bidang
TEKNIK

PENERAPAN STRING MATCHING MENGGUNAKAN ALGORITMA BOYER-MOORE PADA TRANSLATOR BAHASA PASCAL KE C

DIANA EFFENDI, TONO HARTONO, ANDRI KURNAEDI
Program Studi Sistem Informasi, FTIK
Universitas Komputer Indonesia

Translator is an automatic solution for translating process of programmatic source code from a source language to a target language. Translator Pascal to C runs to do parsing source code, then it applies the rule of translating. In addition, it presents the result of the target translation. Translator Pascal to C is able to optimize the time needed for users to carry out translation of Pascal to C.

The intoducing and translating function can be created accurately. To gain the rule of translating needs to be compared the syntactic rule from C and Pascal. The grammmar elements that must be explored are the structure of programme, the customs declarations(variable, type, constancy, etc), an expression, a statement, the regulation of the object naming of programme, etc. Using algorithm with fitting string, Boyer-Moore algorithm becomes an important option in translating process. The similarities between Pascal and C enables both of them to be translated each other completely.

Moreover, Pascal and C are an introduction language for the beginner programmers. The first value can influence the end one very much. If in the beginning there have been the difficulties, they will cause many issues around the programming knowledge to the beginner ones next time. Compounding the value of translator that is connected to the mastery improvement of the base programming knowledge, Hopefully it can be created a translator application with the superiority of academic point of view.

Keywords : Translator, Pascal, Boyer-Moore

PENDAHULUAN

Seperti diketahui, proses alih kode perangkat lunak dari satu bahasa ke bahasa lain seringkali menjadi kendala dikalangan programmer. Ada berbagai alasan yang melatarbelakangi hal ini, contohnya ketidakcukupan penguasaan terhadap bahasa target, waktu, referensi, dan lain-lain. Keperluan alih kode bahasa pun dapat terjadi dengan alasan perubahan proses bisnis, integrasi, regulasi perusahaan, adaptasi dengan teknologi baru, hardware, maupun hal lainnya.

Proses alih kode tidaklah mudah, terlebih jika kode sumber aplikasi yang diterjemahkan kompleks dan memiliki ukuran yang besar. Proses alih kode akan membutuhkan banyak waktu dan tenaga, serta rentan terhadap kesalahan. Maka diperlukan suatu solusi untuk melakukan proses alih kode secara otomatis dengan bantuan sebuah alat. Alat yang dimaksud adalah program yang dapat menterjemahkan kode sumber ke bahasa lain yang dikehendaki. Alat tersebut di kenal sebagai translator. Translator adalah program yang menterjemahkan suatu huruf dengan himpunan terkecil hingga terbesar dari bahasa asal ke bahasa target.

Saat ini, banyak kegiatan akademik di bidang ilmu kajian pemrograman menggunakan Pascal sebagai pengenalan ilmu pemrograman. Pascal dipilih karena keluwesan sintaks dan lebih banyak menggunakan bahasa manusia dalam penggunaannya. Bahasa lain yang juga biasa digunakan adalah C. C lebih banyak menggunakan simbol-simbol dalam pengunaannya serta C memiliki nilai lebih dibandingkan Pascal. C lebih mendorong penggunanya untuk berfikir lebih kreatif dalam menyusun logika pemrograman, karena C memiliki kompleksitas dan variasi sintaks yang lebih rumit dibandingkan Pascal.

Pascal dan C memiliki banyak kesamaan, diantaranya sama-sama berparadigma prosedural, penerus seri bahasa ALGOL, waktu pertama kali muncul yang hampir bersamaan, dan lain-lain. C bersifat robust, fleskibel, efisien, ekspresif dan permisif. Pascal memiliki sintaks yang jelas dan mudah dimengerti, sehingga banyak dipakai di lingkungan akademik. C banyak dipakai di lingkungan industri dan sering digunakan sebagai antarmuka dengan software sistem.

Untuk mendapatkan aturan penerjemahan dari Pascal ke C, penulis perlu memadankan sintaksis Pascal dan C. Alih kode pada beberapa aspek bahasa Pascal ke C tidak bisa dilakukan secara *straight-forward* (kata per kata), namun memerlukan proses pencocokan sintaksis untuk menyesuaikan lingkungan Pascal maupun C yang menghasilkan nilai kebenaran maksimal, sehingga masing-masing lingkungan dapat terdefinisi dengan baik. Untuk itu, dibutuhkannya suatu algoritma pencocokan sintaksis dengan efisiensi proses terhadap waktu. Dengan menggunakan Algoritma Boyer Moore sebagai metode pencocokan kata, diharapkan aplikasi ini dapat menghasilkan keluaran dengan kebenaran mutlak.

Berdasarkan latar belakang masalah tersebut, dapat diidentifikasikan beberapa masalah sebagai berikut :

1. Dibutuhkannya banyak waktu dan tenaga dalam proses alih kode Pascal ke C, serta rentan terhadap kesalahan.
2. Alih kode pada beberapa aspek bahasa Pascal ke C tidak bisa dilakukan secara *straight-forward.*
3. Diperlukannya suatu solusi algoritma untuk melakukan proses alih kode Pascal secara otomatis ke bahasa C.

Berdasarkan identifikasi masalah yang mengacu pada permasalahan di atas, maka dirumuskanlah beberapa masalah sehubungan pembangunan Translator, yaitu :

1. Bagaimana alur algoritma translator dapat mengalihkan Pascal ke C secara otomatis.
2. Bagaimana metode penerjemahan kode yang digunakan dapat mengalihkan Pascal ke C dengan benar.
3. Bagaimana translator dapat melakukan pencocokan lingkungan kode Pascal ke C dengan waktu yang relatif singkat.

