Aplikasi Graf pada Teori Automata

Devina Ekawati - 13513088
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
devina_ekawati@s.itb.ac.id

***Abstrak*—Graf adalah struktur diskrit yang terdiri dari simpul (*vertex*) yang merepresentsikan objek tertentu dan sisi (*edge*) yang menghubungkan sepasang simpul. Teori graf banyak digunakan dalam berbagai bidang kehidupan, misalnya penyusunan jadwal kuliah, menentukan jalur terpendek ke tempat tertentu, dan lain- lain. Teori graf juga digunakan pada berbagai bidang ilmu, seperti ilmu Kimia, ilmu Biologi, ilmu Geografi, Teori Bahasa Formal, dan Teori Otomata. Dalam makalah ini akan dibahas beberapa aplikasi graf pada Teori Otomata, terutama dalam *Deterministic Finite Automata* (DFA) dan *Pushdown Automata* (PDA).**

***Keywords*—aplikasi graf, DFA, PDA, teori otomata, teori graf**

I. PENDAHULUAN

Teori Otomata merupakan ilmu dalam bidang informatika yang mempelajari mesin- mesin abstrak. Menurut Inggriani Liem, mesin abstrak adalah mesin yang dianggap ada dan diasumsikan mampu melakukan suatu mekanisme yang telah didefinisikan untuk mesin tersebut. Kerja yang dilakukan oleh mesin abstrak seringkali tidak dirasakan oleh pengguna dan pengguna biasanya hanya melihat hasil keluarannya saja. Mesin abstrak terdiri dari mesin karakter dan mesin integer.

Mesin abstrak memiliki banyak manfaat. Mesin abstrak diimplementasikan pada *vending machines* (mesin jaja), pembacaan struktur DNA, mencari angka dalam sebuah deret angka, dan lain- lain. Dalam bidang informatika, mesin abstrak sering digunakan untuk menggambarkan komputasi yang dilakukan oleh komputer. Tanpa adanya mesin abstrak, komputer hanyalah sebuah mesin yang mampu melakukan operasi dasar/ primitif seperti yang dilakukan oleh kalkulator. Agar komputer mampu menyelesaikan masalah yang lebih kompleks, misalnya menghitung jumlah kata, perlu didefinisikan mesin abstrak.

Terdapat empat jenis otomata, yaitu *Finite- state machine*, *pushdown otomata*, *linear- bounded automata*, dan mesin turing. Teori otomata tersebut seringkali direpresntasikan dalam bentuk graf untuk menggambarkan proses- proses yang terjadi. Dalam graf tersebut, proses direpresentasikan dengan simpul, sedangkan sisi merepresentasikan transisi dari suatu status ke status lainya. Tujuan direpresentasikannya proses- proses tersebut dalam suatu graf tidak lain adalah untuk memudahkan manusia untuk memahami dan mempelajari proses- proses yang digambarkan dalam suatu otomata.

II. DASAR TEORI

Graf adalah struktur diskrit yang terdiri dari simpul (*vertex*) yang merepresentasikan objek tertentu dan sisi (*edge*) yang menghubungkan sepasang simpul. Secara matematis, graf G didefinisikan sebagai G = (V, E), yang dalam hal ini V merepresentasikan himpunan tidak kosong dari simpul (*vertex*) dan E merepresentasikan himpunan sisi (*edge*) yang menghubungkan dua buah simpul. Himpunan simpul V dalam graf G boleh tak berhingga. Graf yang memiliki himpunan simpul yang tak berhingga (*infinite*) disebut graf tak berhingga (*infinite graph*). Sedangkan graf yang himpunan simpulnya berhingga (*finite*) disebut graf berhingga (*finite graph*).

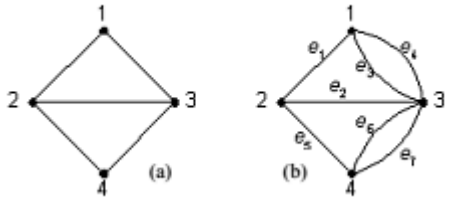
Berdasarkan definisi di atas, suatu graf harus memiliki minimal sebuah simpul. Namun sebuah graf boleh tidak memiliki sisi. Graf yang hanya memiliki satu atau beberapa simpul tetapi tidak mempunyai sisi disebut graf kosong. Sisi yang menghubungkan dua buah simpul yang sama disebut sisi- ganda. Sedangkan sisi yang berawal dan berakhir pada simpul yang sama disebut gelang atau kalang (*loop*).

