Push Down Automata

Kuliah Teori Bahasa dan Automata
Program Studi Ilmu Komputer
Fasilkom UI

Prepared by:
Rahmad Mahendra

Definisi PDA

Push Down Automata (PDA) adalah Finite State Machine yang dilengkapi dengan sebuah *unlimited stack*.

PDA M adalah tupel $(K, \Sigma, \Gamma, \Delta, s, A)$ dengan:

- K : himpunan berhingga *states*.
- Σ : alfabet input
- Γ : alfabet *stack*
- $s \in K$, adalah *start state*
- $A \subseteq K$, adalah himpunan *accepting states*
- Δ : relasi transisi, yang merupakan *subset* berhingga dari

$$\underbrace{\left(K \times (\Sigma \cup \{\varepsilon\}) \times \Gamma^* \right)}_{\substack{\text{state} \quad \text{input or } \varepsilon \quad \text{string of symbols to pop from top of stack}}} \times \underbrace{\left(K \times \Gamma^* \right)}_{\substack{\text{state} \quad \text{string of symbols to push on top of stack}}}$$

Konfigurasi PDA

- Konfigurasi dari PDA *M* adalah elemen dari $K \times \Sigma^* \times \Gamma^*$, yang secara berurutan menunjukkan:
  - status saat ini (*current state*)
  - input yang belum dibaca
  - isi *stack*
- Konfigurasi awal (*initial configuration*) dari PDA *M* untuk string input *w* adalah (*s*, *w*, ε)
- *Stack* pada *M* dinyatakan sebagai string. Elemen teratas dalam *stack* bersesuaian dengan karakter pertama (paling kiri) string.

Relasi Yield-in-One-Step

- Seperti pada FSM, pada PDA juga dapat didefinisikan relasi *yield-in-one-step*, $\vdash_M$
- Misalkan $c \in \Sigma \cup \{\varepsilon\}$, $\gamma_1, \gamma_2, \gamma \in \Gamma^*$, $w \in \Sigma^*$

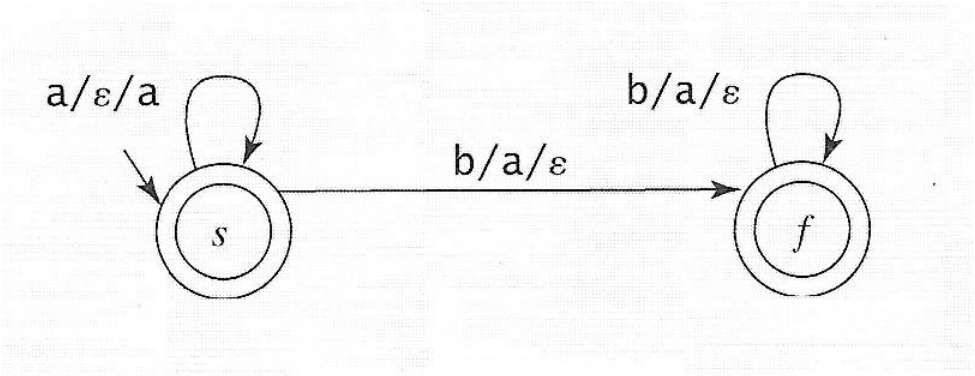
  $(q_1, cw, \gamma_1\gamma) \vdash_M (q_2, w, \gamma_2\gamma)$ **iff** $((q_1, c, \gamma_1), (q_2, \gamma_2)) \in \Delta$
- Transisi $((q_1, c, \gamma_1), (q_2, \gamma_2))$ dapat dinyatakan juga sebagai $\boldsymbol{c / \gamma_1 / \gamma_2}$
  - *M* hanya mengambil transisi jika string γ_1 bersesuaian dengan elemen teratas pada *stack*. *M pop* γ_1 kemudian *push* γ_2
  - Jika $\gamma_1 = \varepsilon$, maka *M* tidak perlu mengecek status *stack* saat ini. TIDAK sama artinya dengan *stack* dalam keadaan kosong.