Alasan utama pembangunan translator ini adalah untuk menciptakan sebuah aplikasi pendukung yang dapat mengalihkan suatu kode bahasa program ke bahasa program yang lain, yaitu Pascal ke C. Sehingga dapat menjadi solusi ketika dibutuhkannya perubahan kode secara besar dari kode Pascal ke C. Sedangkan tujuan dari pembangunan translator Pascal ke C, yaitu :

1. Menemukan alur algoritma translator yang dapat mengalihkan Pascal ke C secara otomatis.
2. Menemukan metode penerjemahan kode yang digunakan dapat mengalihkan Pascal ke C dengan benar.
3. Mengembangkan metode pencocokan lingkungan Pascal ke C dengan pertimbangan lama waktu proses.

Penelitian pembangunan translator memiliki manfaat besar terhadap kegiatan akademik penulis yaitu sebagai pemenuhan syarat kelulusan Strata 1. Dan sebagai media pendalam ilmu di bidang algoritma pemrograman. Dari sudut pandang pengguna, translator Pascal ke C memiliki manfaat sebagai translator otomatis yaitu merubah Pascal ke C secara cepat dan

akurat. Selain itu, translator juga dapat digunakan sebagai pembanding penggunaan sintaksis kode Pascal terhadap C, sehingga membantu pengguna untuk menyimpulkan penggunaan suatu sintaksis Pascal pada C.

Dalam penelitian ini, Translator hanya dapat menterjemahkan kode sumber Pascal ke C, dengan spesifikasi kode sumber sebagai berikut :

1. Pernyataan
   a. Sederhana
      - Pernyataan *Assignment*, ditandai dengan penggunaan operator *assignment* (:=).
      - Pemanggilan procedure *input-keyboard* dan *output-monitor* serta pernyataan goto

   a. Terstruktur
      - Pernyataan *Conditional*, pernyataan seleksi kondisi.
      - Pernyataan *Repetitive*, pernyataan perulangan.
      - Pernyataan *Compound*, gabungan antara pernyataan sederhana dan terstruktur. Ditandai dengan penggunaan "begin" dan "end".

2. Tipe Data Sederhana
   a. Integer
   b. Real
   c. Karakter dan string
   d. Boolean
3. Ekspresi
   a. Numerik / aritmatika
   b. String
   c. Boolean / logika

Dari spesifikasi di atas, kode sumber yang dimaksud adalah kode sumber sub-program tanpa keikutsertaan bagian deklarasi dan lainnya, dengan begin – end, if – else, while – do, repeat – until, for – do, goto, *Input* (read/readln ke scanf, masukan dari *keyboard*), *Output* (write/writeln ke printf, keluaran ke layar) sebagai pernyataan yang dapat dialihkan. Selain itu, tranlator hanya dapat dijalankan pada Sistem Operasi Windows XP dan setelahnya.

KAJIAN PUSTAKA

Algoritma adalah urutan langkah-langkah logis penyelesaian masalah yang disusun secara sistematis dan logis. Kata logis merupakan kata kunci dalam algoritma. Langkah-langkah dalam algoritma harus logis dan harus dapat ditentukan bernilai salah atau benar. Dalam beberapa konteks, algoritma adalah spesifikasi urutan langkah untuk melakukan pekerjaan tertentu.

Pertimbangan dalam pemilihan algoritma adalah algoritma haruslah benar. Artinya algoritma akan memberikan keluaran yang dikehendaki dari sejumlah masukan yang diberikan. Tidak peduli serumit apapun algoritma, jika memberikan keluaran yang salah, pastilah algoritma tersebut bukanlah algoritma yang baik.

Pertimbangan lain yang harus diperhatikan adalah seberapa baik hasil yang dicapai oleh algoritma tersebut. Hal ini penting terutama pada algoritma untuk menyelesaikan masalah yang memerlukan aproksimasi hasil. Algoritma yang baik harus mampu memberikan hasil yang sedekat mungkin dengan nilai yang sebenarnya.

Selanjutnya adalah efisiensi algoritma. Efisiensi algoritma dapat ditinjau dari 2 hal yaitu efisiensi waktu dan memori. Meskipun algoritma memberikan keluaran yang benar, tetapi jika harus menunggu lama untuk mendapatkan keluarannya, algoritma tersebut bukanlah algoritma yang baik, setiap orang menginginkan keluaran yang cepat.

Dalam kenyataannya, setiap orang bisa membuat algoritma yang berbeda untuk menyelesaikan suatu permasalahan, walaupun terjadi perbedaan dalam menyusun algoritma, tentunya diharapkan keluaran yang sama.

Pascal merupakan pengembangan dari bahasa Algol 60, bahasa pemrograman untuk sains komputasi. Tahun 1960, beberapa ahli komputer bekerja untuk mengembangkan bahasa Algol, salah satunya adalah Dr. Niklaus Wirth dari Swiss Federal Institute of Technology (ETH-Zurich),

yang merupakan anggota grup yang membuat Algol.

Tahun 1971, diterbitkan suatu spesifikasi untuk *highly-structured language* (bahasa tinggi yang terstruktur) yang menyerupai Algol yaitu Pascal. Pascal bersifat data oriented, yaitu programmer diberi keleluasaan untuk mendefinisikan data sendiri. Pascal juga merupakan *teaching language* (banyak dipakai untuk pengajaran tentang konsep pemrograman). Kelebihan yang lain adalah penulisan kode Pascal yang luwes.

Susunan dari kode-kode dalam teks Pascal harus ditulis secara terurut, pernyataan-pernyataan yang ditulis lebih awal akan dieksekusi lebih dahulu. Oleh karena itu, suatu pernyataan yang menyangkut suatu variabel di dalam program, maka variable itu harus terdefinisi dahulu sebelumnya. Hal ini terutama menyangkut pada pemanggilan sub-program oleh sub-program yang lain.

Blok dengan batas-batas yang jelas.Pascal memberikan pembatas yang jelas pada tiap-tiap blok, seperti pada blok program utama, sub-program, struktur kontrol (pengulangan/ pemilihan), dll. Pemakaian kata kunci "begin" untuk mengawali operasi pada blok dan "end" untuk menutupnya memudahkan programmer menyusun programnya dengan mudah.

Berasal dari bahasa BCPL (*Basic Combined Programming Language*) oleh Martin Richard, Cambridge tahun 1967, Ken Thompson membuat bahasa B untuk dipakai pada komputer DEC PDP-7 dibawah sistem operasi UNIX pada Bell laboratory, Murray Hill, New Jersey tahun 1970.

