MODUL 11: PUSHDOWN AUTOMATON

Pengantar Pushdown Automaton

Dalam pembahasan bahasa regular telah diperkenalkan pula suatu mesin dengan jumlah status yang terbatas atau dikenal dengan nama mesin FA. Karena keterbatasan tempat penyimpanan informasi yang telah terjadi pada string masukan yang sedang dibaca maka bahasa yang dapat dikenalinya pun memiliki sifat yang amat terbatas yaitu bahasa regular.

Apabila mesin FA itu dilengkapi oleh suatu storage tak terhingga kapasitasnya serta dengan struktur stack (yaitu struktur dengan aturan FIFO) maka mesin ini mampu menyimpan sejumlah informasi yang disimpan pada saat pembacaan simbol-simbol di bagian depan string, dan dapat dipergunakan pada saat pembacaan simbol-simbol berikutnya dari string. Mekanisme stack yang dimiliki menyebabkan mesin dinamakan Pushdown Automaton (PDA).

Contoh
Untuk dapat mengenali bahasa $L = \{a^i b^j \mid i, j \geq 0\}$ maka suatu mesin FA cukup untuk dapat mengenali bahasa ini karena hal yang penting perlu diingat adalah apabila setelah simbol b di baca maka selanjutnya tidak boleh ada lagi simbol a. Keserhanaan pengenalannya terlihat dari jumlah status minimal yang diperlukan untuk mengenalinya.

Sementara, tidak ada suatu mesin FA pun yang mengenali $L = \{a^i b^j \mid i \geq j \geq 0\}$ karena diperlukan suatu cara yang tidak dimiliki oleh FA untuk menyimpan berapa banyak simbol a yang telah dibaca. Kita ingat bahasa ini merupakan suatu CFL. PDA sanggup mengenali bahasa tersebut karena dengan stack yang tak terbatas kapasitasnya itu setiap simbol a dapat disimpan dan kemudian "dipasangkan" pada setiap pembacaan b. Jika seluruh simbol dalam string telah dibaca dan tersisa simbol a dalam stack atau setidaknya stack kosong maka string diterima.

Jadi dengan stack ini kemampuan mesin menjadi bertambah, dalam arti kelas bahasa yang dapat dikenalinya meningkat.

Definisi: Suatu pushdown automaton (PDA) adalah 7-tuple $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$ di mana

- Q himpunan berhingga status-status
- Σ himpunan alfabet masukan
- Γ himpunan alfabet stack
- q_0 status inisial $q_0 \in Q$
- Z_0 simbol inisial dari stack
- A himpunan status menerima $A \subseteq Q$
- δ fungsi transisi dengan
  $\delta: Q \times (\Sigma \cup \{\Lambda\}) \rightarrow$ (subset berhingga dari $Q \times \Gamma^*$).

Notasi dan Terminologi
Konfigurasi. Setiap saat PDA dinyatakan oleh status, substring yang belum diproses, dan isi stack, yang mana ketiganya disebut konfigurasi mesin saat yang sedang berlangsung. Dari satu konfigurasi mesin bergerak (move) ke konfigurasi lain dengan mengaplikasikan suatu aturan produksi. Pada mesin M, move ini dinotasikan dengan simbol " $\vdash_M$" memperantarai konfigurasi sebelum dan sesudah move. Jika mesin yang mana secara implisit sudah diketahui maka pada identitas mesin tersebut bisa dihilangkan sehingga notasi menjadi " $\vdash$ ".

$$(p, x, a) \vdash_M (q, y, b)$$

Jika terjadi dalam beberapa (≥ 0) move maka penulisan dapat dipersingkat dengan simbol " $\vdash^*_M$" (atau " $\vdash^*$" jika sudah jelas pada mesin mana).

