アルゴリズム論 1
第 6 回
プッシュダウンオートマトン (1)

関川 浩

2017/05/24

第 4 回から第 7 回の目標

正規表現と fa: よくできたシステムだが能力が低い

より能力が高いシステムを導入する

- 文脈自由文法 (第 4, 5 回)
- プッシュダウンオートマトン (第 6, 7 回)

第 6 回の目標:

- プッシュダウンオートマトンの導入
- プッシュダウンオートマトンの設計 (次回に続く)
  - ゲスについて

1. プッシュダウンオートマトン
   - プッシュダウンオートマトン (pda) の導入
   - pda の定義
   - 様相
   - pda によって認識される言語
   - 例題
2. プッシュダウンオートマトンの設計
   - pda の設計
   - 例題 2 の場合

プッシュダウンオートマトン (pda) の導入

fa の能力の限界: 内部の記憶状態が有限個
$\Longrightarrow$ 記憶状態を無限個にするため, 補助記憶装置を導入

無限個の状態をどう記述するか (すべてを書き下すことは不可能)

$\Longrightarrow$ 補助記憶装置の入出力動作を制限
$\Longrightarrow$ 有限で記述可能に

プッシュダウンオートマトン (pushdown automaton. pda と略す):
補助記憶装置としてプッシュダウンスタック (スタックと略す) を使用

スタック

以下の 4 種類の動作が許されている

(a) スタックの先頭の記号を読む
(読んだ記号以外については一切情報が得られない)

(b) スタックの先頭の記号を取り去る (ポップ)

(c) 新たな記号 (複数可) をスタックの先頭に詰め込む (プッシュ)

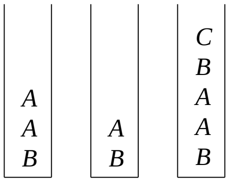
(d) スタックの内容には手をつけない

(1) の状態 (スタックが AAB を保持) から:

(a) を実行 $\Longrightarrow$ 情報として A が得られる

(b) を実行 $\Longrightarrow$ (2)

(c) を実行 (新たな記号は B, C) $\Rightarrow$ (3)

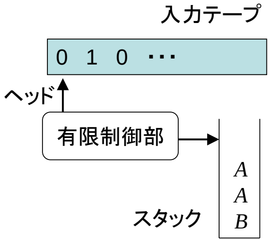


(1) (2) (3)

pda の構造, 動作

構造

- 入力テープと有限制御部は fa と同じ
- 補助記憶装置として 1 個のスタック



1 ステップの動作

(1) ヘッドの下の入力記号とスタックの先頭記号を読んで入力ヘッドは 1 コマ右に移動
(2) (1) で読んだ記号と現在の状態で決まるスタック動作 (b) または (c) または (d) を実行
(3) (1) で読んだ記号と現在の状態で決まる次の状態に遷移

pda の例

例

スタックは最初, 空とする

- 0 を読んだらスタックに 1 個の記号 Z をプッシュ
- 1 を読んだらスタックをポップ

入力列が 0^m1^n とすると:

- $m = n \iff$ 入力列を読み終わったときスタックがちょうど空
- $m > n \iff$ 入力列を読み終わったときスタックが空ではない
- $m < n \iff 1^n$ を読んでいる途中でスタックが空

pda の定義

定義 1 (pda)

pda: 六つ組 $M = (K, \Sigma, \Gamma, \delta, s_0, Z_0)$

- K, Σ, Γ: 空ではない有限集合
- 状態: K の要素
  入力記号: Σ の要素
  スタック記号: Γ の要素
  初期状態: K のある要素 s_0
  スタック初期記号: Γ のある要素 Z_0
- $\delta : K \times \Sigma \times \Gamma \longrightarrow (K \times \Gamma^*$ の有限部分集合族): 状態遷移関数 (非決定性)

注意

非決定性とした理由: 文脈自由言語との対応のため (詳細は次回)

pda の定義に関する注意

1 ステップの動作の例

- $\delta(s, a, A) = \{(s', ABC),\ (s'', \varepsilon)\}$

  nfa の場合と同様で, (s', ABC) も (s'', ε) も可能 (非決定性)

  (s', ABC): スタックの先頭記号 A を $ABC \in \Gamma^*$ で 置き換え, 状態 s' に遷移

スタック動作との対応

- スタックの先頭の記号を A とする

ポップ	$\Longleftrightarrow$	A を ε で置き換え
B, C をプッシュ	$\Longleftrightarrow$	A を CBA で置き換え
何もしない	$\Longleftrightarrow$	A を A で置き換え

