IMPELEMENTASI STRUKTUR DATA PATRICIA TREE PADA AUTOCOMPLETE SEACRH BOX

Zusni Adisya[1], Ade Romadhony[2], Adiwijaya[3]

[1]Teknik Informatika, Fakultas Teknik Informatika, Universitas Telkom

---

Abstrak

Autocomplete pada search box berhubungan dengan data yang begitu besar. Sehingga ketika dilakukan pencarian frase/kata pada database terdapat kendala, dimana ketika semua frase harus ditelusuri untuk mendapatkan hasil dan terdapat hubungan antara server dan client, maka hal seperti ini akan membebani kinerja server. Sehubungan dengan itu, diperlukan metode khusus dalam hal pengambilan data, agar prosesnya ringan dan cepat. Salah satu yang dapat dilakukan yaitu dengan menggunakan suatu struktur data patricia tree.

Penggunaan patricia tree didasarkan karena pencarian dilakukan pada frase awal dari keseluruhan frase yang diinginkan. Sehingga ketika dilakukan pencarian pada patricia tree, tidak perlu menelusuri semua struktur patricia tree, cukup pada struktur patricia tree yang karakter awalnya sesuai saja. Pada patricia tree ini node yang dibangun bisa diberi bobot, sehingga pada kasus ini pemunculan suggestion dapat diprioritaskan berdasarkan bobotnya.

Setelah dilakukan pengujian, penelitian ini memberikan hasil bahwa patricia tree mampu memberikan respon hasil pencarian yang lebih cepat dibandingkan prefiks tree sebagai struktur data pembanding. Pembentukan tree dengan pemberian bobot juga memberikan hasil yang lebih baik dalam ketepatan pencarian.

Kata Kunci : patricia tree, trie, autocomplete, search box.

Abstract

Autocomplete in the search box associated with a large of data. So when do the search phrase / word in the database there are constraints, which when all the phrases must be traced to obtain the results and there is a connection between the server and client, this would overload the server's performance. Accordingly, a methods are needed specifically in terms of data retrieval, so that the process is lightweight and fast. One that can be done by using a patricia tree data structure.

Patricia tree is based on the use of a search performed on the initial phrase of the whole phrase desired. So when do a search on the patricia tree, no need to browse through all the patricia tree structure, simply on the structure of the character originally patricia tree corresponding course. Patricia tree at this node is built can be weighted, so that in this case the appearance of suggestion can be prioritized based on its weight.

After testing, this study provides results that patricia tree capable of providing search results more quickly than a prefix tree data structure as a comparison. The establishment of tree by assigning weights also gives better results in search accuracy.

Keywords : patricia tree, trie, autocomplete, search box.

---

1. Pendahuluan

1.1 Latar Belakang Masalah

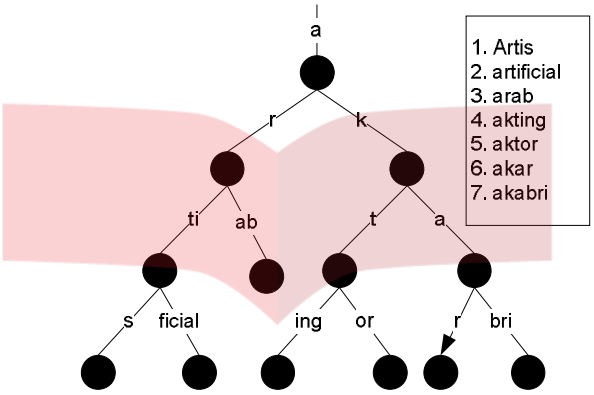
Search box mempunyai peranan penting dalam pencarian informasi digital. *Search box* digunakan untuk menerima *key word* dalam suatu pencarian informasi[1]. *Key word* yang di berikan oleh *searcher* akan diproses oleh mesin pencarian untuk mendapatkan informasi yang diinginkan. Informasi yang didapat bergantung kepada *key word* yang diberikan oleh *searcher* tersebut. Bisa jadi *searcher* tidak mendapatkan informasi apapun ketika *searcher* salah mengetikkan *key word* tersebut. Oleh karena itu, terkadang *search box* dilengkapi dengan fitur *autocomplete,* untuk memberikan perkiraan kata/frase yang akan digunakan *searcher* sebagai *key word* pencarian.

Sebenarnya, *autocomplete* tidak hanya dapat digunakan pada *search box*, tetapi dapat juga digunakan pada *web browser*, *e-mail program*, *source code editor*, dan lain-lain [1]. Namun, *autocomplete* pada mesin pencarian berhubungan dengan data yang begitu besar. Sehingga ketika dilakukan pencarian frase/kata pada database terdapat kendala, dimana ketika semua frase harus ditelusuri untuk mendapatkan hasil dan terdapat hubungan antara server dan client, maka hal seperti ini akan membebani kinerja server[2,3]. Sehubungan dengan itu, diperlukan metode khusus dalam hal pengambilan data, agar prosesnya ringan dan cepat. Salah satu yang dapat dilakukan yaitu dengan menggunakan suatu struktur data *trie*.