2.1 Jenis- jenis Graf

Berdasarkan ada tidaknya gelang atau sisi ganda pada suatu graf, graf digolongkan menjadi dua jenis, yaitu:

1. Graf sederhana (*simple graph*)
   Graf sederhana adalah graf yang tidak mengandung gelang maupun sisi- ganda.

2. Graf tak sederhana (*unsimple graph)*
   Graf tak sederhana adalah graf yang mengandung sisi ganda atau gelang.



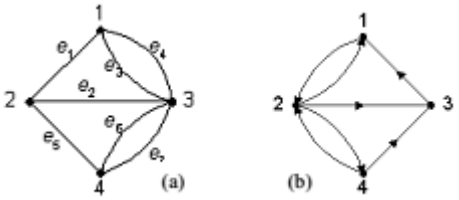
Gambar 2.1.1 (a) Graf sederhana, (b) Graf tak-sederhana

[3] source: Munir, Rinaldi. 2008. "Diktat Kuliah IF 2091 Struktur Diskrit". Bandung:Program Studi Teknik Informatika STEI ITB.

Berdasarkan orientasi arah pada sisi, graf dibedakan atas dua jenis:

1. Graf tak- berarah (*undirected graph*)
   Graf tak- berarah adalah graf yang sisinya tidak mempunyai orientasi.

2. Graf berarah (*directed graph* atau *digraph*)
   Graf berarah adalah graf yang sisinya memiliki orientasi arah.



Gambar 2.1.2 (a) Graf tak- berarah, (b) Graf berarah

[3] source: Munir, Rinaldi. 2008. "Diktat Kuliah IF 2091 Struktur Diskrit". Bandung:Program Studi Teknik Informatika STEI ITB.

2.2 Terminologi Graf

1. Ketetanggaan (*Adjacent*)
   Dua buah simpul dikatakan bertetangga apabila keduanya terhubung langsung.

2. Bersisian (*Incidency*)
   Sebuah simpul v dan sebuah sisi e dikatakan bersisian apabila simpul v dihubungkan ke simpul lain melalui sisi e.

3. Simpul Terpencil (*Isolated Vertex*)
   Sebuah simpul dikatakan terpencil (*isolated*) jika simpul tersebut tidak mempunyai sisi yang bersisian dengannya.

4. Graf Kosong (*null graph* atau *empty graph*)
   Graf kosong adalah graf yang himpunan sisinya merupakan himpunan kosong.

5. Derajat (*Degree*)
   Derajat sebuah simpul adalah jumlah sisi yang bersisian dengan simpul tersebut.

6. Lintasan (*Path*)
   Lintasan merupakan barisan berselang- seling simpul dan sisi- sisi yang berbentuk v_o, e_1, v_1, e_2, v_2, ... , v_{n-1}, e_n, v_n sedemikian sehingga $e_1 = (v_o, v1)$, $e_2 = (v_1, v_2)$, ... , $e_n = (v_{n-1}, v_n)$ adalah sisi-sisi dari graf.

7. Siklus (*Cycle*) atau Sirkuit (*Circuit*)
   Sirkuit atau siklus adalah lintasan yang berawal dan berakhir pada simpul yang sama.

8. Terhubung (*Connected*)
   Dua buah simpul v_1 dan simpul v_2 dikatakan terhubung apabila terdapat lintasan dari v_1 ke v_2.

9. Upagraf (*Subgraph*) dan Komplemen Upagraf
   $G_1 = (V_1, E_1)$ merupakan upagraf dari $G = (V, E)$ jika $V_1 \subseteq V$ dan $E_1 \subseteq E$. Sedangkan komplemen dari upagraf G_1 merupakan graf $G_2 = (V_2, E_2)$ sedemikian sehingga $E_2 = E - E_1$ dan V_2 adalah himpunan simpul- simpul yang bersisian dengan anggota E_2.

10. Upagraf Rentang (*Spanning Subgraph*)
    $G_1 = (V_1, E_1)$ merupakan upagraf rentang dari $G = (V, E)$ jika $V_1 = V$, yang dalam hal ini G_1 mengandung semua simpul dari G.

11. *Cut- Set*
    Cut- set dari graf terhubung adalah himpunan sisi yang jika dibuang dari graf terebut menyebabkan graf menjadi tidak terhubung.