Komputasi oleh PDA

- Jika C adalah komputasi yang dilakukan oleh M pada input $w \in \Sigma^*$
  - C: *accepting computation* **iff** $C = (s, w, \varepsilon) \vdash_{M^*} (q, \varepsilon, \varepsilon)$ untuk suatu $q \in A$
  - C: *rejecting computation* **iff** $C = (s, w, \varepsilon) \vdash_{M^*} (q, w', \alpha)$ di mana C bukan *accepting computation* dan tidak ada *move* yang mungkin dari (q, w', α), dengan kata lain konfigurasi *halt*.
- M menerima (*accept*) w **iff** paling tidak satu komputasi menerimanya. M menolak (*reject*) w **iff** semua komputasi menolaknya.
- Bahasa yang diterima oleh M, notasi $L(M)$, adalah himpunan semua string w yang diterima oleh M

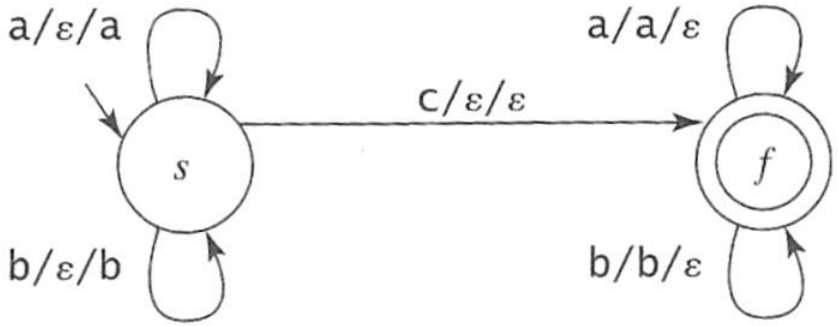
Contoh-1 PDA

- $A^nB^n = \{a^nb^n : n \geq 0\}$
- Bahasa A^nB^n diterima oleh $M = (K, \Sigma, \Gamma, \Delta, s, A)$ di mana
  - $K = \{s, f\}$ $A = \{s, f\}$
  - $\Sigma = \{a, b\}$ $\Gamma = \{a\}$
  - $\Delta = \{((s, a, \varepsilon), (s, a)), ((s, b, a), (f, \varepsilon)), ((f, b, a), (f, \varepsilon))\}$