Trie atau *prefix tree* merupakan suatu struktur pohon dimana tiap *path* dari akar sampai daun sesuai dengan 1 kunci yang telah direpresentasikan sebelumnya[3,4,5]. Penggunaan *trie* didasarkan karena pencarian dilakukan pada frase awal dari keseluruhan frase yang diinginkan. Sehingga ketika dilakukan pencarian pada *trie*, tidak perlu menelusuri semua struktur *trie,* cukup pada struktur *trie* yang karakter awalnya sesuai. Dengan stuktur data *trie*, ketika *searcher* mengetikan *key word* berupa huruf atau kata, pencarian tidak dilakukan pada database melainkan pada struktur data tersebut, lalu hasil pencarian akan dikembalikan kepada *searcher* sebagai *suggestion* dari *key word* tersebut.

Dengan *trie*, pencarian menjadi lebih mudah dan efisien. Namun ada 1 kelemahan dimana struktur pohon yang dibangun akan menjadi struktur yang besar, karena dalam *trie* 1 node diisi oleh 1 alphabet [5]. Maka, perlu dilakukan pemangkasan yaitu dengan melakukan *concatenation* pada root yang hanya memiliki 1 daun. *Trie* seperti ini disebut *patricia tree* atau *radix*

tree. *Patricia tree* mempunyai persyaratan dimana minimal *child* yang dimiliki oleh tiap node adalah 2, jadi ketika suatu node hanya memiliki 1 child maka akan dilakukan *concatenation*[3]. Dengan demikian, struktur pohon yang dihasilkan menjadi lebih simple [3,6]. Seperti dicontohkan pada Gambar 1.1.



Gambar 1-1 Patricia Tree

Kelebihan lain dari *patricia tree* adalah patricia tree cenderung lebih mampu menampung string yang panjang dan berjumlah tak terbatas[3]. Pada patricia tree ini node yang dibangun bisa diberi bobot, sehingga pada kasus ini pemunculan *suggestion* dapat diprioritaskan berdasarkan bobotnya. Diharapkan struktur *patricia tree* ini dapat diimplemetasikan dalam pencarian data pada *autocomplete search box,* sehingga dapat dibangun suatu stuktur pohon yang dapat menghasilkan pencarian yang lebih cepat dan efisien.

1.2 Perumusan Masalah

Permasalahan yang diangkat pada Tugas Akhir ini antara lain:

1. Bagaimana mengimplementasikan *patricia tree* sebagai suatu struktur data *trie*, yang digunakan dalam *autocomplete search box*.
2. Bagaimana membuat *autocomplete search box* yang dinamis, yang dapat diimplentasikan di berbagai halaman web.

Penelitian ini dibatasi oleh beberapa hal, antara lain:

1. Data sampel awal yang akan digunakan adalah data *digital library* Institut Teknologi Telkom
2. Data yang digunakan berdasarkan data yang diberikan oleh user.
3. Preprosesing data yang digunakan adalah proses penghilangan *stop words* dan *tokenization.*

1.3 Tujuan

Tujuan yang ingin dicapai dalam penelitian ini adalah

1. Menganalisis dan mengimplementasikan *patricia tree* dalam penyimpanan dan pencarian data berupa string
2. Membangun *autocomplete search box* dengan stuktur data *patricia tree*
3. Menganalisis kecepatan pencarian dengan *patricia tree search* dibandingkan dengan *prefix tree search*
4. Menganalisis ketepatan hasil pencarian pada *patricia tree search* dibandingkan pada *prefix tree search*

1.4 Metodologi Penyelesaian Masalah

1. Studi Literatur

   Pada tahap ini dilakukan pemahaman literatur tentang stuktur data, terutama tentang *prefix tree* dan *patricia tree* serta domain permasalahan (*autocomplete search box*)

2. Pengumpulan Data

   Pada tahap ini dilakukan pengumpulan data sebagai sample database, sekaligus pemilihan data yang akan digunakan.

3. Implementasi Struktur Data

   Struktur data yang digunakan adalah *patricia tree*. Pada tahap ini, akan dilakukan pembangunan struktur data berdasarkan sample database yang ada. Pada tahap ini juga dibangun suatu struktur data *prefix tree* sebagai pembanding.