12. Graf Berbobot (*Weighted Graph*)
    Graf berbobot adalah graf yang setiap sisinya diberi sebuah harga (bobot).

2.3 Graf Khusus

1. Graf Lengkap (*Complete Graph*)
   Graf lengkap adalah graf sederhana yang setiap simpulnya mempunyai sisi ke semua simpul lainnya.

2. Graf Lingkaran
   Graf lingkaran adalah graf sederahana yang setiap simpulnya berderajat dua.

3. Graf Teratur (*Regular Graph*)
   Graf teratur adalah graf yang setiap simpulnya memiliki derajat yang sama.

4. Graf *Bipartite* (*Bipartite Graph*)
   Graf *bipartite* adalah graf yang himpunan

simpulnya dapat dipisahkan menjadi dua himpunan bagian V_1 dan V_2 sehingga setiap sisi pada graf tersebut menghubungkan sebuah simpul di V_1 ke sebuah simpul di V_2.

III. TEORI OTOMATA

Teori otomata adalah ilmu yang mempelajari mesin abstrak dan segala masalah yang dapat diselesaikan menggunakan mesin abstrak. Kata otomata (*automata*) berasal dari bahasa Yunani yang berarti bekerja sendiri (*self- acting*). Melalui teori otomata, proses yang dilakukan oleh komputer seperti melakukan perhitungan dan menyelesaikan masalah dapat dipahami oleh manusia.

Teori otomata pertama kali dikembangkan pada abad ke- 20. Pada abad tersebut, matematikawan mulai membuat berbagai macam cara untuk menciptakan mesin yang mampu melakukan pekerjaan manusia serta menyelesaikan perhitungan secara cepat dan tepat. Awalnya, para matematikawan hanya mampu menciptakan mesin abstrak dengan komputasi yang terbatas. Namun, pada tahun 1930-an, Alan Turing berhasil mendefinisikan sebuah mesin yang mampu memodelkan komputasi yang tidak terbatas. Mesin turing ini yang sekarang digunakan oleh komputer.

3.1 Jenis- jenis Otomata

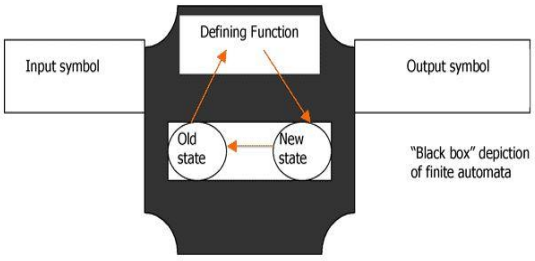
Terdapat empat jenis otomata, yaitu:

1. *Finite- state machine*

   Otomata yang himpunan status- statusnya berhingga disebut *finite- state machine* (FSM). FSM adalah mesin abstrak yang terdiri dari himpunan status (Q), himpunan input (I), himpunan output (Z), dan fungsi transisi. Fungsi transisi adalah fungsi yang menyebabkan perpindahan/ transisi dari satu status ke status lainnya.

   Secara formal, *finite- state machine* didefinisikan dalam 5- *tuple* (Q, Σ, δ, q_0, F) sedemikian sehingga:

   - Q = himpunan status yang berhingga
   - Σ = himpunan alfabet yang diinput
   - δ = fungsi transisi
   - q_0 = status awal
   - F = himpunan satus yang diterima (*accepted states*)



Gambar 3.1.1 Diagram FSM

source: http://cs.stanford.edu/people/eroberts/courses/soco/projects/2004-05/automata-theory/basics.html

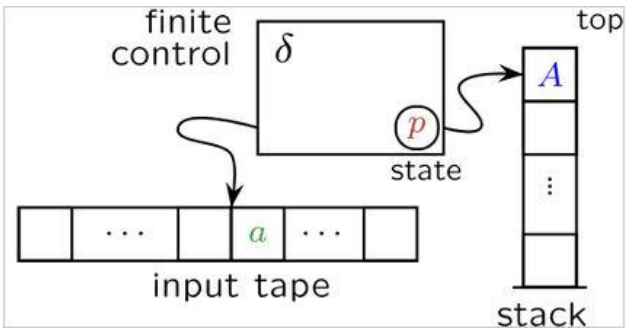
FSM dibagi menjadi dua, yaitu *Deterministic Finite Automata* (DFA) dan *Non- Deterministic Automata* (NFA). Menurut Mogensen, perbedaan DFA dan NFA adalah NFA memungkinkan terjadinya perpindahan/ transisi dari suatu status ke banyak status akibat satu simbol. Sementara pada DFA, satu simbol hanya menyebabkan satu transisi. Selain itu, pada NFA dimungkinkan terjadinya transisi spontan atau transisi ε (epsilon) yang seringkali disebut sebagai NFA- ε.