Bahasa B merupakan suatu bahasa pemrograman yang tidak memiliki jenis suatu data seperti halnya PL/M. Berdasarkan gambaran bahasa B, Dennis Ritchie menulis bahasa C (Abdul Kadir 2003). Nama C diambil berdasarkan urutan sesudah B dari bahasa BCPL. Tujuan bahasa C pada mulanya untuk membentuk suatu sistem operasi yang akan digunakan pada mesin komputer DEC PDP-11 yang baru.

Pada tahun 1975, sistem operasi UNIX versi 6 dan bahasa C mulai diberikan kepada Universitas maupun Akademi. Dan pada tahun 1979, sistem operasi UNIX versi 7 dikeluarkan dengan bahasa C. Sistem operasi ini (versi 7) seluruhnya ditulis dalam bahasa C.

Pada 1978 Dennis Ritchie dan Brian Kernighan kemudian mempublikasikan buku *The C Programming Language* yang semakin memperluas pemakaiannya dan dijadikan standar oleh ANSI *(American National Standard Institute)* pada tahun 1989. C kemudian dikembangkan lagi oleh Bjarne Stroustrup menjadi C++ (1986). C dan/atau C++ banyak digunakan sebagai bahasa pemrograman untuk membuat sistem operasi.

Program C pada hakekatnya tersusun atas sejumlah blok fungsi. Sebuah program minimal mengandung sebuah fungsi. Fungsi pertama yang harus ada dalam program C dan sudah ditentukan namanya adalah main(). Setiap fungsi terdiri atas satu atau beberapa pernyataan, yang secara keseluruhan dimaksudkan untuk melaksanakan tugas khusus. Bagian pernyataan fungsi (sering disebut tubuh fungsi) diawali dengan tanda kurung kurawal buka ({) dan diakhiri dengan tanda kurung kurawal tutup (}). Di antara kurung kurawal itu dapat dituliskan statemen-statemen program C.

Namun pada kenyataannya, suatu fungsi bisa saja tidak mengandung pernyataan sama sekali. Walaupun fungsi tidak memiliki pernyataan, kurung kurawal haruslah tetap ada. Sebab kurung kurawal mengisyaratkan awal dan akhir definisi fungsi. Bahasa C dikatakan sebagai bahasa pemrograman terstruktur karena strukturnya menggunakan fungsi-fungsi sebagai program-program bagiannya (subroutine). Fungsi-fungsi yang ada selain fungsi utama (main()) merupakan program-program bagian. Fungsi-fungsi ini dapat ditulis setelah fungsi utama atau diletakkan di file pustaka (library). Jika fungsi-fungsi diletakkan di file pustaka dan akan dipakai

di suatu program, maka nama file judulnya (header file) harus dilibatkan dalam program yang menggunakannya dengan preprocessor directive berupa #include.

String matching adalah pencarian sebuah *pattern* pada sebuah teks (Ronald L. Rivest dkk. 1994). Prinsip kerja algoritma string matching adalah sebagai berikut:

1. Memindai teks dengan bantuan sebuah window yang ukurannya sama dengan panjang *pattern*.
2. Menempatkan window pada awal teks.
3. Membandingkan karakter pada window dengan karakter dari *pattern*. Setelah pencocokan (baik hasilnya cocok atau tidak cocok), dilakukan shft ke kanan pada window. Prosedur ini dilakukan berulang-ulang sampai window berada pada akhir teks. Mekanisme ini disebut mekanisme sliding-window.

Algoritma string matching mempunyai tiga komponen utama, yaitu:

1. *Pattern*, yaitu deretan karakter yang akan dicocokkan dengan teks, dinyatakan dengan x[0..m-1], panjang *pattern* dinyatakan dengan m.
2. Teks, yaitu tempat pencocokan *pattern* dilakukan, dinyatakan dengan y[0..n-1], panjang teks dinyatakan dengan n.
3. Alfabet, yang berisi semua simbol yang digunakan oleh bahasa pada teks dan *pattern*, dinyatakan dengan $\sum$ dengan ukuran dinyatakan dengan ASIZE.

Algoritma Boyer-Moore termasuk algoritma string matching yang paling efisien dibandingkan algoritma-algoritma string matching lainnya (Ronald L. Rivest dkk. 1994). Karena sifatnya yang efisien, banyak dikembangkan algoritma string matching dengan bertumpu pada konsep algoritma Boyer-Moore, beberapa di antaranya adalah algoritma Turbo BM dan algoritma Quick Search.

Algoritma Boyer-Moore menggunakan metode pencocokan string dari kanan ke kiri yaitu memindai karakter *pattern* dari kanan ke kiri dimulai dari karakter paling kanan. Algoritma Boyer-Moore menggunakan dua fungsi shift yaitu good-suffix shift dan bad-character shift untuk mengambil langkah berikutnya setelah terjadi ketidakcocokan antara karakter *pattern* dan karakter teks yang dicocokkan.

Stack adalah suatu bentuk khusus dari linear list (suatu struktur data yang merupakan himpunan terurut yang dapat berkurang atau bertambah setiap saat) di mana operasi penyisipan dan penghapusan atas elemen-elemennya hanya dapat dilakukan pada satu sisi saja yang disebut sebagai "TOP". Ada empat operasi dasar yang didefinisikan pada stack, yaitu:

1. Create, operator ini berfungsi untuk membuat sebuah stack kosong.
2. IsEmpty, operator ini berfungsi untuk menentukan apakah suatu stack adalah stack kosong.
3. Push, operator ini berfungsi untuk menambahkan satu elemen ke dalam stack.
4. Pop, operator ini berfungsi untuk mengeluarkan satu elemen dari dalam stack.

Dalam penggunaannya, untuk menempatkan stack biasanya digunakan sebuah array. Tetapi perlu diingat di sini bahwa stack dan array adalah dua hal yang berbeda. Selain itu penggunaan stack dengan array dirasakan kurang tepat. Penggunaan stack pada pembangunan Translator ini menggunakan record sebagai implementasi stack.

