Aplikasi Pohon Pengurai pada Natural Language Processing

Albertus Djauhari Djohan - 13515054
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13515054@std.stei.itb.ac.id

***Abstrak*—Manusia sudah sejak lama memimpikan adanya entitas kecerdasan buatan (*Artificial Intelligent)* yang mampu berpikir dan mengambil keputusan sendiri dengan meniru pola pikir manusia. Kecerdasan buatan pertama kali ditemukan pada mitologi Yunani, yaitu cerita tentang Hephaestus dan Pygmalion. Sejak komputer ditemukan, kecerdasan buatan berkembang dengan pesat. Salah satu contoh kecerdasan buatan adalah kecerdasan buatan yang mampu memproses bahasa manusia (*Natural Language Processing*). Ada dua komponen dalam melakukan pemrosesan bahasa manusia, yaitu proses memahami bahasa manusia (*Natural Language Understanding*) dan proses membangkitkan bahasa manusia (*Natural Language Generation*). Dalam makalah ini, akan dijelaskan pengaplikasian *parsing tree* (pohon pengurai) pada proses pemahaman bahasa manusia.**

Kata Kunci — Natural Language Understanding, parsing tree, syntactic analysis, Context Free Grammar

I. PENDAHULUAN

Berbagai teknologi diciptakan manusia untuk mempermudah hidupnya. Kendaraan, mesin cuci, kompor listrik, dan sebagainya adalah beberapa contoh teknologi yang mempermudah kehidupan manusia. Dari berbagai teknologi hasil penemuan manusia, salah satu yang paling penting adalah komputer.

Penemuan komputer membuka banyak cabang ilmu baru, salah satunya adalah kecerdasan buatan (*Artificial Intelligent*). Ada sangat banyak penerapan kecerdasan buatan dalam kehidupan sehari-hari. AI dalam permainan catur, mobil pintar (*Smart Car*), rekomendasi video di youtube, dan asisten personal adalah beberapa contoh kecerdasan buatan dalam kehidupan sehari-hari.

Salah satu contoh kecerdasan buatan yang pembuatannya paling sulit adalah AI yang mampu memproses bahasa manusia (*Natural Language Processing*). Contoh AI yang mampu memproses bahasa manusia adalah Siri milik Apple dan Cortana milik Microsoft.

Gambar 1 : Contoh kecerdasan buatan yang dapat memproses bahasa manusia, Siri buatan Apple.

Pemrosesan bahasa manusia melibatkan dua buah komponen, yaitu proses pemahaman bahasa manusia (*Natural Language Understanding*) dan proses membangkitkan (menghasilkan) bahasa manusia (*Natural Language Generation*). Di antara kedua proses ini, yang paling sulit untuk dilakukan adalah proses pemahaman bahasa manusia yang disebabkan beberapa kendala yang akan dijelaskan lebih lanjut.

Bagian yang akan dibahas secara mendetail pada makalah ini adalah proses pemahaman bahasa manusia yang menggunakan pohon pengurai (*parsing tree*).

II. DASAR TEORI

2.1 Definisi Pohon

Pada dasarnya, pohon dalam ilmu komputer didefinisikan sebagai suatu graf khusus yang tidak berarah, terhubung, tidak memuat sirkuit.

Berdasarkan definisi tersebut, dapat disimpulkan sifat-sifat dari pohon adalah sebagai berikut.

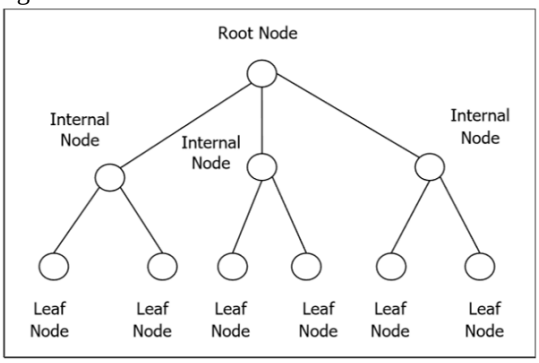
- Memiliki n buah simpul dan tepat n-1 buah sisi.
- Tidak memiliki sirkuit.
- Sepasang simpul dalam pohon terhubung dengan tepat satu lintasan tunggal.
- Penambahan satu sisi pada pohon akan membentuk sebuah sirkuit.