2. *Pushdown automata*

   Pushdown automata (PDA) adalah jenis otomata yang menggunakan *stack* atau tumpukan untuk merepresentasikan tata bahasa bebas konteks (*context- free grammar*). Yang dimaksud dengan tata bahasa bebas konteks adalah tata bahasa yang tidak memiliki batasan terhadap hasil produksinya. Sama halnya dengan FSM, PDA terdiri dari *Deterministic Pushdwon Automata* dan *Non- Deterministic Pushdown Automata*. *Deterministic Pushdown Automata* hanya dapat mengenali *deterministic context- free grammar*, sementara *Non- Deterministic Pushdown Automata* dapat mengenali seluruh *context- free grammar*.

   Secara formal. *Pushdown automata* didefinisikan dalam 7- *tuple* (Q, Σ, Γ, δ, q_0, Z_0, F) sedemikian sehingga:

   - Q = himpunan status yang berhingga
   - Σ = himpunan alfabet yang diinput
   - Γ = alfabet yang terdapat dalam stack
   - δ = fungsi transisi
   - q_0 = status awal
   - Z_0 = simbol awal
   - F = himpunan status akhir (*accepted states*)



Gambar 3.1.2 Diagram Pushdown Automata

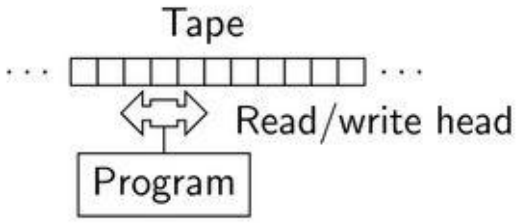
source: http://en.wikipedia.org/wiki/Pushdown_automaton

3. *Turing machine*

   Turing machine (TM) atau mesin turing adalah jenis otomata yang mampu membaca suatu pita karakter. Pembacaan pita karakter tersebut dapat dilakukan secara maju dan mundur. Selama proses pembacaan pita karakter, mesin turing akan melakukan perubahan terhadap pita karakter dan menentukan arah pembacaan pita karakter selanjutnya, apakah ke kiri atau ke kanan.

   Secara formal. Mesin turing didefinisikan dalam 7- *tuple* (Q, Σ, Γ, δ, q_0, B, F) sedemikian sehingga:

- Q = himpunan status yang berhingga
- Σ = alfabet yang diinput
- Γ = alfabet yang terdapat dalam pita karakter
- δ = fungsi transisi
- q_0 = status awal
- B = simbol *blank*/ kosong
- F = himpunan status akhir (*accept states*)



Gambar 3.1.2 Diagram mesin turing
source: http://www.decodedscience.com/examples-of-turing-machines-loops-halts-and-rewriting/10894

4. *Linear- bounded automata*

Linear- bounded automata (LBA) identik dengan mesin turing. Perbedaannya adalah selama perhitungan, pergerakan pita karakter tidak boleh melebihi area pada pita karakter.

IV. PENGGUNAAN GRAF PADA TEORI OTOMATA

Teori otomata menggambarkan proses- proses yang terjadi pada mesin abstrak. Representasi proses- proses tersebut dapat digambarkan dengan tabel dan graf. Representasi dengan tabel lebih sering diimplementasikan pada komputer, sementara representasi dengan graf digunakan untuk memudahkan manusia memahami proses apa saja yang terjadi pada otomata tersebut. Dalam bahasan ini, akan dipaparkan penggunaan graf pada DFA dan PDA.

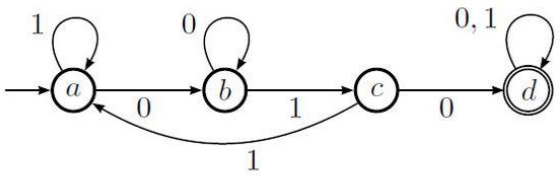
4.1 Penggunaan graf pada DFA

Jenis graf yang digunakan untuk merepresentasikan DFA adalah graf tak sederhana dan graf berarah. Hal ini berarti, graf tersebut dapat mengandung sisi ganda serta gelang. Selain itu, setiap sisi pada graf memiliki arah tertentu.