Tree/pohon merupakan struktur data yang tidak linear/non linear yang digunakan terutama untuk merepresentasikan hubungan data yang bersifat hierarkis antara elemen-elemennya. Kumpulan elemen yang salah satu elemennya disebut dengan *root* (akar) dan sisa elemen yang lain disebut sebagai simpul (*node/vertex*) yang terpecah menjadi sejumlah himpunan yang tidak saling berhubungan satu sama lain, yang disebut s*ubtree*/cabang.

Sebuah pohon biner T dapat didefinisikan sebagai sekumpulan terbatas dari elemen-elemen yang disebut nodes/ simpul dimana :

a. T dikatakan kosong (disebut null *tree*/ pohon null atau *empty tree*/pohon kosong)

b. T terdiri dari sebuah node khusus yang dipanggil R, disebut root dari T dan node-node T lainnya membentuk sebuah pasangan terurut dari *binary tree* T1 dan T2 yang tidak berhubungan yang kemudian dipanggil *subtree* kiri dan *subtree* kanan.

Jika T1 tidak kosong maka rootnya disebut *successor* kiri dari R dan jika T2 tidak kosong, maka rootnya disebut *successor* dari R. Jenis-jenis *binary tree* :

1. *Complete Binary Tree*
   Suatu *binary tree* T akan disebut *complete*/lengkap jika semua levelnya memiliki *child* 2 buah kecuali untuk level paling akhir. Tetapi pada akhir level setiap *leaf*/daun muncul terurut dari sebelah kiri.
2. *Extended Binary Tree : 2-Tree*
   Sebuah *binary tree* dikatakan 2-*tree* atau *extended binary tree* jika tiap simpul N memiliki 0 atau 2 anak. Simpul dengan 2 anak disebut dengan simpul internal (internal node), dan simpul dengan 0 anak disebut dengan external node. Kadang-kadang dalam diagram node-node tersebut dibedakan dengan menggunakan tanda lingkaran untuk internal node dan kotak untuk eksternal node.

OBJEK DAN METODE PENELITIAN

Metodologi penelitian yang digunakan dalam penelitian ini menggunakan metode studi kepustakaan, dimana studi kepustakkan yang dilakukan dalam penelitian ini bertujuan untuk mencari sumber data sekunder yang akan mendukung penelitian dan untuk mengetahui sampai ke mana ilmu yang berhubungan dengan penelitian terkait dengan algoritma *string matching* telah berkembang, sampai ke mana terdapat kesimpulan dan degeneralisasi yang pernah dibuat.

Langkah yang dilakukan dalam studi kepustakaan yaitu mengidentifikasikan teori secara sistematis, penemuan pustaka, dan analisis dokumen yang memuat informasi yang berkaitan dengan Algoritma *String Matching*.

Idealnya, suatu teks akan dikenali sebagai pernyataan Pascal jika teks tersebut telah sukses melalui proses *compile* pada Turbo Pascal 7.0. Pernyataan Pascal tidak bisa secara langsung diterjemahkan ke bentuk C. Proses penerjemahan pun harus memiliki kualifikasi yang baik dari sudut lama waktu proses yang dibutuhkan. Maka harus diterapkannya beberapa fungsi pengenalan, diantaranya:

1. Penyesuaian algoritma Boyer-Moore pada setiap kasus pencocokan.
2. Proses pencocokan string terhadap teks, dimaksudkan agar setiap kata dapat dikenali oleh translator dan dapat diterjemahkan.
3. Aturan terjemahan, dimaksudkan sebagai aturan yang diterapkan jika ditemukannya sebuah bentuk pernyataan di dalam teks.
4. Mengenali jenis ekspresi yang dioperasikan dan menemukan bentuk hasil operasi dari Pascal ke C.
5. Proses *handling* untuk setiap bentuk penyataan yang tidak memiliki aturan penerjemahan.
6. Metode pemberian pesan kesalahan.

HASIL PENELITIAN

Pengggunaan *stack* dengan *single link list* dimaksudkan sebagai manajemen memori pada *runtime* Translator. *Single link list* difungsikan sebagai media penyimpanan data sementara hasil proses algoritma Boyer-Moore. Dimana *stack* memiliki nilai batas penyimpanan yang dinamis. Sehingga dapat memaksimalkan alokasi memori terhadap kebutuhan Translator yang sedang berjalan. Di Visual Basic 6.0, *stack* dengan *single link list* dapat diimplementasikan sebagai *collection control*.

Representasi metode pemanfaatan *collection* untuk mendukung kinerja Translator dengan diasumsikan BmBc sebagai *class module*, Pos dan Char sebagai elemen *class module* (Gambar 1 dan 2).

Sama halnya dengan implementasi *stack*, *binary tree* difungsikan sebagai solusi pemecahan hirarki operator pada sebuah ekspresi (Gambar 3). Pada implementasinya, *binary tree* hanya digunakan sebagai tempat penyimpanan sementara dari potongan-potongan teks yang dikenali sebagai sebuah ekpresi. Berikut algoritma penerapan *binary tree* pada translator:

1. Lakukan pencarian posisi dengan menggunakan prosedur BM untuk 2 kata kunci pada teks. Kata kunci-1 merupakan kata kunci dengan arti buka (tanda yang menyebutkan sebuah awalan) dan Kata kunci-2 merupakan kata kunci dengan arti tutup (tanda yang menyebutkan sebuah akhiran). Contoh, kata kunci-1 : "(" dan kata kunci-2 : ")"
2. Jika jumlah posisi kedua kata kunci yang ditemukan tidak sama (tidak sepasang) maka berikan pesan kesalahan.
3. Jika jumlah posisi kedua karakter sama, maka bandingkan antara posisi paling pertama dari kata kunci-1 terhadap posisi paling pertama dari kata kunci-2 pada teks.
4. Jika kata kunci-1 memiliki nilai perbandingan terbesar, maka proses ulang (*rekursif*) teks dengan menambahkan kata kunci awalan dan akhiran pada kedua ujung teks.
5. Tambahkan setiap nilai posisi yang ditemukan ke dalam *binary tree. Binary tree* pada posisi ini merupakan *binary tree* yang tersusun dengan 1 *subtree* kanan, 0 *subtree* kiri. Difungsikan untuk memudahkan dalam pengurutan *node.*
6. Urutkan setiap posisi yang berkesesuaian (pasangan) antara posisi kata kunci-1 dan 2 dengan menggunakan metode pengurutan *Pre-Order. Pre-Order* dimaksudkan agar pengolahan operasi terjadi dari posisi buka-tutup terluar (hirarki terkecil). Langkah *Pre-Order* -dilakukan dengan mengunjungi root *Bineary Tree,* mengunjungi *subtree* kiri, lalu *subtree* kanan secara rekursif.
7. Pada setiap proses pengunjungan, dilakukan perubahan bentuk pada *binary tree* hingga menjadi sebuah *binary tree transversal* dengan *-Pre-Order* sebagai tolok ukur struktur *tree*.
8. Setelah posisi-posisi kata kunci-1 dan 2 telah tersusun sebagaimana semestinya, terjemahkan bagian teks yang berada pada posisi yang berkesesuaian terhadap masing-masing posisi pada *binary tree.* Proses pengunjungan *subtree* dilakukan dengan metode *Post-Order* secara rekursif, dengan tujuan agar teks dapat terpecah dari bagian paling dalam (hirarki terbesar).

Fungsi utama pada Translator terhadap teks yang dibentuk menjadi pernyataan atau serangkaian pernyataan Pascal adalah pencocokan suatu *pattern* tertentu yang diasumsikan sebagai pemisah rangkaian pembentuk pernyataan Pascal. Pencocokan ini dilakukan untuk memenggal teks berdasarkan posisi *pattern* terhadap teks, sehingga teks dapat dikenali sebagai pernyataan Pascal dan diterjemahkan ke pernyataan C.

Kaidah pertama dari algoritma Boyer-Moore ini adalah masing-masing karakter dari *pattern* pencocokan harus mempunyai kode ASCII yang sama terhadap target karakter yang ada pada teks serta karakter-karakter antara *pattern* dan target harus terurut sama persis.

Kaidah yang kedua adalah hasil penerjemahan teks berdasarkan pencocokan *pattern* akan bernilai benar secara utuh jika teks telah sukses melalui proses compile pada compiler Turbo Pascal 7.0. Walaupun demikian, Translator menerapkan fungsi-fungsi yang dapat memastikan kebenaran sintaksis pernyataan berdasarkan perolehan posisi *pattern* terhadap teks dan mengabaikan beberapa kesalahan sederhana pada sintaksis sehingga dapat diterjemahkan ke bentuk C. Dengan kata lain, suatu teks memungkinkan diterjemahkan secara utuh, sebagian, atau tidak sama sekali. Untuk itu, dibutuhkan suatu basis pengetahuan dalam pencocokan posisi dan pemenggelan

pattern terhadap teks.

Secara Umum, langkah-langkah dari teknik penerapan Boyer-Moore pada pencocokan *pattern* pada teks dapat direpresentasikan sebagai berikut :

1. Menjalankan prosedur preBmBc dan preBmGs untuk mendapatkan nilai perbandingan pergeseran.
2. Menjalankan prosedur preBmBc. Fungsi dari prosedur ini adalah untuk menentukan berapa besar pergeseran yang dibutuhkan untuk mencapai karakter tertentu pada *pattern* dari karakter *pattern* terakhir/terkanan. Hasil dari prosedur preBmBc disimpan pada stack bmBc (Gambar 4).
3. Menjalankan prosedur preBmGs. Sebelum menjalankan isi prosedur ini, prosedur *suffix* dijalankan terlebih dulu pada *pattern*. Fungsi dari prosedur *suffix* adalah memenggal sejumlah karakter yang dimulai dari karakter terakhir/ terkanan dengan sejumlah karakter yang dimulai dari setiap karakter yang lebih kiri. Hasil dari prosedur *suffix* disimpan pada stack suff. Dengan demikian suff[i] mencatat panjang dari *suffix* yang cocok dengan segmen dari *pattern* yang di akhiri karakter ke-i.
4. Dengan prosedur preBmGs, dapat diketahui berapa banyak langkah pada *pattern* dari sebuah segmen ke segmen lain yang sama, dimana letaknya lebih kiri dengan karakter di sebelah kiri segmen yang berbeda. Prosedur preBmGs menggunakan stack *suff* untuk mengetahui semua pasangan segmen yang sama (Gambar 5).

Menjalankan prosedur BM, proses pencocokan *pattern* dengan menggunakan hasil dari prosedur preBmBc dan preBmGs yaitu stack bmBc dan bmGs sebagai landasan keputusan pergeseran (Gambar 6). Proses pengambilan nilai BmBc dan BmGs pada database dimaksudkan agar proses perhitungan nilai keputusan pergeseran oleh prosedur BmBc dan BmGs dapat menjadi lebih optimal, karena nilai keputusan telah tersimpan untuk beberapa *pattern* dengan nilai ketetapan. Namun proses filter pada database (recordset) membutuhkan estimasi waktu tertentu. Maka jika dibandingkan proses pencocokan nilai keputusan oleh prosedur preBmBc dan preBmGs untuk *pattern* yang lebih pendek, proses cek database akan menjadi lebih lama. Untuk itu, keputusan pengecekan database ditentukan dari parameter prosedur BM pada saat pemanggilan.

Terdapat pula parameter plusPos yang berfungsi sebagai nilai tambah untuk setiap posisi yang ditemukan. plusPos digunakan pada saat pencocokan string yang membutuhkan posisi akhir, tengah, maupun posisi tertentu dari sebuah karakter yang terdapat pada pattern terhadap teks.

Terdapat pula kriteria jumlah pencocokan, kriteria ini difungsikan sebagai parameter jumlah pencocokan yang dibutuhkan. Sebagai contoh, jika terdapat 10 kecocokan, maka pencocokan akan dihentikan ketika mencapai kecocokan ke-5, dimana 5 adalah kriteria jumlah pencocokan. Sehingga menjadikan algoritma Boyer-Moore berjalan lebih optimal dan sesuai kebutuhan Translator.