$(p, x, a) \vdash_M \ldots \vdash_M \ldots \vdash_M (q, y, b)$ dapat dituliskan dengan

$$(p, x, a) \vdash^*_M (q, y, b)$$

Dengan notasi ini maka bahwa suatu string diterima oleh suatu PDA $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$ apabila terdapat sejumlah (≥ 0) move sehingga $(q_0, x, Z_0) \vdash^*_M (q, \Lambda, a)$, dengan $q \in A$ dan $a \in \Gamma^*$.

Suatu bahasa L dikatakan "diterima" oleh mesin M apabila L adalah himpunan seluruh string yang dapat diterima oleh M (kita tulis bahwa $L = L(M)$).

Konfigurasi menerima (*accepting configuration*) menyatakan konfigurasi dengan statusnya adalah status menerima. Jadi disini tidak peduli bagaimana isi stack-nya; oleh sebab itu situasi ini dikatakan penerimaan dengan status final. Apabila isi stack dipentingkan sementara status tidak maka situasi ini disebut penerimaan dengan stack kosong. Kedua situasi ini ekivalen, dalam arti jika suatu string diterima dengan stack kosong maka juga akan ada mesin M yang dapat menerima string tersebut dengan status final.

PDA Deterministik

Definisi: Jika $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$ merupakan PDA, M dikatakan PDA deterministik (DPDA) jika tidak terdapat konfigurasi dimana M memiliki sejumlah pilihan kemana ia akan berpindah satu langkah.

Dpl, M deterministik jika ia memenuhi dua kondisi berikut.

1. untuk $q \in Q$, dan $a \in \Sigma \cup \{\Lambda\}$, serta $X \in \Gamma$, maka himpunan $\delta(q, a, X)$ paling banyak hanya berisi satu anggota himpunan.
2. Untuk $q \in Q$, dan $X \in \Gamma$, maka bila $\delta(q, \Lambda, X) \neq \varnothing$ maka $\delta(q, a, X) = \varnothing$ untuk setiap $a \in \Sigma$.

DCFL

Suatu bahasa L disebut CFL Deterministik (DCFL) bila terdapat suatu DPDA yang dapat menerima L.

Contoh

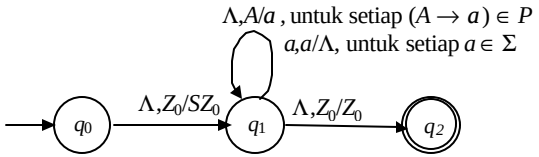
Saat pembahasan bahasa regular, setiap bahasa regular yang dikenali oleh NFA dapat pula dikenali oleh suatu FA, dan sebaliknya, setiap bahasa regular yang dikenali oleh FA dapat pula dikenali oleh suatu NFA. Sifat ini tidak berlaku pada CFL. Setiap bahasa DCFL dapat dibentuk oleh CFG nondeterministik tetapi sebaliknya tidak semua CFL dapat dibentuk oleh DCFG.

PDA dari suatu CFG

Ide dasar dari pembentukan PDA dari CFG ini adalah memanfaatkan stack yang dimiliki PDA sebagai ruang untuk mengsimulasikan penurunan-penurunan berdasar aturan-aturan produksi grammar. Jadi mesin PDA sebelum membaca string masukan, m

Jika $G = (V, \Sigma, S, P)$ suatu CFG, maka dapat dicari suatu PDA $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$ sehingga $L(M) = L(G)$. $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$ dengan

- $Q = \{q_0, q_1, q_2\}$
- $\Gamma = V \cup \Sigma \cup \{Z_0\}$, $Z_0 \notin (V \cup \Sigma)$
- $A = \{q_2\}$
- δ adalah fungsi transisi yang didefinisikan sebagai berikut.
  (1) $\delta(q_0, \Lambda, Z_0) = \{(q_1, SZ_0)\}$
  (2) u/ setiap $A \in V$, $\delta(q_1, \Lambda, A) = \{(q_1, a) \mid$ setiap $A \rightarrow a$ adalah aturan produksi dalam $G\}$
  (3) u/ setiap $a \in \Sigma$, $\delta(q_1, a, a) = \{(q_1, \Lambda)\}$
  (4) $\delta(q_1, \Lambda, Z_0) = \{(q_2, Z_0)\}$



Contoh

$L = \{x \in \{a, b\}^* \mid n_a(x) > n_b(x)\}$ adalah CFL yang berdasarkan CFG dengan aturan produksi sbb.