Graf memiliki himpunan simpul dan sisi. Dalam DFA, himpunan simpul tersebut merepresentasikan status-status, sedangkan himpunan sisi merepresentasikan perpindahan dari satu status ke status yang lain. Setiap sisi pada graf tersebut memiliki label, yang dalam hal ini label tersebut melambangkan alfabet pada DFA tersebut.

Graf yang digunakan untuk merepresentasikan DFA memiliki beberapa modifikasi yaitu:

- Terdapat penunjuk/ panah dengan label *start* yang menandakan status awal
- Simpul yang merepresentasikan status akhir memiliki dua lingkaran yang konsentris.



Gambar 4.1.1 Implementasi graf pada DFA
source: http://www.itsalif.info/content/dfa-nfa-regular-expression-without-using-gnfa

Gambar 4.1.1 merupakan DFA yang menerima *string* yang mengandung 010. Definisi DFA tersebut secara formal adalah

- Q = {a, b, c, d}
- Σ = {0, 1}
- q_0 = a
- F = d
- Fungsi transisi

δ	0	1
→a	B	a
b	B	c
c	D	a
*d	D	d

Simpul/ status a merupakan status awal karena terdapat penunuk/ panah dengan label *start* yang menunjuk status a. Selanjutnya terdapat dua sisi yang bersisian dengan simpul a, yaitu satu sisi yang kembali lagi ke simpul a dan satu sisi lain yang menuju status b. Bobot pada sisi yang kembali lagi ke simpul a adalah 1. Hal ini berarti jika komponen alfabet yang sedang diproses adalah 1, maka tidak terjadi perubahan status. Jika komponen alfabet yang sedang diproses adalah 0, maka akan terjadi perubahan status dari status a ke status b. Demikian seterusnya untuk status yang lain. Simpul/ status d merupakan status akhir karena simpul d digambarkan dengan dua lingkaran yang konsentris.

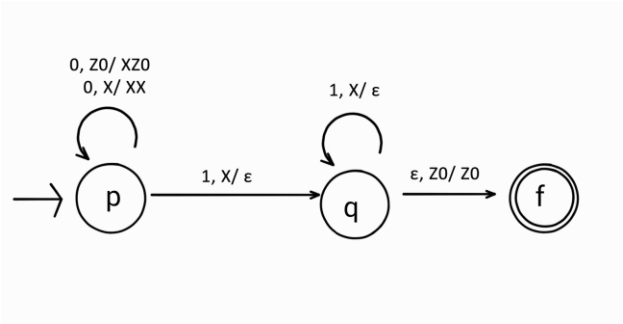
Misalnya DFA tersebut akan memroses *string* 0100. Pertama- tama DFA akan membaca karakter pertama, yaitu 0. Berdasarkan fungsi transisi pada status a, 0 akan menyebabkan perpindahan status dari a ke b. Selanjutnya DFA akan membaca karakter ke- dua, yaitu 1. Berdasarkan fungsi transisi pada status b, 1 menyebabkan perpindahan status dari b ke c. Setelah itu, DFA akan membaca karakter ke- tiga, yaitu 0. Berdasarkan fungsi transisi pada status c, 0 menyebabkan perpindahan status dari c ke d. Selanjutnya, DFA akan membaca karakter ke-empat, yaitu 0. 0 tidak menyebabkan perubahan status sehingga status sekarang tetap d. Karena d merupakan status akhir, *string* tersebut diterima.

4.2 Penggunaan graf pada PDA

Graf yang digunakan pada PDA hampir sama dengan graf yang digunakan pada DFA. Perbedaannya hanya terdapat pada label yang dimiliki oleh setiap sisi pada graf.