2.2 Pohon Merentang (*Spanning Tree*)

Pohon merentang adalah istilah yang digunakan untuk menyebut subgraf dari suatu graf yang mencakup semua simpul dari graf tersebut dan memenuhi sifat-sifat pohon. Pohon merentang tidak dibahas lebih lanjut dalam makalah ini karena tema dari makalah ini tidak mengaplikasikan pohon merentang, tetapi pohon pengurai yang merupakan pohon berakar.

2.3 Pohon Berakar (*Rooted Tree*)

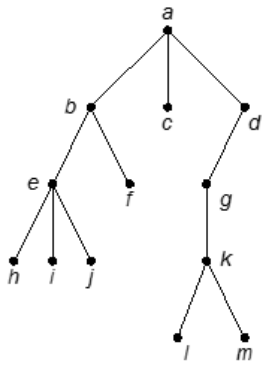
Pohon berakar adalah pohon yang salah satu simpulnya diperlakukan sebagai akar dan simpul-simpul lainnya sebagai anak.



Gambar 2 : Pohon berakar dan komponen-komponennya

Berikut adalah terminologi dari pohon berakar beserta penjelasannya.

2.3.1 Anak (*Child*) dan Orang Tua (*Parent*)



Gambar 3 : Pohon dengan *parent* dan *child*

Pada gambar 3, simpul a adalah orang tua dari simpul b, c, dan d. Sebaliknya, simpul b, c, dan d adalah anak dari simpul a.

2.3.2 Lintasan (*path*)

Lintasan pada pohon adalah sisi-sisi yang menghubungkan antara kedua simpul. Pada gambar 3, contoh lintasan adalah lintasan dari simpul a ke simpul k, yaitu a-d, d-g, g-k.

2.3.3 Saudara Kandung (*Siblings*)

Siblings adalah dua simpul yang memiliki *parent* yang sama. Sebagai contoh, pada gambar 3 b dan c adalah *siblings* karena keduanya memiliki *parent* yang sama yaitu simpul a.

2.3.4 Subpohon atau Upapohon (*Subtree*)

Upapohon adalah bagian dari suatu pohon yang juga merupakan pohon. Contohnya, salah satu upapohon dari pohon pada gambar 3 adalah pohon yang dibentuk oleh simpul b, e, f, h, i, dan j.

2.3.5 Derajat (*Degree*)

Derajat dari suatu simpul adalah jumlah anak pada simpul tersebut. Contohnya, pada gambar 3, simpul a berderajat 3, sedangkan simpul d berderajat 1.

Derajat suatu pohon adalah derajat maksimum dari simpul-simpul pada pohon tersebut.

2.3.6 Daun (*Leaf*)

Daun dari suatu pohon adalah simpul yang memiliki derajat nol (tidak punya anak). Pada pohon berakar, daun terletak di bagian ujung bawah dari pohon. Sebagai contoh, daun pada pohon di gambar 3 adalah simpul h, i, j, l, dan m.

2.3.7 Simpul Dalam (*Internal Node*)

Simpul dalam adalah simpul yang memilki orang tua dan anak. Contohnya, pada gambar 3, simpul dalam adalah b, c, d, e, f, g, dan k.

2.3.8 Tingkat (*Level*)

Tingkat menunjukkan seberapa jauh suatu simpul dari suatu simpul yang menjadi akar (tidak mempunyai *parent*). Contoh, pada gambar 3, tingkat dari simpul a adalah 0, tingkat dari simpul b, c, dan d adalah 1, dan seterusnya.