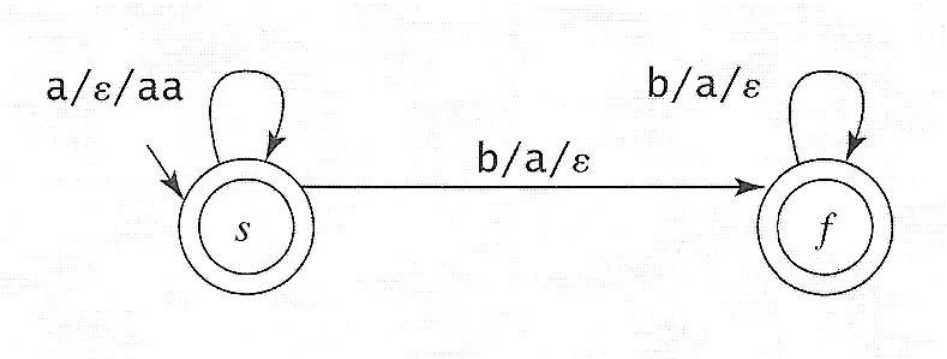


Contoh-2 PDA

- $WcW^R = \{wcw^R: \ w \in \{a, b\}^*\}$



- $A^nB^{2n} = \{a^nb^{2n}: n \geq 0\}$

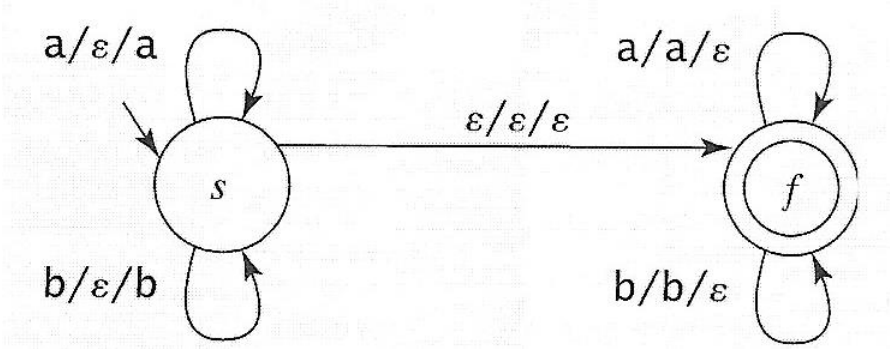


PDA Deterministik

- PDA *M* adalah deterministik **iff** tidak terdapat konfigurasi *M* yang memiliki pilihan lebih dari satu transisi.
  1. Δ_M tidak mengandung sekelompok transisi yang berkompetisi satu sama lain
  2. Jika $q \in A$, tidak terdapat transisi $((q, \varepsilon, \varepsilon), (p, x))$. Transisi pada *accepting state M* deterministik harus membaca input atau mengeluarkan isi (*pop*) *stack*.
- PDA deterministik hanya memiliki *path* komputasi tunggal untuk setiap string.

Contoh-3 PDA

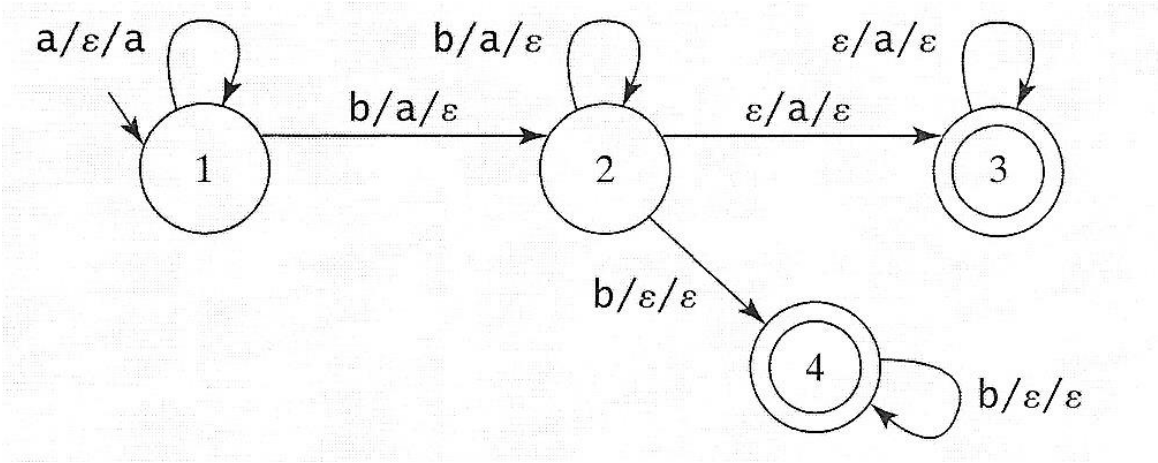
- *PalEven* = $\{ww^R: \ w \in \{a, b\}^*\}$



M non deterministik karena tidak tahu kapan akan mencapai tengah input. Setiap membaca input, *M* menebak apakah sudah sampai di tengah input.

Contoh-4 PDA

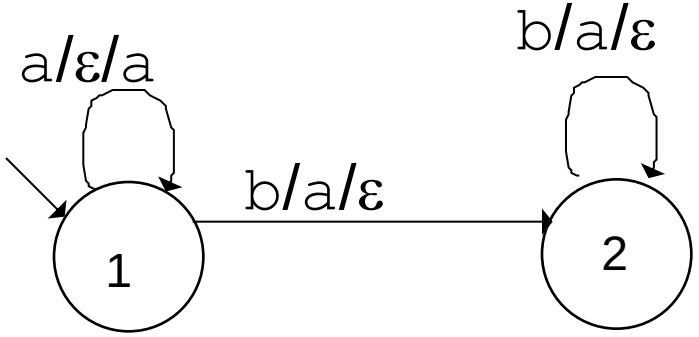
- $L = \{a^m b^n : m \neq n;\ m, n > 0\}$
- L dipecah menjadi dua *sub-language* $\{a^m b^n : 0 < m < n\}$ dan $\{a^m b^n : 0 < n < m\}$