Basis pengetahuan algoritma Translator Pascal ke C dibangun oleh 2 dasar sintaksis seperti yang telah dijelaskan sebelumnya. Berikut representasi sintaksis pemenggalan teks berdasarkan sintaksis pernyataan Pascal (Gambar 7). Masing-masing pernyataan memiliki alur yang bernilai tetap. Alur-alur tersebut membentuk sebuah pola pembentukan pernyataan yang dapat dimanfaatkan oleh algoritma Translator untuk mengkondisikan kriteria pencocokan *pattern* algoritma Boyer-Moore terhadap teks. Terlihat pula bahwa suatu pernyataan dapat memiliki badan berupa pernyataan lainnya, dan tentunya hal ini berdasarkan pengkondisian pernyataan induk. Maka dengan mengaitkan per pola variasi pernyataan Pascal terhadap C, didapatlah suatu aturan baru mengenai pemenggalan dan penggabungan hasil dari pencocokan algoritma Boyer-Moore terhadap teks.

Dengan pemanfaatan Database

Microsoft Accsess 2003 sebagai media penyimpanan data, pengkondisian pola sintaksis dirancang dan disimpan berdasarkan keterkaitan yang telah dijelaskan pada pembahasan sebelumnya. Sehingga dapat digunakan oleh Translator sebagai basis pengetahuan pemberian *pattern* pencocokan algoritma Boyer-Moore dan menvalidasi nilai posisi yang dihasilkan. Sehingga didapatkan nilai posisi pemenggalan suatu teks berdasarkan sintaksis Pascal, dan penggabungan teks berdasarkan sintaksis C.

Dalam penggunaannya, Translator menggunakan enam query sebagai Control Recordset Visual Basic 6.0 untuk mendukung dan meningkatkan proses pencocokan dan penerjemahan.

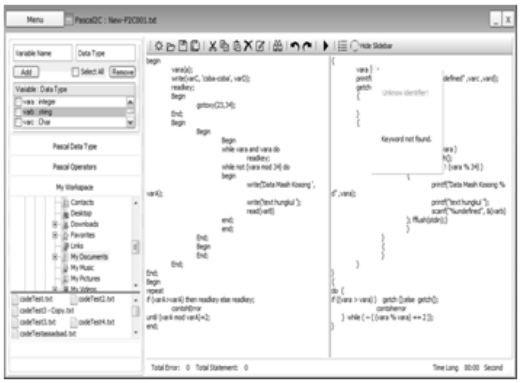
IMPLEMENTASI

Translator Mode Ruang Edit Standar

Pada awal penggunaan program, pengguna diharuskan menginputkan pada bagian deklarasi yang terdapat pada slidebar.

Gambar 8
Translator Mode Ruang Edit Standar – Masukan Awal

Pada slidebar, seperti yang terlihat pada gambar terdapat inputan textbox. Pengguna diharuskan mendaftarkan variabel yang akan digunakan dengan mengikut sertakan tipe data dari variable tersebut. Proses hapus dapat dengan memilih variabel terlebih dahulu pada box di bawahnya, lalu menekan tombol hapus untuk menghilangkan variabel dari daftar.



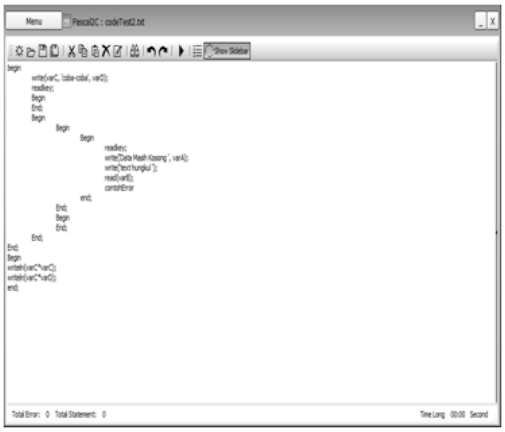
Gambar 9
Translator Mode Ruang Edit Standar – Masukan Akhir

Setelah mendaftarkan variabel, pengguna bebas untuk melakukan proses edit teks. Selayaknya editor standar, pengguna dapat menggunakan fungsi *copy, paste,* dll. Untuk melakukan *tracking file*, pengguna diharuskan memilih direktori terlebih dahulu, lalu klik ganda pada item yang ditemukan. Fungsi tracking ditemukan pada frame My Workspace.

Sedangkan frame-frame lainnya pada slidebar, user tidak diharuskan untuk melakukan eksekusi fungsi. Frame-frame tersebut berisi informasi mengenai tipe data dan operator pada Pascal.

Translator Mode Ruang Edit Penuh

Pada penggunaanya, mode edit penuh dan mode standar memiliki kesamaan fungsionalitas yang sama. Hanya saja, ruang untuk melakukan edit teks pada mode ini dirancang lebih lebar.



Gambar 10
Translator Mode Ruang Edit Penuh – Masukan Awal

Terlihat dari gambar di atas, mode ini menghilangkan slidebar. Dan untuk mengaktifkan mode ini, digunakan tombol toolbar paling kanan "Hide/Show Slidebar". Begitu pula jika pengguna ingin menonaktifkannya kembali.

Toolbar Translator

Gambar 11 Toolbar Translator

Secara garis besar, penggunaan keseluruhan fungsi-fungsi yang ada pada toolbar hampir sama persis dengan fungsi-fungsi pada editor teks lainnya. Perbedaan terjadi pada 3 tombol terakhir dari paling kanan. Tombol pertama berfungsi untuk menterjemahkan masukan, tombol kedua untuk menampilkan dan menyembunyikan editor teks untuk kedua sisi bahasa. Dan yang terakhir adalah untuk menyembunyikan slidebar/mode edit penuh.

