekspresi reguler dengan string masukan adalah kemampuan algoritma runut balik menangani pemeriksaan eskpresi reguler yang kompleks.

Ekspresi reguler yang memiliki aturan-aturan seperti *, ?, |, [ ] atau + ditangani dengan algoritma runut balik untuk memeriksa karakter-karakter sebelum dan sesudah simbol-simbol tersebut. Misalnya, untuk simbol '*', tidak terdapat dijelaskan berapa jumlah karakter sebelum simbol tersebut yang akan muncul. Untuk simbol '|', diperlukan algoritma untuk memastikan karakter-karakter yang mengikutinya tetap diperiksa ketika string masukan pengguna tidak cocok dengan karakter-karakter yang berada di sebelah kiri simbol.

Untuk menjelaskan implementasi algoritma runut balik pada pencocokan ekspresi reguler dengan string masukan, akan digunakan beberapa contoh.

*Diberikan sebuah ekspresi reguler /^ab*bc/. Berikut adalah skema runut balik yang terjadi ketika diterima masukan string abbbbc.*



Gambar 1. Pembentukan pohon pada algoritma runut balik untuk ekspresi reguler /^ab*bc/ dengan masukan abbbbc

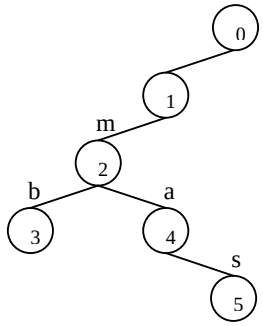
Pada ekspresi reguler yang diberikan, terdapat simbol asterik '*' yang menandakan pengulangan karakter yang berada di sebelah kirinya, yaitu karakter 'b'. Seperti yang sudah disebutkan, tidak terdapat batasan pengulangan karakter yang mendahului simbol '*'. Berakhirnya pengulangan ditandai dengan ditemukannya karakter yang dideklarasikan pada sebelah kanan '*'.

Pada kasus ekspresi reguler /^ab*bc/, pengulangan berhenti ketika ditemukan karakter 'b' yang diikuti karakter 'c'.

Ketika membaca karakter 'a', algoritma runut balik akan mulai memeriksa masukan berupa karakter 'b'. Karena terdapat simbol '*' pada ekspresi reguler, maka diharapkan terjadi pengulangan 'b' untuk jumlah yang tidak ditentukan.

Pada simpul keenam pohon yang dibentuk, ternyata tidak ditemukan karakter 'b', melainkan karakter 'c'. Karenanya, algoritma runut balik akan segera melangkah mundur sampai simpul keempat, karena simpul kelima tidak lagi akan dibaca sebagai bagian dari pengulangan karakter 'b', melainkan sebagai pembatas pengulangan yang dilanjutkan dengan karakter 'c'.

Pada ekspresi reguler [Ember|Emas]. Ketika diberi masukan berupa string 'emas', berikut adalah algoritma runut balik yang terjadi.



Gambar 2. Pembentukan pohon pada algoritma runut balik untuk ekspresi reguler [Ember|Emas] dengan masukan 'emas'

Pada ekspresi reguler yang diberikan, terdapat simbol '|' yang menyatakan bahwa input yang diterima dapat berupa string di sebelah kiri atau di sebelah kanan simbol.

Algoritma runut balik akan memeriksa terlebih dahulu apabila input masukan sama dengan string di sebelah kiri simbol '|'. Apabila tidak sama, maka dilakukan langkah mundur untuk selanjutnya mencocokkan string masukan dengan string di sebelah kanan simbol '|'. Hal ini mungkin dilakukan karena dua karakter pertama string masukan dan string pada ekspresi reguler (baik di sebelah kiri maupun kanan simbol '|') sama.

5. KESIMPULAN

Meskipun mempunyai kompleksitas waktu yang lebih besar dibandingkan algoritma lainnya, algoritma runut balik lebih fleksibel untuk melakukan pencocokan ekspresi reguler tertentu dengan string masukan. Ekspresi reguler yang ditangani oleh algoritma runut balik terutama ekspresi yang mengharuskan dilakukan pemeriksaan karakter sebelum dan sesudah simbol tertentu (khususnya simbol perulangan), seperti '*', '+', '?', dan sebagainya.

REFERENSI

[1] Munir, Rinaldi. Diktat Kuliah IF2251 Strategi Algoritmik. Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung. 2006.

[2] http://www.perlmonks.org/?node_id=390117, diakses pada tanggal 19 Mei 2008.

[3] http://en.wikipedia.org/wiki/Regular_expressions, diakses pada tanggal 19 Mei 2008.