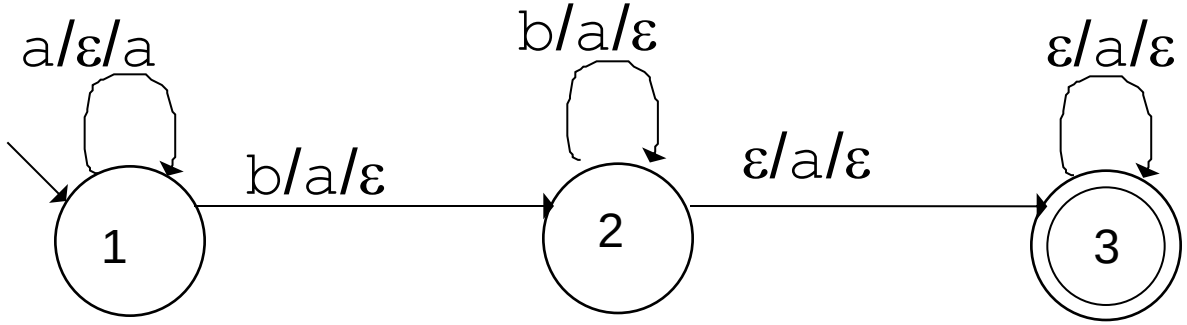
- Apakah PDA di atas deterministik?

Contoh 4 (con't)

$L = \{a^m b^n : m \neq n; m, n > 0\}$

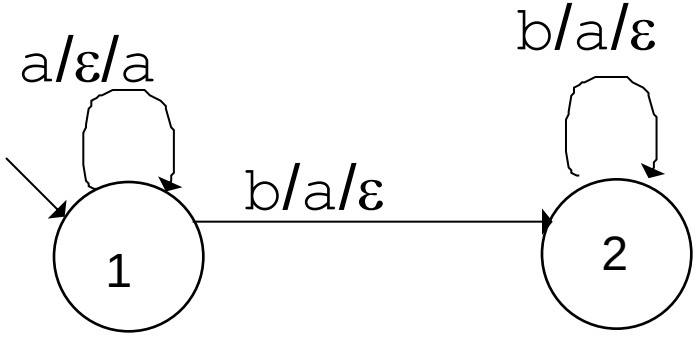


- Jika input empty, stack masih ada ($m < n$) (accept):

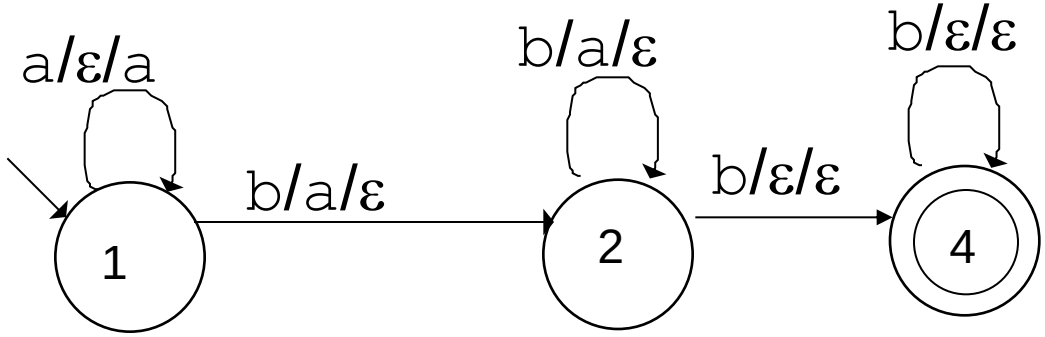


Contoh 4 (con't)

$L = \{a^m b^n : m \neq n; m, n > 0\}$

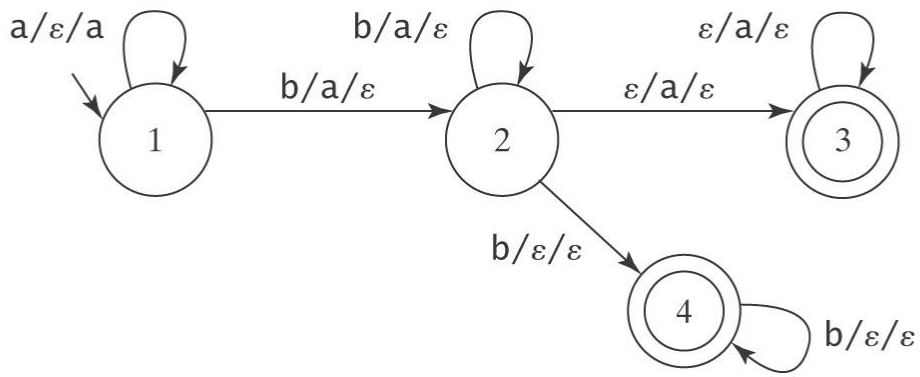


- Stack empty, input masih ada ($m > n$) (accept):