2.3.9 Tinggi (*Height*) atau Kedalaman (*Depth*)

Tinggi atau kedalaman dari suatu pohon adalah tingkat maksimum dari pohon tersebut. Pohon pada gambar 3 memiliki kedalaman 4.

2.4 Tata Bahasa Bebas Konteks (*Context Free Language*)

Tata bahasa bebas konteks adalah suatu bentuk rekursif yang digunakan untuk mengenali bahasa bebas konteks (*Context Free Language*). Prinsip dasar untuk mengenali bahasa bebas konteks dengan tata bahasa bebas konteks adalah, kita mulai dengan suatu *string* yang sudah pasti merupakan anggota dari suatu bahasa bebas konteks. Kemudian, dengan aturan – aturan yang terdapat pada tata bahasa bebas konteks yang bersangkutan, kita dapat menurunkan string-string lain yang juga termasuk dalam bahasa yang bersangkutan.

Suatu tata bahasa bebas konteks dapat dikenali dari 4 identitas yang dimilikinya , yaitu variabel (V), terminal (T), simbol mulai (S), dan aturan produksi (P). V adalah variabel apa saja yang digunakan dalam pendefinisian tata bahasa bebas kontesk terkait. Terminal adalah alphabet apa saja yang diijinkan sebagai input dalam tata bahasa

bebas konteks tersebut. Simbol mulai (S) adalah salah satu anggota dari himpuanan variabel V yang merupakan langkah awal untuk melakukan penurunan string yang dikenali bahasa terkait. Aturan produksi (P) merupakan kumpulan aturan yang mendefinisikan bahasa apa saja yang diterima. P terdiri dari variabel dan terminal.

Contoh sederhana untuk menjelaskan cara kerja tata bahasa bebas konteks adalah tata bahasa yang mengenali string palindrom, yaitu string yang dibaca sama dari kiri ke kanan dan dari kanan ke kiri. Contohnya adalah string "malam" dan "12321". Untuk penyederhanaan, contoh yang dibahas di bagian ini adalah string palindrom yang hanya terdiri dari angka 1 dan 0.

Variabel (V) yang digunakan dalam tata bahasa bebas konteks ini hanya satu, yaitu S. Terminal terdiri dari 0 dan 1. Simbol mulai adalah S. Aturan produksi (P) adalah sebagai berikut.

S -> 0
S -> 1
S - > 0S0
S - > 1S1
S -> ε

Aturan yang pertama menjelaskan bahwa suatu string yang hanya terdiri dari satu string '0' adalah string palindrom. Demikian pula dengan aturan yang kedua, string yang hanya terdiri dari satu string '1' adalah string palindrom. Aturan pertama dan kedua adalah basis dari rekursif, sedangkan aturan ketiga dan keempat adalah rekurensnya. Aturan ketiga menjelaskan bahwa jika suatu string S adalah palindrome, maka string yang diperoleh dengan menambahkan string '0' pada awal dan akhir string S juga merupakan string palindrome. Sebagai contoh, dari aturan pertama, string '0' adalah palindrom. Maka, menurut aturan ketiga, string '000' juga merupakan string palindrom. Aturan keempat juga memiliki makna yang hampir sama, hanya saja string '0' diganti dengan string '1'. Aturan kelima menyatakan bahwa string kosong juga merupakan string palindrom. Hal ini supaya aturan produksi P juga dapat mengenali string palindrome yang jumlah karakternya genap. Sebagai contoh, karena string kosong adalah palindrom, maka menurut aturan ketiga, string "00" juga merupakan string palindrom.

Ada dua cara umum untuk menurunkan suatu string yang termasuk ke dalam bahasa tertentu dari aturan produksi. Kedua cara ini pada dasarnya sama, hanya saja urutan pengerjaannya yang berbeda.