Kebutuhan Perangkat Lunak dan Perangkat Keras

Perangkat yang dibutuhkan dalam instalasi Translator ini terdiri dari 2 bentuk perngkat, yaitu perangkat keras dan perangkat lunak.
Perangkat keras yang dibutuhkan yaitu:

1. Unit komputer lengkap (netbook), prosessor Intel Atom CPU N280, RAM 2 Gb, VGA 256 Mb, tersedia Mouse (atau yang setara)
2. Perangkat lunak yang dibutuhkan sebagai sistem operasi yang ada pada komputer dan sebagai media pengembangan Translator antara lain:
   - Sistem Operasi Windows 7
   - Program Visual Basic 6.0
   - Program Microsoft Access 2003

KESIMPULAN DAN SARAN

Kesimpulan yang dapat diambil setelah memaparkan pembahasan-pembahasan yang telah disajikan pada bab-bab sebelumnya antara lain:

1. Penerapan sebuah pencocokan string dapat dilakukan dengan menggunakan bermacam-macam algoritma pencocokan string, salah satunya adalah algoritma Boyer-Moore.
2. Algoritma Boyer-Moore terbukti sebagai metode pencocokan string yang sangat efisien. Perbandingan hasil dan waktu sangat berimbang dan sangat sesuai untuk diterapkan pada pencocokan string yang memiliki panjang *pattern* pencarian kecil ataupun besar terhadap objek yang memiliki panjang yang besar.

Saran yang dapat diberikan penulis untuk pembaca yang berkeinginan mengembangkan Translator ini adalah :

1. Translator dapat dikembangkan menjadi translator penuh dari sebuah kode sumber Pascal ke C.
2. Dengan mengeneralisasikan algoritma translator, translator dapat dikembangkan menjadi translator bahasa pemrograman selain Pascal ke C.

DAFTAR PUSTAKA

Cormen, Thomas. H., Leiseson , Charles. E., Rivest , Ronald. L., 1994, *Algorithm.* McGraw-Hill Book Company, North America.

Effendi , Diana., Kurnaedi, Andri., 2012, *Pengembangan Algoritma Boyer Moore pada Translator Bahasa Pemrograman*, Jurnal Informatika Universitas Kristen Maranatha, Bandung.

Kadir , Abdul., 2003, *Pemrograman Dasar Turbo C Untuk IBM .* Andy Offset, Yogyakarta.

http://*www-igm.univ-mlv.fr/~lecroq/string/node14.html#SECTION00140/* 2 Februari 2011

http://*www-igm.univ-mlv.fr/~lecroq/string/node14.html#SECTION00140/* 2 Februari 2011

http://www.informatika.org/~rinaldi/Stmik/2010-2011/Makalah2010/MakalahStima2010-058.pdf 4 Maret 2011

http://*satriaskyterror.wordpress.com/2010/10/22/perbedaan-pascal-dan-c/* 6 Februari 2011

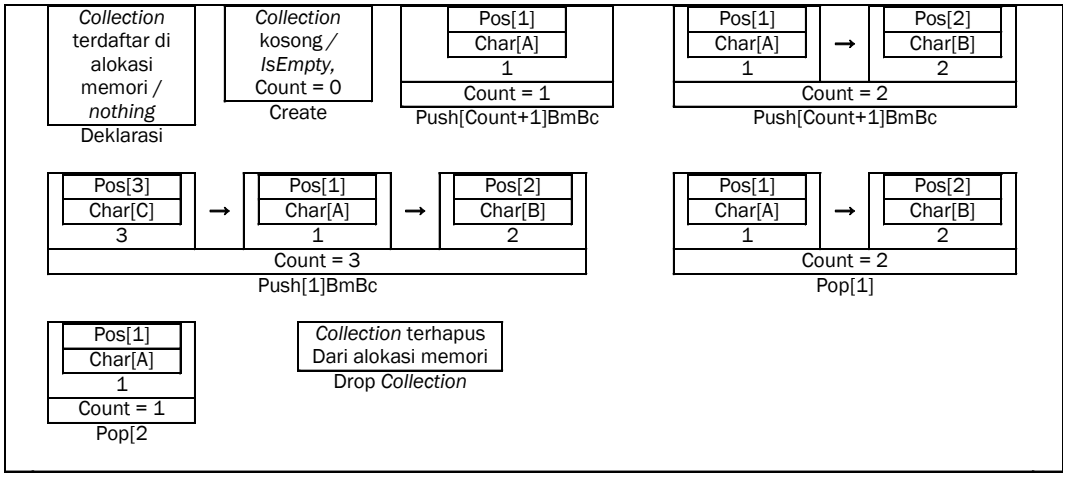
http://*www.informatika.org/~rinaldi/Stmik/2007-2008/Makalah2008/MakalahIF2251-2008-053.pdf* 6 Februari 2011

http://*writerguy.users.btopenworld.com/Pascal/PascalFrameset.html* 20 Februari 2011

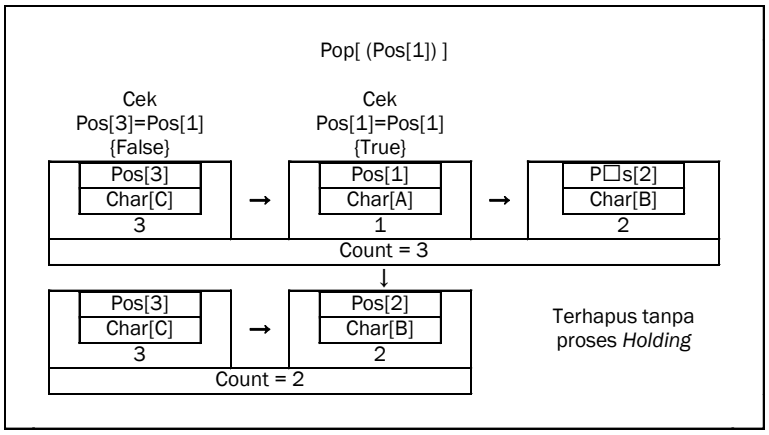
http://*www.inf.fh-flensburg.de/lang/algorithmen/pattern/bmen.htm* 3 Maret 2011