Contoh 4 – Non Deterministik

$L = \{a^m b^n : m \neq n; m, n > 0\}$



- Sampai state 2 masih deterministik
- Dari state 2, PDA di atas non deterministik dari state 2 bisa ke state 3 (jika input habis) atau ke state 4 (jika stack empty)
- Ada problem?

PDA Non Deterministik

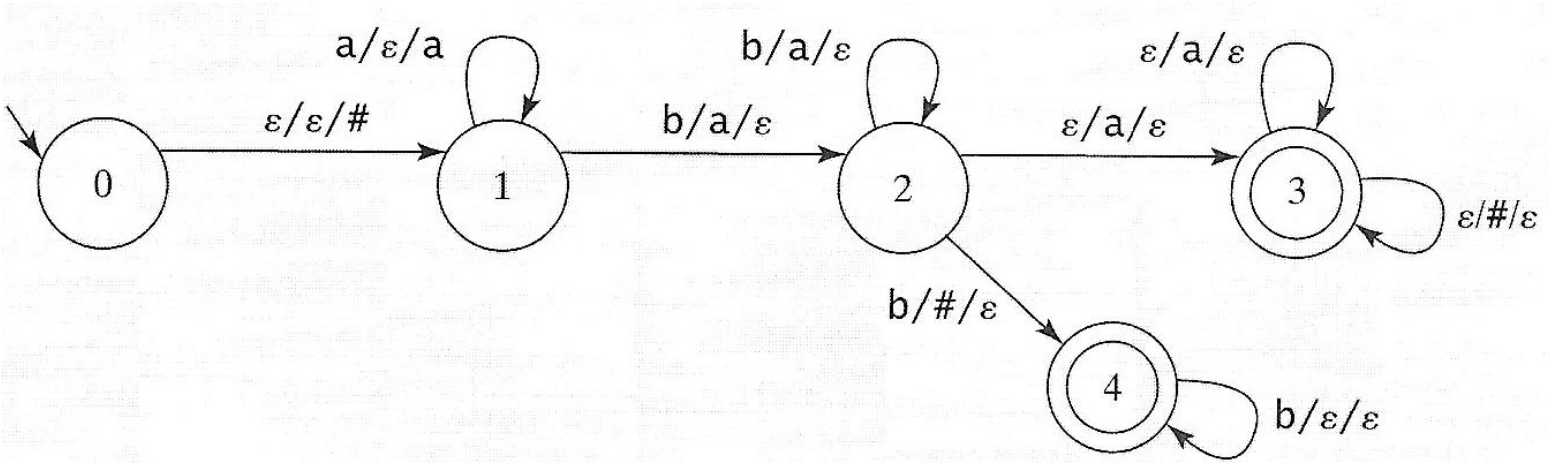
- Non determinisme pada PDA:
  - Transisi yang harus diambil ketika *stack* kosong berkompetisi dengan satu atau beberapa *move* yang membaca sejumlah string pada *stack*.
    - Dapat ditangani dengan menambahkan penanda bagian bawah *stack*.
  - Transisi yang harus diambil ketika *input stream* selesai dibaca berkompetisi dengan satu atau beberapa *move* yang membaca sejumlah karakter input.
    - Dapat ditangani dengan menambahkan penanda akhir string.

Contoh-4 PDA (lanjutan)

- $L = \{a^m b^n : m \neq n;\ \ m, n > 0\}$

Mereduksi Non Determinisme

1. Menambahkan karakter penanda bagian bawah *stack*



Contoh-4 PDA (lanjutan)

- $L = \{a^m b^n : m \neq n;\ \ m, n > 0\}$

Mereduksi Non Determinisme *(Out of Input Problem)*

2. Menambahkan karakter penanda akhir string