Label yang dimiliki sisi tersebut terdiri dari tiga komponen dengan format penulisan a, X/ α sedemikian sehingga:

- a = alfabet yang sedang diproses
- X = elemen puncak dari *stack*
- α = elemen yang akan di-*push* ke *stack*

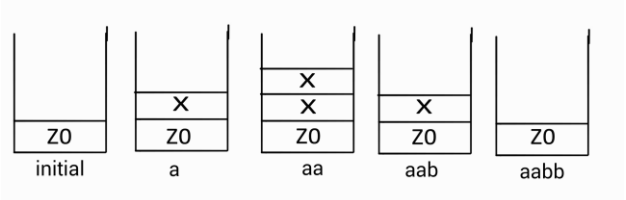


Gambar 4.2.1 Implementasi graf pada PDA

Gambar 4.1.2 merupakan PDA yang menerima *language* L sedemikian sehingga L = { w = 0^n1^n, n ≥ 0}. Definisi PDA tersebut secara formal adalah:

- Q = {q, p, f}
- Σ = {0, 1}
- Γ = {X, Z_0}
- δ
- q_0
- Z_0
- F = f

Misalkan PDA tersebut akan memroses *string* aabb. Pertama- tama PDA akan membaca karakter pertama, yaitu a. Alfabet a akan tidak akan menyebabkan perubahan status. Karena elemen *top* dari *stack* adalah Z_0, simbol X akan di-*push* ke dalam *stack* sehingga simbol di dalam stack adalah X dan Z_0. Selanjutnya, PDA akan membaca karakter ke- dua, yaitu a. Alfabet a tidak akan menyebabkan perubahan status. Karena elemen *top* sekarang dari *stack* adalah X, simbol X akan di-*push* ke dalam *stack* sehingga simbol di dalam stack adalah X, X, dan Z_0. Setelah itu, PDA akan membaca karakter ke- tiga, yaitu b sehingga terjadi perubahan status dari q ke p. Karena elemen *top* dari *stack* adalah X, X akan di-*pop* dari *stack* dan isi *stack* berubah menjadi X dan Z_0. Selanjutnya, PDA akan membaca karakter ke- empat, yaitu b. Alfabet b tidak mengakibatkan perubahan status. Karena elemen *top* dari *stack* adalah X, X akan di-*pop* dari *stack* dan isi *stack* berubah menjadi Z_0. Karena *string* telah selesai dibaca dan elemen *top* dari *stack* adalah Z_0, terjadi perubahan status dari p ke f. f merupakan status akhir sehingga *string* diterima.



Gambar 4.2.2 Isi stack selama proses pembacaan string aabb

V. KESIMPULAN

Graf memiliki banyak manfaat bagi kehidupan manusia. Oleh karena itu, teori graf banyak digunakan dalam berbagai cabang ilmu lain, salah satunya adalah Teori Otomata. Dengan mengaplikasikan graf dalam Teori Otomata, terutama dalam DFA dan PDA, Teori Otomata menjadi lebih mudah dipelajari dan dipahami oleh manusia. Dengan demikian, Teori Otomata diharapkan menjadi lebih bermanfaat bagi kehidupan manusia, terutama dalam bidang komputasi yang dilakukan oleh komputer sehingga pekerjaan yang dilakukan komputer semakin efisien dan efektif.

VI. UCAPAN TERIMA KASIH

Pertama- tama penulis ingin mengucapkan rasa syukur kepada Tuhan Yang Maha Esa karena oleh rahmat- Nya penulis dapat menyelesaikan makalah ini. Penulis juga ingin berterima kasih kepada Dr. Ir. Rinaldi Munir yang telah mengajarkan Matematika Diskrit, termasuk teori graf sehingga penulis mampu untuk membuat makalah ini. Selain itu, penulis ingin mengucapkan terima kasih kepada orang tua dan rekan- rekan yang memberikan semangat dan dorongan untuk terus belajar.

REFERENSI

[1] Hopcroft, John & Ullman, Jeffry. 2006. "Introduction to Automata Theory, Languages, and Computation". USA:Addison-Wesley.
[2] K. H. Rosen. 2007. "Discrete Mathematics and its Applications". New York:McGraw-Hill.
[3] Munir, Rinaldi. 2008. "Diktat Kuliah IF 2091 Struktur Diskrit". Bandung:Program Studi Teknik Informatika STEI ITB.
[4] http://cs.stanford.edu/people/eroberts/courses/soco/projects/2004-05/automata-theory/basics.html, 8 Desember 2014
[5] thesis.binus.ac.id/doc/Bab2/2011-2-00004-MTIF%20Bab2001.pdf, diakses 9 Desember 2014
[6] http://www.cs.wcupa.edu/rkline/fcs/pdas.html, diakses 10 Desember 2014

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, 4 Desember 2014

Devina Ekawati (13513088)