$S \rightarrow a \mid aS \mid bSS \mid SSb \mid SbS$

Berdasarkan bukti teorema di atas maka dapat dibuat suatu PDA $M = (\{q_0, q_1, q_2\}, \{a, b\}, \{S, a, b, Z_0\}, Z_0, \{q_2\}, \delta)$ dan fungsi δ dinyatakan dalam tabel berikut.

$\delta(q_0, \Lambda, Z_0) = \{(q_1, SZ_0)\}$
$\delta(q_0, \Lambda, S) = \{(q_1, a), (q_1, aS), (q_1, bSS), (q_1, SSb), (q_1, SbS)\}$
$\delta(q_1, a, a) = \{(q_1, \Lambda)\}$
$\delta(q_1, b, b) = \{(q_1, \Lambda)\}$
$\delta(q_1, \Lambda, Z_0) = \{(q_2, Z_0)\}$

Jika string $abbaaa \in L$ hendak dikenali oleh PDA ini maka akan terjadi konfigurasi-konfigurasi sbb.

$\delta(q_0, abbaaa, Z_0)$ $\delta(q_1, abbaaa, SZ_0)$ $\delta(q_1, abbaaa, SbSZ_0)$
$\delta(q_1, abbaaa, abSZ_0)$ $\delta(q_1, bbaaa, bSZ_0)$ $\delta(q_1, baaa, SZ_0)$
$\delta(q_1, baaa, bSSZ_0)$ $\delta(q_1, aaa, SSZ_0)$ $\delta(q_1, aaa, aSZ_0)$
$\delta(q_1, aa, SZ_0)$ $\delta(q_1, aa, aSZ_0)$ $\delta(q_1, a, SZ_0)$ $\delta(q_1, a, aZ_0)$
$\delta(q_1, \Lambda, Z_0)$ $\delta(q_2, \Lambda, Z_0)$

Untuk string ini PDA berhenti di status menerima q_2 dengan stack kosong (berisi Z_0).

CFG dari suatu PDA

Jika $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$ merupakan PDA yang menerima bahasa L dengan stack kosong, alias $L = L_e(M)$., maka akan terdapat suatu CFG $G = (V, \Sigma, S, P)$ dengan $L(G) = L$. Spesifikasi G diperoleh dari M sebagai berikut.

V adalah variabel-variabel yang dibentuk sebagai berikut. Pertama adalah suatu variabel awal S. kemudian variabel-variabel lainnya dibuat dengan penamaan tertentu yang merupakan bentukan dari semua kemungkinan tiga harga: p, A, q (untuk setiap p dan q $\in$ Q dan untuk setiap A $\in$ Γ) sebagai berikut.

$$V = \{S\} \cup \{[p, A, q] \mid A \in \Gamma, \text{ dan } p, q \in Q\}$$

P adalah produksi-produksi yang dibentuk tepatnya hanya menggunakan aturan pembentukan berikut ini.

1. Untuk setiap status $q \in Q$, dibuat produksi $S \rightarrow [q_0, Z_0, q]$.
2. Untuk setiap status-status $q, q_1 \in Q$, setiap simbol $a \in \Sigma \cup \{\Lambda\}$, dan setiap simbol stack $A \in \Gamma$, apabila dalam $\delta(q, a, A)$ terdapat (q_1, Λ), maka dibuat produksi-produksi berbentuk $[q, A, q_1] \rightarrow a$.
3. Untuksetiap status-status $q, q_1 \in Q$, setiap simbol $a \in \Sigma \cup \{\Lambda\}$, dan setiap simbol stack $A \in \Gamma$, apabila dalam $\delta(q, a, A)$ terdapat $(q_1, B_1B_2\ldots B_m)$ dengan $m \geq 1$, maka untuk setiap kemungkinan status-status $q_2, \ldots, q_{m+1} \in Q$ dibuat produksi-produksi berbentuk $[q, A, q_{m+1}] \rightarrow a[q_1, B_1, q_2][q_2, B_2, q_2] \ldots [q_m, B_m, q_{m+1}]$.