4. *Autocomplete Search Box Development*

   Pada tahap ini dibangun suatu *search box* dengan tambahan fitur *autocomplete*. Dimana proses pencarian data menggunakan struktur data yang telah dibangun.

5. Pengujian dan Analisa Hasil

   Pada tahap ini dilakukan pengujian terhadap kecepatan dan ketepatan pencarian *autocomplete search box* dengan implementasi stuktur data *patricia tree* dan *prefix tree*.

5. Kesimpulan dan Saran

5.1 Kesimpulan

Dari hasil pengujian dan analisis, dan hal-hal yang diperoleh selama pengerjaan Tugas Akhir ini, diperoleh beberapa kesimpulan:

1. Sistem berhasil dibangun dengan diawali proses preprocessing data kemudian mampu melakukan pembentukan tree, dan dapat diimplementasikan pada autocomplete search box. Adapun untuk performansi, patricia tree bergantung kepada kekuatan memori client, karena proses insert dan search patricia tree dilakukan di sana.
2. Kelebihan patricia tree dibandingkan prefiks tree adalah proses search yang cenderung lebih cepat dan stabil, karena node yang harus ditelusuri lebih sedikit daripada prefiks tree. Kelebihan lain adalah pertumbuhan kebutuhan ruang patricia lebih 4x lebih lambat dibandingkan prefiks tree.
3. Kekurangan patricia tree adalah proses insert yang cenderung 3x lebih lama dibandingkan dengan prefiks tree karna banyaknya proses yang harus dieksekusi.
4. Pembobotan data berpengaruh memberikan hasil yang lebih baik terhadap ketepatan hasil pencarian baik dilihat dari sisi user maupun sisi system, namun ada kelemahan dimana diperlukannya tambahan waktu untuk melakukan pengurutan ranking.
5. Patricia tree dan prefiks tree sama-sama memiliki kompleksitas waktu yang sama untuk kasus terburuk, yaitu O(nk) untuk insert sejumlah n data atau search sejumlah n data dengan maksimal k karakter untuk setiap data, namun prefiks tree tumbuh lebih cepat dalam mendekati O(nk) dibandingkan patricia tree.

5.2 Saran

Hasil analisis dan evaluasi terhadap tugas akhir ini menunjukan bahwa system yang dibangun masih dapat dikembangkan lagi. Beberapa saran untuk pengembangan lebih lanjut yang dapat dilakukan adalah:

1. Menggunakan metode khusus untuk menghindari loading data yang memakan waktu lama ketika pembentukan tree diawal proses.
2. Menggunakan metode lain untuk membuat fitur autocomplete pada suatu search box

3. Menerapkan autocomplete pada kasus lain, seperti browser, code editor, email program, dan lain lain.
4. Melakukan perbandingan struktur data patricia tree dengan metode lain, selain prefiks tree.

DAFTAR PUSTAKA

[1] Freedman, Alan. 2010. Computer Desktop Encyclopedia. USA: Computer Language Company Inc. [Online] Available at: http://computer.yourdictionary.com/

[2] Daskam, Steve. 2009. Autocomplete Data Structure. Dalas [Online] Available at : http://stevedaskam.wordpress.com/2009/05/17/autocomplete-data-structures/

[3] Zhao, Xiaoyan. 2000. Trie Methods for structure data on secondary storage. McGill University.

[4] Aoe, Jun-Ichi dkk. 1992. An Efficient Implementation of Trie Structures. Japan: University Of Tokushima.

[5] Hariyanto, Bambang. 2003. Struktur Data: Memuat Objek Pengembangan Orientasi Objek. Bandung: Informatika.

[6] Mcllroy, Peter M dkk. Engineering Radix Sort. Berkeley: Computer Science Research Group University of California.

[7] Rehman, Tahseen Ur . 2008. Prefix Searching with Radix Tree. [Online] Available at: http://bootstrapping.wordpress.com/2008/01/24/prefix-searching-with-radix-tree/

[8] Baeza-Yates, Ricardo A. 1996. Fast Text Searching for Regular Expressions or Automaton Searching on Tries. Chile: University of Chile.

[9] Devroye, Luc. 2004. Universal Asymptotic for Random Tries and Patricia Trees. Canada: School of Computer Science McFill University.

[10] D. Zandolin, A. Pietracaprina. Mining Frequent Itemsets using Patricia Tries. Department of Information Engineering, University of Padova, on Proceedings of FIMI'03.

[11] M.B.H Hamida, and Y. Slimani. A Patricia-Tree Approach for Frequent Closed Itemsets. World Academy of Science, Engineering and Technology 4 2005.

[12] School of Computer Science and Software Engineering, Monash University, Australia 3168.[Online] Available at: http://www.csse.monash.edu.au/~lloyd/tildeAlgDS/Tree/PATRICIA/