様相

定義 2 (様相)

様相 (s, y, σ): 状態 s, Σ^* 上の列 y, Γ^* 上の列 σ の三つ組

様相は動作途中の pda の状況を表現

pda M の様相が (s, y, σ)

$\iff$ M は入力テープの一部を読み終わっているが

- まだ列 y を読み残しており
- スタックの内容は列 σ
- 有限制御部の状態は s

様相が与えられれば, このあとの動作が定まる

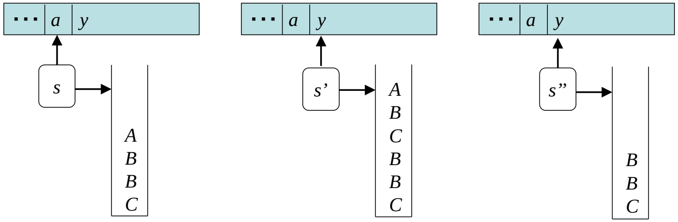
様相の 1 ステップ変化

pda の状況が (a) で, $\delta(s, a, A) = \{(s', ABC), (s'', \varepsilon)\}$ とする
$\Longrightarrow$ 1 ステップ後の状況は (b) あるいは (c)

様相 $(s, ay, ABBC)$ は

様相 $(s', y, ABCBBC)$ (あるいは, 様相 (s'', y, BBC))

に 1 ステップで変わりうると呼ぶ



(a) (b) (c)

pda によって認識される言語

定義 3

様相 S_1 から S_2 へ 1 ステップで変わりうるとき $S_1 \Rightarrow S_2$ と書く

定義 4

$S_0, S_1, \ldots, S_k$ $(k \geq 0)$ を様相として, $S_0 \Rightarrow S_1 \Rightarrow \cdots \Rightarrow S_k$ であるとき, S_0 は S_k へ何ステップかで変わりうると呼び, $S_0 \overset{*}{\Rightarrow} S_k$ と書く

定義 5

$M = (K, \Sigma, \Gamma, \delta, s_0, Z_0)$: pda

- $x \in \Sigma^*$ がある $s \in K$ に対して条件 $(s_0, x, Z_0) \overset{*}{\Rightarrow} (s, \varepsilon, \varepsilon)$ を満たすとき, x は M によって受理されるという
- 受理される列の全体を M によって認識される言語と呼ぶ

注意: (s_0, x, Z_0) を初期様相, $(s, \varepsilon, \varepsilon)$ を受理様相という

例題 1 (1/3)

例題 1

$\{xcx^{\mathrm{R}} \mid x \in \{a,b\}^*\}$ を認識する pda を構成せよ

解答 (1/2)

$\{s_0, s_1\}$: 状態集合 (初期状態は s_0)

$\{A, B, Z_0\}$: スタック記号

- c が現れるまで
  読んだ入力記号に対応するスタック記号をスタックにプッシュ (最初のみ, Z_0 を読んだ入力記号で置き換え) (積み上げモード)

  対応: $a \longleftrightarrow A$, $b \longleftrightarrow B$ (以下, この対応で記号を同一視)

- c が現れたら
  次の入力記号から, 入力記号とスタック先頭の記号とが一致するかチェックしてはポップ (チェックモード)

例題 1 (2/3)

解答 (2/2)

状態遷移関数:

$$
\begin{array}{lll}
\delta(s_0, a, Z_0) = \{(s_0, A)\}, & \delta(s_0, b, Z_0) = \{(s_0, B)\}, & \delta(s_0, c, Z_0) = \{(s_0, \varepsilon)\}, \\
\delta(s_0, a, A) = \{(s_0, AA)\}, & \delta(s_0, b, A) = \{(s_0, BA)\}, & \delta(s_0, c, A) = \{(s_1, A)\}, \\
\delta(s_0, a, B) = \{(s_0, AB)\}, & \delta(s_0, b, B) = \{(s_0, BB)\}, & \delta(s_0, c, B) = \{(s_1, B)\}, \\
\delta(s_1, a, A) = \{(s_1, \varepsilon)\}, & \delta(s_1, b, B) = \{(s_1, \varepsilon)\} &
\end{array}
$$