2.4.1 *Leftmost Derivation* (Penurunan Ujung Kiri)

Leftmost Derivation adalah penurunan suatu string dengan aturan selalu melakukan subtitusi pada variabel yang paling kiri. Penurunan dimulai dari variabel S yang merupakan simbol mulai, kemudian memilih aturan produksi yang mengandung variabel S dan melakukan subtitusi terhadap variabel S tersebut. Hasil subtitusinya dapat berupa semuanya terminal, gabungan terminal dan variabel, serta seluruhnya variabel. Jika hasilnya seluruhnya terminal (string) tidak ada subtitusi lebih lanjut yang dapat dilakukan dan langkah pengerjaan selesai. Jika masih terdapat variabel, maka lakukan subtitusi pada variabel yang paling kiri. Contoh dari *leftmost derivation* adalah sebagai berikut.

S -> E
E -> E+E
E -> E*E
E -> (E)
E -> I
I -> 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0
I -> I1 | I2 | I3 | I4 | I5 | I6 | I7 | I8 | I9 | I0

Misalnya diberikan aturan produksi seperti di atas. Bahasa yang direpresentasikan oleh aturan produksi tersebut adalah bahasa yang menerima ekspresi matematika yang hanya terdiri dari operasi tambah dan kali. Simbol '|' berarti "atau". Contohnya, pada aturan produksi yang keenam, maksudnya adalah variabel I dapat disubtitusi dengan 1 atau 2 atau 3 dan seterusnya.

Misalnya kita ingin menurunkan string (2 + 3)*7 dari aturan produksi tersebut. Dengan menggunakan *leftmost derivation*, penurunannya adalah sebagai berikut.

S => E => E*E => (E)*E =>(E+E)*E => (I+E)*E => (2+E)*E => (2+I)*E => (2+3)*E => (2+3)*I => (2+3)*7

Pada contoh di atas, proses subtitusi selalu konsisten dilakukan pada variabel yang paling kiri.

2.4.2 *Rightmost Derivation* (Penurunan Ujung Kanan)

Pada dasarnya, *Rightmost Derivation* sama dengan *Leftmost Derivation*. Perbedaannya, *rightmost derivation* konsisten melakukan subtitusi pada variabel yang paling kanan. Contoh *rightmost derivation* pada aturan produksi ekspresi matematika pada contoh sebelumnya adalah sebagai berikut.

S => E*E => E*I => E*7 => (E)*7 => (E+E)*7 => (E+I)*7 => (E+3)*7 => (2+3)*7

Proses subtitusi selalu dilakukan pada variabel yang paling kanan. Perhatikan bahwa terdapat perbedaan dalam langkah-langkah penurunan antara *leftmost derivation* dan *rightmost derivation*.

2.5 *Parse Tree* (Pohon Pengurai)

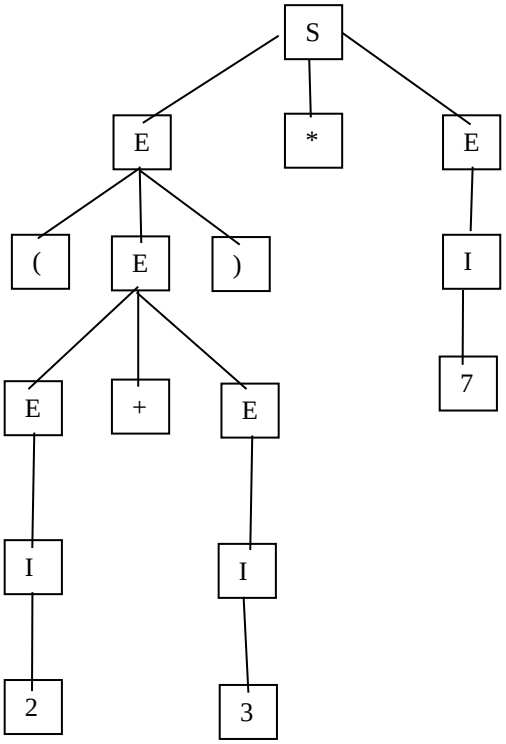
Parse Tree adalah representasi penurunan suatu string dari tata bahasa tertentu dalam struktur pohon. *Parse tree* banyak digunakan dalam teknologi *compiler* untuk mengenali *source code* program.