Contoh: Berikut ini tabel transisi dari suatu PDA untuk mengenali bahasa $L = \{xcx^r \mid x \in \{a, b\}^*\}$ (di samping tabel dituliskan nomor dari masing-masing aturan move untuk memudahkan penunjukkan nanti).

Status	Input	Simbol stack	move	
q_0	a	Z_0	(q_0, AZ_0)	(1)
q_0	b	Z_0	(q_0, BZ_0)	(2)
q_0	a	A	(q_0, AA)	(3)
q_0	b	A	(q_0, BA)	(4)
q_0	b	B	(q_0, BB)	(5)
q_0	a	B	(q_0, AB)	(6)
q_0	c	Z_0	(q_1, Z_0)	(7)
q_0	c	A	(q_1, A)	(8)
q_0	c	B	(q_1, B)	(9)
q_1	a	A	(q_1, Λ)	(10)
q_1	b	B	(q_1, Λ)	(11)
q_1	Λ	Z_0	(q_1, Λ)	(12)

Berdasarkan aturan pertama maka dibuat
$S \rightarrow [q_0, Z_0, q_0]$
$S \rightarrow [q_0, Z_0, q_1]$

Berdasarkan aturan kedua dan aturan move ke (10) dan ke (11) diperoleh

$[q_1, A, q_1] \rightarrow a$

$[q_1, B, q_1] \rightarrow b$

Berdasarkan aturan kedua dan aturan move ke (12) diperoleh

$[q_1, Z_0, q_1] \rightarrow \Lambda$

Berdasarkan aturan ke tiga dan aturan move ke (8) diperoleh

$[q_0, Z_0, q_0] \rightarrow c[q_1, Z_0, q_0]$

$[q_0, Z_0, q_1] \rightarrow c[q_1, Z_0, q_1]$

Berdasarkan aturan ke tiga dan aturan move ke (9) diperoleh

$[q_0, A, q_0] \rightarrow c[q_1, A, q_0]$

$[q_0, A, q_1] \rightarrow c[q_1, A, q_1]$

Berdasarkan aturan ke tiga dan aturan move ke (10) diperoleh

$[q_0, B, q_0] \rightarrow c[q_1, B, q_0]$

$[q_0, B, q_1] \rightarrow c[q_1, B, q_1]$

Berdasarkan aturan ke tiga dan aturan move ke (1) diperoleh

$[q_0, Z_0, q_0] \rightarrow a[q_0, A, q_0][q_0, Z_0, q_0]$

$[q_0, Z_0, q_0] \rightarrow a[q_0, A, q_1][q_1, Z_0, q_0]$

$[q_0, Z_0, q_1] \rightarrow a[q_0, A, q_0][q_0, Z_0, q_1]$

$[q_0, Z_0, q_1] \rightarrow a[q_0, A, q_1][q_1, Z_0, q_1]$

Berdasarkan aturan ke tiga dan aturan move ke (2) diperoleh

$[q_0, Z_0, q_0] \rightarrow b[q_0, B, q_0][q_0, Z_0, q_0]$

$[q_0, Z_0, q_0] \rightarrow b[q_0, B, q_1][q_1, Z_0, q_0]$

$[q_0, Z_0, q_1] \rightarrow b[q_0, B, q_0][q_0, Z_0, q_1]$

$[q_0, Z_0, q_1] \rightarrow b[q_0, B, q_1][q_1, Z_0, q_1]$

Berdasarkan aturan ke tiga dan aturan move ke (3) diperoleh

$[q_0, A, q_0] \rightarrow a[q_0, A, q_0][q_0, A, q_0]$

$[q_0, A, q_0] \rightarrow a[q_0, A, q_1][q_1, A, q_0]$

$[q_0, A, q_1] \rightarrow a[q_0, A, q_0][q_0, A, q_1]$

$[q_0, A, q_1] \rightarrow a[q_0, A, q_1][q_1, A, q_1]$

Berdasarkan aturan ke tiga dan aturan move ke (4) diperoleh