http://*if.unikom.ac.id/andri/download/strukdat/TREE.pdf* 13 April 2011

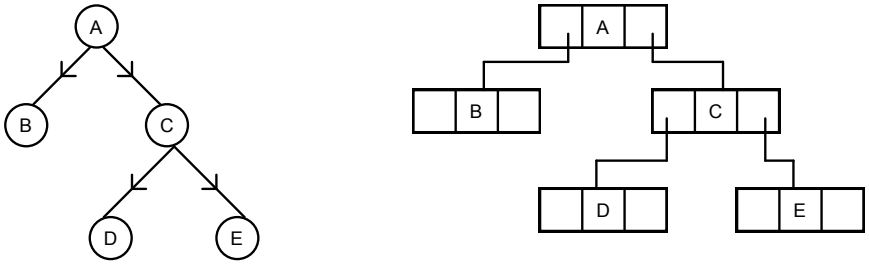
DAFTAR GAMBAR



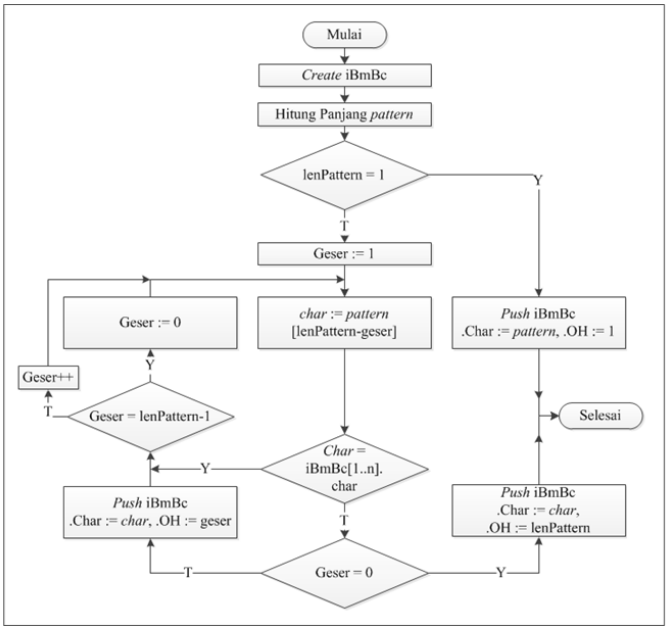
Gambar 1 Operasi standar pada *collection* Visual Basic 6.0.



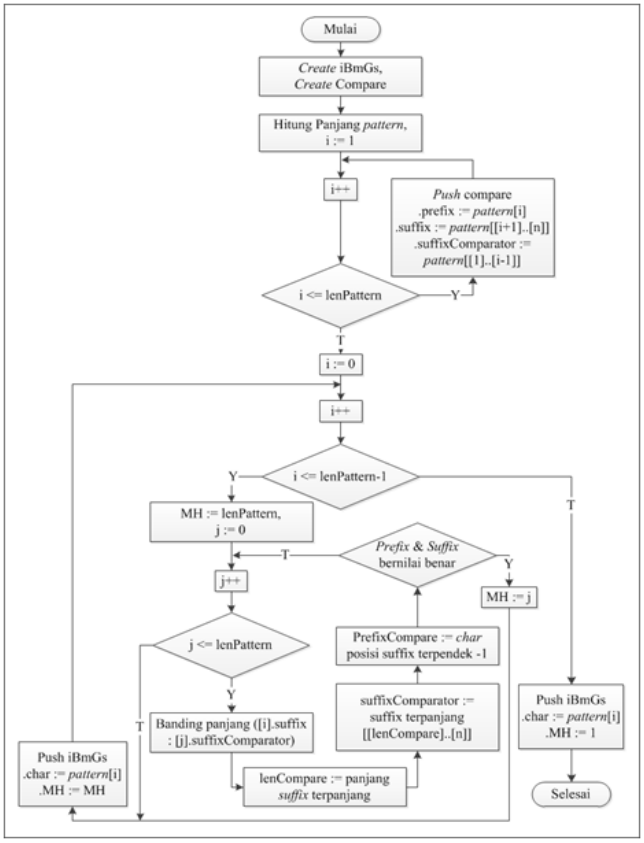
Gambar 2 Kelebihan operasi *collection Control* pada Visual Basic 6.0.



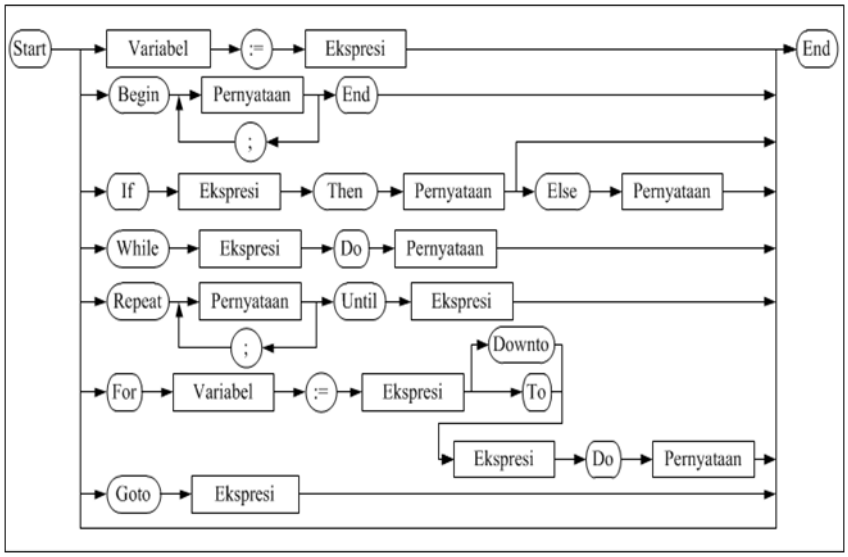
Gambar 3 Representasi *binary tree*



Gambar 4 Flowchart pengembangan prosedur preBmBc



Gambar 6 Flowchart detail pengembangan algoritma Boyer-moore



Gambar 7 Flowchart Sintaksis Pernyataan