Jika suatu tata bahasa G memiliki impunan variabel V, terminal T, aturan produksi P, dan simbol mulai S, maka *parse tree* -nya didefinisikan sebagai berikut.

- Setiap simpul dalam adalah variabel anggota V.
- Setiap daun adalah variabel dalam V, atau terminal dalam T, atau string kosong ε.
- Jika suatu simpul A dalam pohon memiliki anak-anak A_1, A_2, A_3, … A_k secara berurutan dari kiri

ke kanan, maka aturan produksi A -> $A_1A_2A_3…A_k$ terdapat di dalam P.

Berikut ini adalah contoh *parse tree* yang menunjukkan penurunan string (2+3)*7 dari tata bahasa ekspresi matematika.



Gambar 4 : *Parse Tree* yang menunjukan penurunan string (2+3)*7 dari tata bahasa yang mengenali ekspresi matematika.

Simbol-simbol pada daun dari pohon di atas jika digabungkan akan membentuk string (2+3)*7.

III. PEMBAHASAN

3.1 *Natural Language Processing* (Pemrosesan Bahasa Manusia)

Natural Language Processing (NLP) adalah suatu metode bagi kecerdasan buatan (AI) untuk berkomunikasi menggunakan bahasa natural (bahasa manusia). Contohnya adalah bahasa Inggris. Contoh aplikasi dari NLP adalah robot yang dapat melakukan instruksi yang dieberikan manusia secara lisan. Ada dua bentuk input dari NLP, yaitu input lisan (artinya AI juga harus dapat mengenali pelafalan kata manusia) dan input berupa teks (AI harus dapat mengenali huruf).

3.2 Komponen dari *Natural Language Processing*

Ada dua komponen dari NLP, yaitu sebagai berikut.

3.2.1 *Natural Language Understanding* (NLU)

NLU adalah tahapan dimana AI memahami bahasa manusia. Setelah menerima input bahasa manusia, AI harus dapat melakukan pemetaan dari komponen-komponen bahasa tersebut dan mengubahnya menjadi struktur yang mudah untuk diproses. Setelah itu, AI harus menganalisis aspek-aspek dari bahasa tersebut. Dapat dikatakan, NLU adalah kemampuan AI untuk mendengarkan dan membaca bahasa manusia. NLU adalah tahapan yang paling sulit. Berikut ini adalah kesulitan dan kendala pada tahapan NLU.

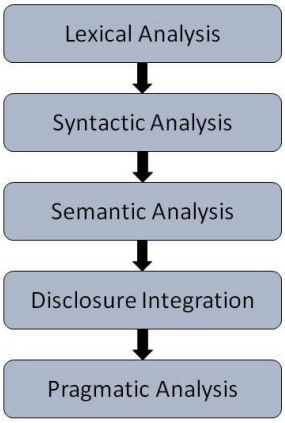
- *Lexical Ambiguity*, yaitu sifat ambigu dari suatu kata. Suatu kata dapat memiliki beberapa makna. Sebagai contoh, kata "*board*" dalam bahasa Inggris dapat memiliki dua arti, sebagai kata benda yang artinya papan dan sebagai kata kerja yang artinya naik.
- *Syntax Level Ambiguity*, yaitu sifat ambigu dari suatu kalimat. Kalimat majemuk yang jika diuraikan menjadi kalimat-kalimat tunggal memiliki lebih dari satu penguraian. Contohnya, kalimat bahasa Inggris "*He lifted the beetle with red cap*." memiliki arti dia mengangkat kumbang dengan topi merah. Apakah yang dimaksud dia menggunakan topi merah untuk mengangkat kumbang atau dia mengangkat kumbang yang bertopi merah.
- *Referential Abiguity*, yaitu sifat ambigu yang timbul dari penggunaan kata ganti orang. Contohnya, "Kemarin adik saya bertemu dengan Tuti. Dia bilang bahwa dia sudah menyelesaikan tugasnya." Pertanyaannya adalah siapakah yang sudah menyelesaikan tugasnya, adik saya atau Tuti.