$[q_0, A, q_0] \rightarrow b[q_0, B, q_0][q_0, A, q_0]$

$[q_0, A, q_0] \rightarrow b[q_0, B, q_1][q_1, A, q_0]$

$[q_0, A, q_1] \rightarrow b[q_0, B, q_0][q_0, A, q_1]$

$[q_0, A, q_1] \rightarrow b[q_0, B, q_1][q_1, A, q_1]$

Berdasarkan aturan ke tiga dan aturan move ke (5) diperoleh

$[q_0, B, q_0] \rightarrow b[q_0, B, q_0][q_0, B, q_0]$

$[q_0, B, q_0] \rightarrow b[q_0, B, q_1][q_1, B, q_0]$

$[q_0, B, q_1] \rightarrow b[q_0, B, q_0][q_0, B, q_1]$

$[q_0, B, q_1] \rightarrow b[q_0, B, q_1][q_1, B, q_1]$

Berdasarkan aturan ke tiga dan aturan move ke (6) diperoleh

$[q_0, B, q_0] \rightarrow a[q_0, A, q_0][q_0, B, q_0]$

$[q_0, B, q_0] \rightarrow a[q_0, A, q_1][q_1, B, q_0]$

$[q_0, B, q_1] \rightarrow a[q_0, A, q_0][q_0, B, q_1]$

$[q_0, B, q_1] \rightarrow a[q_0, A, q_1][q_1, B, q_1]$

Menghasilkan total 35 aturan produksi. Aturan-aturan produksi di atas bisa ditulis ulang dengan penamaan variabel yang lebih sederhana, sbb.

$S \rightarrow M \mid N$

$M \rightarrow aRM \mid aTP \mid bUM \mid bVP \mid cP$

$N \rightarrow aRN \mid aTQ \mid bUN \mid bVQ \mid cQ$

$R \rightarrow aRR \mid aTW \mid bUR \mid bVW \mid cW$

$T \rightarrow aRT \mid aTX \mid bUT \mid bVX \mid cX$

$U \rightarrow aRU \mid aTY \mid bUU \mid bVY \mid cY$

$V \rightarrow aRV \mid aTZ \mid bUV \mid bVZ \mid cZ$

$Q \rightarrow \Lambda$

$X \rightarrow a$

$Z \rightarrow b$

Tampak adanya sejumlah variabel yang tidak berlanjut: *P*, *W*, dan *Y*. Bisa terjadi terdapat sejumlah variabel yang tidak akan tercapai dari *S* sehingga variabel-variabel berikut aturan produksi ybs bisa dieliminasi.

Untuk masukan *bacab* maka PDA akan menerima masukan ini melalui beberapa move sbb.

$(q_0, bacab, Z_0)$ $(q_0, acab, BZ_0)$ (q_0, cab, ABZ_0) (q_1, ab, ABZ_0)

(q_1, b, BZ_0) (q_1, Λ, Z_0) (q_1, Λ, Λ)

Menurut grammar yang dihasilkan akan diperoleh penurunan kiri (*leftmost derivation*) sbb.

$$
\begin{aligned}
S &\Rightarrow [q_0, Z_0, q_1] \\
&\Rightarrow b[q_0, B, q_1][q_1, Z_0, q_1] \\
&\Rightarrow ba[q_0, A, q_1][q_1, B, q_1][q_1, Z_0, q_1] \\
&\Rightarrow bac[q_1, A, q_1][q_1, B, q_1][q_1, Z_0, q_1] \\
&\Rightarrow baca[q_1, B, q_1][q_1, Z_0, q_1] \\
&\Rightarrow bacab[q_1, Z_0, q_1] \\
&\Rightarrow bacab
\end{aligned}
$$