上記に現れないものは状態遷移関数の値を $\emptyset$ とする

- 受理する例: $abaaacaaaba$

$$
\begin{array}{lll}
& (s_0, abaaacaaaba, Z_0) & \Rightarrow (s_0, baaacaaaba, A) \\
\Rightarrow & (s_0, aaacaaaba, BA) & \Rightarrow (s_0, aacaaaba, ABA) \\
\Rightarrow & (s_0, acaaaba, AABA) & \Rightarrow (s_0, caaaba, AAABA) \\
\Rightarrow & (s_1, aaaba, AAABA) & \Rightarrow (s_1, aaba, AABA) \\
\Rightarrow & (s_1, aba, ABA) & \Rightarrow (s_1, ba, BA) \\
\Rightarrow & (s_1, a, A) & \Rightarrow (s_1, \varepsilon, \varepsilon)
\end{array}
$$

- 受理しない例 1: $abaaacaabba$

$$
\begin{aligned}
&(s_0, abaaacaabba, Z_0) \Rightarrow (s_0, baaacaabba, A)\\
\Rightarrow\ &(s_0, aaacaabba, BA) \Rightarrow (s_0, aacaabba, ABA)\\
\Rightarrow\ &(s_0, acaabba, AABA) \Rightarrow (s_0, caabba, AAABA)\\
\Rightarrow\ &(s_1, aabba, AAABA) \Rightarrow (s_1, abba, AABA)\\
\Rightarrow\ &(s_1, bba, ABA)
\end{aligned}
$$

: 受理様相ではなく, 次の様相も存在しない

- 受理しない例 2: $abaaacaaa$

$$
\begin{aligned}
&(s_0, abaaacaaa, Z_0) \Rightarrow (s_0, baaacaaa, A)\\
\Rightarrow\ &(s_0, aaacaaa, BA) \Rightarrow (s_0, aacaaa, ABA)\\
\Rightarrow\ &(s_0, acaaa, AABA) \Rightarrow (s_0, caaa, AAABA)\\
\Rightarrow\ &(s_1, aaa, AAABA) \Rightarrow (s_1, aa, AABA)\\
\Rightarrow\ &(s_1, a, ABA) \Rightarrow (s_1, \varepsilon, BA)
\end{aligned}
$$

: 受理様相ではなく, 次の様相も存在しない

例題 2 (1/3)

例題 2

$\{xx^{\mathrm{R}} \mid x \in \{a,b\}\{a,b\}^*\}$ を認識する pda を構成せよ

解答

- 基本的には例題 1 と同じ
  前半が積み上げモード, 後半がチェックモード

- 状態遷移関数:

$$
\begin{aligned}
&\delta(s_0, a, Z_0) = \{(s_0, A)\}, && \delta(s_0, b, Z_0) = \{(s_0, B)\}, \\
&\delta(s_0, a, A) = \{(s_0, AA), (s_1, \varepsilon)\}, && \delta(s_0, b, A) = \{(s_0, BA)\}, \\
&\delta(s_0, a, B) = \{(s_0, AB)\}, && \delta(s_0, b, B) = \{(s_0, BB), (s_1, \varepsilon)\}, \\
&\delta(s_1, a, A) = \{(s_1, \varepsilon)\}, && \delta(s_1, b, B) = \{(s_1, \varepsilon)\}
\end{aligned}
$$

  上記に現れないものは状態遷移関数の値を $\emptyset$ とする

例題 2 (2/3)

受理する例: $abaaaaaaba$

$$
\begin{aligned}
& (s_0, abaaaaaaba, Z_0) \Rightarrow (s_0, baaaaaaba, A) \\
\Rightarrow\ & (s_0, aaaaaaba, BA) \ \Rightarrow (s_0, aaaaaba, ABA) \\
\Rightarrow\ & (s_0, aaaaba, AABA) \Rightarrow (s_0, aaaba, AAABA) \\
\Rightarrow\ & (s_1, aaba, AABA) \quad \Rightarrow (s_1, aba, ABA) \\
\Rightarrow\ & (s_1, ba, BA) \qquad\quad \Rightarrow (s_1, a, A) \\
\Rightarrow\ & (s_1, \varepsilon, \varepsilon)
\end{aligned}
$$

もし, $(s_0, aaaaba, AABA) \Rightarrow (s_0, aaaba, AAABA)$ ではなく,
$(s_0, aaaaba, AABA) \Rightarrow (s_1, aaaba, ABA)$ と遷移すると:

$(s_1, aaaba, ABA) \Rightarrow (s_1, aaba, BA)$: 受理様相ではなく,
次の様相も存在しない

注意

受理様相に至る様相の系列が一つでも存在すれば受理
$\Longrightarrow$ 受理に失敗する様相の系列が存在しても差し支えない

例題 2 (3/3)

受理しない例: $abaaaaabba$

ちょうど中央右隣でチェック状態に移行した場合:

$$\begin{aligned}
& (s_0, abaaaaabba, Z_0) \Rightarrow (s_0, baaaaabba, A) \\
\Rightarrow\ & (s_0, aaaaabba, BA) \Rightarrow (s_0, aaaabba, ABA) \\
\Rightarrow\ & (s_0, aaabba, AABA) \Rightarrow (s_0, aabba, AAABA) \\
\Rightarrow\ & (s_1, abba, AABA) \Rightarrow (s_1, bba, ABA)
\end{aligned}$$

: 受理様相ではなく, 次の様相も存在しない

注意

- 受理しない場合は, すべての様相の系列を非受理とすること
- この例題では, チェック状態に入ると, 状態遷移は存在するなら一意
  $\Longrightarrow$ 非受理は容易に確認可能

1. プッシュダウンオートマトン
2. プッシュダウンオートマトンの設計

プッシュダウンオートマトンの設計

pda の設計

複雑なため, 状態遷移関数を書き下すことはあまりない

$\Longrightarrow$ 動作を必要にして十分な詳しさで説明することに

例題 2 ($\{xx^{\mathrm{R}}\}$ を認識する M) の場合 (1/4)

M の動作: 積み上げモードとチェックモード

1. 積み上げモードの 1 ステップの動作

   入力テープの現在読んでいる記号 c が列の中央右隣か否かをゲス

   (∗) ゲス (guess): 根拠なしの推測

   - 否定的にゲス $\Longrightarrow$ c をスタックにプッシュ, このモード続行
   - 肯定的にゲス $\Longrightarrow$ c とスタックの先頭記号を比較
     - 同じならポップしてチェックモードへ
     - 異なれば停止

2. チェックモードの動作の 1 ステップの動作

   入力テープの現在読んでいる記号とスタックの先頭記号を比較

   - 同じならポップしてこのモード続行
   - 異なれば停止

例題 2 ($\{xx^{\mathrm{R}}\}$ を認識する M) の場合 (2/4)

- 入力テープの中央右隣の場所を知りたい
  しかし, 左から順に入力記号を読んでいく限り不可能
  $\Longrightarrow$ ゲスする

- 各場面でありうる複数のゲスを状態遷移関数に取り込む
  $\Longrightarrow$ 状態遷移関数の値を複数の動作の集合とする

例題 2 ($\{xx^{\mathrm{R}}\}$ を認識する M) の場合 (3/4)

列が受理される定義の言い換え

列が受理される $\Longleftrightarrow$ 受理様相に至るゲスの列が存在する

例: $abaaaaaaba$ の場合

- 最初から 5 番目の記号までは中央右隣ではないとゲス
- 6 番目の記号で中央右隣であるとゲスすれば受理

$$
\begin{aligned}
& (s_0, abaaaaaaba, Z_0) \Rightarrow (s_0, baaaaaaba, A) \\
\Rightarrow\ & (s_0, aaaaaaba, BA) \ \ \Rightarrow (s_0, aaaaaba, ABA) \\
\Rightarrow\ & (s_0, aaaaba, AABA) \Rightarrow (s_0, aaaba, AAABA) \\
\Rightarrow\ & (s_1, aaba, AABA) \quad\ \Rightarrow (s_1, aba, ABA) \\
\Rightarrow\ & (s_1, ba, BA) \qquad\qquad \Rightarrow (s_1, a, A) \\
\Rightarrow\ & (s_1, \varepsilon, \varepsilon)
\end{aligned}
$$

例題 2 ($\{xx^{\mathrm{R}}\}$ を認識する M) の場合 (4/4)

注意

前スライドの言い換えで, 個々のゲスが正しいか否かには言及していない
(状態遷移関数との対応から, ゲスの結果を対等に扱うのは当然)

ゲスは推測

$\Longrightarrow$ 受理すべき列は, ゲスが当たりのとき受理, が分かりやすい方針

ゲスがはずれたとき, 上記方針に沿った設計なら

- 受理すべき列は受理/非受理どちらでもよい
  (当たりのゲスの列で受理されるので)
- 受理すべきではない列は受理してはいけない

$\Longrightarrow$ ゲスがはずれたときは, いかなる列も非受理にするのが安全