3.2.2 *Natural Language Generating* (NLG)

NLG adalah proses menghasilkan suatu kalimat bermakna dalam bahasa manusia dari struktur internal yang sudah ada. Pada dasarnya, NLG adalah kemampuan AI untuk dapat berbicara dalam bahasa manusia.

3.3 Tahapan dalam NLP

Ada lima tapahan dalam pemrosesan bahasa manusia oleh komputer. Kelima tahapan tersebut adalah sebagai berikut.



Gambar 5 : Tahapan dalam NLP

3.3.1 *Lexical Analysis*

Pada tahap ini, dilakukan identifikasi dan analisis terhadap struktur kata. Pada tahap ini, teks dibagi menjadi paragraph-paragraf, kalimat, dan kata.

3.3.2 *Syntactic Analysis*

Pada tahap ini, dilakukan analisis terhadap kumpulan kata dalam kalimat untuk mencari tahu bagaimana kata yang satu dengan yang lain terhubung. Intinya, tahap ini melakukan validasi apakah input yang diberikan memenuhi aturan tata bahasa. Kata-kata dimasukkan ke dalam suatu struktur, dalam makalah ini struktur tersebut adalah pohon. Contoh dari kalimat yang ditolak pada tahapan ini adalah "Kue memakan saya".

3.3.3 *Semantic Analysis*

Pada tahap ini, dilakukan pemeriksaan terhadap arti dari suatu kalimat dengan proses pemetaan. Contoh kalimat yang ditolak adalah "Es teh panas".

3.3.4 *Discourse Integration*

Pada tahap ini, pengartian suatu kalimat dilakukan berdasarkan pada arti dari kalimat sebelumnya, juga dari kalimat sesudahnya. Intinya, dilakukan pengartian kalimat-kalimat sebagai satu kesatuan. Contohnya, kalimat "Kemudian dia membelinya". Arti dari kalimat ini bergantung dari kalimat sebelumnya. Untuk mengetahui apa yand dia beli, diperlukan data dari kalimat sebelumnya.

3.3.5 *Pragmatic Analysis*

Pada tahapan ini, input yang diberikan diinterpretasikan kembali untuk memperoleh makna sesungguhnya dari kalimat tersebut. Hal ini memerlukan pengetahuan seperti halnya yang dimiliki manusia pada umumnya. Sebagai contoh, kalimat "Tutup pintunya ?" memiliki makna permintaan dan bukan perintah.

3.4 Implementasi dari *Syntactic Analysis*

Ada berbagai algoritma untuk diimplementasikan dalam *Syntactic Analysis*. Metode yang paling mudah dan yang akan dibahas di sini adalah *Context Free Grammar* dan *Parse Tree*.

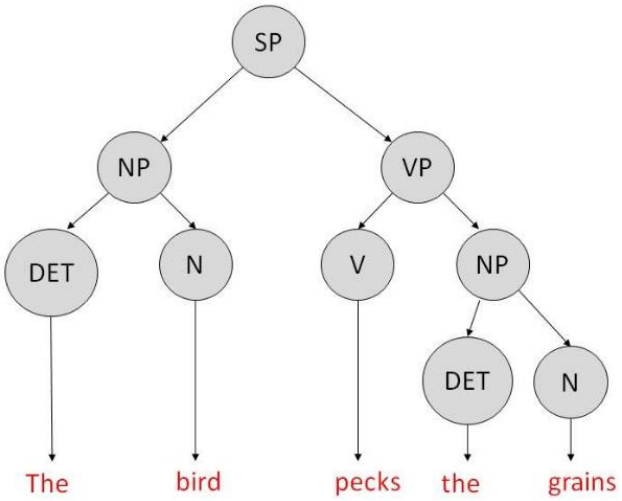
Berikut ini adalah contoh implementasi *Context Free Grammar* untuk mengenali suatu bahasa Inggris sederhana.

DET -> a | the
ADJ -> beauriful | ugly
N -> bird | birds | grain | grains
V -> peck | pecks | pecking
S -> NP VP
NP -> DET N | DET ADJ N
VP -> V NP

Dari aturan produksi di atas, misal kita ingin menurunkan string "The bird pecks the grains". Dengan menggunakan *leftmost derivation*, penurunannya adalah sebagai berikut.

S => NP VP => DET N VP => The N VP => The bird VP => The bird V NP => The bird pecks NP => The bird pecks DET N => The bird pecks the grains

Berikut adalah *parses tree* yang merepresentasikan penurunan dari string tersebut.



Gambar 6 : *Parse Tree* yang menunjukkan penurunan string "The bird pecks the grains".

Implementasi pada contoh hanya terbatas pada beberapa kata saja dan struktur kalimat yang sederhana. Untuk pengaplikasian yang sesungguhnya, jumlah aturan produksi yang ada pada tata bahasa bebas konteks jauh lebih banyak, demikian pula dengan jumlah terminal dan variabelnya. *Parses tree* yang dibuat juga akan jauh lebih kompleks.

IV. KESIMPULAN

Teknologi untuk memproses bahasa manusia memegang peranan penting dalam interaksi antara manusia dengan mesin di masa mendatang. *Natural Language Processing* memiliki dua komponen utama, yaitu *Natural Language Understanding* dan *Natural Language Generating*.

Salah satu tahapan *Natural Language Understanding* yang menggunakan pengaplikasian pohon adalah *Syntactic Analysis*. Dengan menggunakan *Context Free Language* dan *Parses Tree*, komputer dapat mengenali suatu bahasa manusia yang sesuai dengan kaidah tata bahasa.

V. UCAPAN TERIMA KASIH

Pada kesempatan ini, Penulis mengucapkan terima kasih kepada Tuhan YME yang telah memberi kesempatan dan kemampuan kepada Penulis untuk menyelesaikan pembuatan makalah ini. Penulis juga ingin berterima kasih kepada kedua orangtua Penulis yang telah memberikan dukungan moral dan finansial serta teman-teman teknik informatika angkatan 2015.

Selain itu, Penulis juga mengucapkan terima kasih sebesar-besarnya untuk Bapak Dr.Ir. Rinaldi Munir, MT. selaku dosen pengajar mata kuliah Matematika Diskrit yang telah mengajar dan membimbing Penulis semester ini. Terakhir, Penulis ingin berterima kasih kepada siapapun yang telah membantu dalam penulisan makalah ini.

DAFTAR PUSTAKA

[1] http://beebom.com/examples-of-artificial-intelligence/, diakses pada 3 Desember 2016.

[2] Crevier, Daniel (1993), *AI: The Tumultuous Search for Artificial Intelligence*, New York, NY: BasicBooks

[3] http://rizaxxi.blogspot.co.id/2015/07/pohon-pada-matematika-diskrit.html, diakses pada 8 Desember 2016.

[4] https://www.tutorialspoint.com/discrete_mathematics/introduction_to_trees.htm, diakses 9 Desember 2016.

[5] https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligence_natural_language_processing.htm, diakses 9 Desember 2016.

[6] Munir, Rinaldi, 2006. *Matematika Diskrit*, Bandung: Penerbit Informatika Bandung.

[7] http://www.mind.ilstu.edu/curriculum/protothinker/natural_language_processing.php, diakses 9 Desember 2016.

[8] Hopcroft, John, 2007. *IntroductionAutomata Theory,Languages, and Computation*, Boston : Pearson Education, Inc.

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, 8 Desember 2016

Albertus Djauhari Djohan/13515054