

6. プッシュダウン・オートマトン:

6.1. プッシュダウン・オートマトン(PDA)の定義

6.2. PDAの言語

6.3. PDAとCFGの等価性

(6.4. 決定性プッシュダウン・オートマトン)

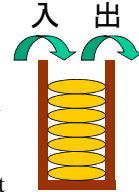
1/27

6.1. プッシュダウン・オートマトン (PDA)の定義

プッシュダウン・オートマトン(PDA)=
オートマトン+プッシュダウンスタック

プッシュダウンスタック...

- 食堂のお盆
- (最近見ない)コインケース
- いわゆる stack



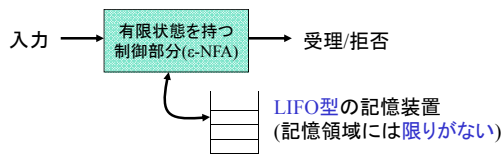
LIFO: Last In First Out

2/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.1. 直感的な説明

PDAとはε-NFAが stack を一つ持った機械モデル



3/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.1. 直感的な説明

PDAとはε-NFAが stack を一つ持った機械モデル

- 受理条件:
1. 受理状態
 2. スタックが空

動作プロセス:

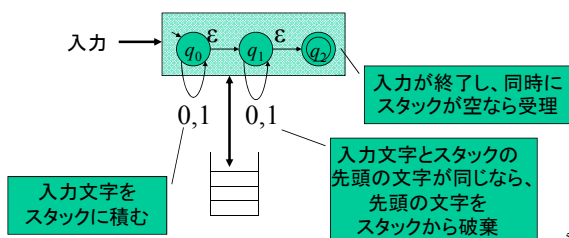
1. 入力を1文字読む
ε-動作のときは0文字
2. 入力 x とスタックの一番上の文字 y に応じて状態遷移する
3. スタックを操作する:
 1. 一番上の文字を取り出して捨てる
 2. 一番上の文字を書き換える
 3. 2に加えて、いくつかの文字を記憶
4. 入力が終わって受理条件を満たしていたら受理。入力が残っているならステップ1へ

4/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.1. 直感的な説明

例) $L = \{ ww^R \mid w \in \{0,1\}^* \}$ を受理する PDA

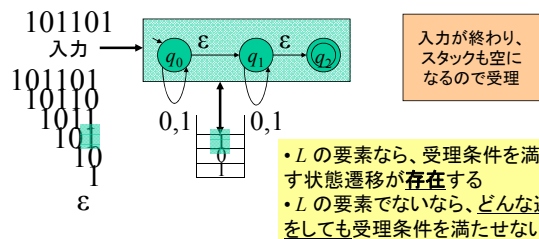


5/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.1. 直感的な説明

例) $L = \{ ww^R \mid w \in \{0,1\}^* \}$ を受理する PDA



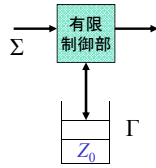
6/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.2. PDA の形式的定義

$$PDA P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$$

- Q : 状態の有限集合
- Σ : 入力アルファベット
- q_0 : $q_0 \in Q$ を満たす初期状態
- F : $F \subseteq Q$ を満たす受理状態
- Γ : スタックアルファベット; スタックに記憶する文字集合
- Z_0 : $Z_0 \in \Gamma$ を満たす開始記号; スタックには最初にこの文字が1つ入っているとす。 (スタックの[底]を判定するための便宜上の文字)



7/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.2. PDA の形式的定義

$$PDA P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$$

関数 $\delta: Q \times \Sigma \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$

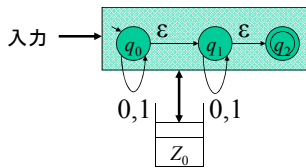
- δ は「現在の状態」「入力1文字」「スタックのトップの文字X」が与えられ、
- 「次の状態」「Xを置き換える文字列Y」を返す関数
 - 「入力1文字」は ϵ も可
 - Yは次が可:
 - Yが ϵ ; Xの破棄
 - Yが1文字; Xの置換
 - Yが2文字以上; 置換+追加
- δ は非決定的なので、同じ入力に対する行き先が複数ありえる

8/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.2. PDAの形式的な定義

例) $L = \{ ww^R \mid w \in \{0,1\}^* \}$ を受理する PDA P



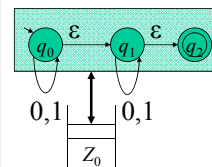
$$P = (\{q_0, q_1, q_2\}, \{0,1\}, \{0,1, Z_0\}, \delta, q_0, Z_0, \{q_2\})$$

9/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.2. PDAの形式的な定義

例) $L = \{ ww^R \mid w \in \{0,1\}^* \}$ を受理する PDA P



q_0 に関する δ

- 入力をスタックに積む

$$\delta(q_0, 0, Z_0) = \{(q_0, 0Z_0)\}, \delta(q_0, 1, Z_0) = \{(q_0, 1Z_0)\}$$

$$\delta(q_0, 0, 0) = \{(q_0, 00)\}, \delta(q_0, 1, 0) = \{(q_0, 10)\}$$

$$\delta(q_0, 0, 1) = \{(q_0, 01)\}, \delta(q_0, 1, 1) = \{(q_0, 11)\}$$

- ϵ 動作で q_1 に遷移

$$\delta(q_0, \epsilon, Z_0) = \{(q_1, Z_0)\}, \delta(q_0, \epsilon, 0) = \{(q_1, 0)\}, \delta(q_0, \epsilon, 1) = \{(q_1, 1)\}$$

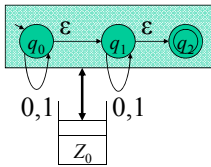
$$P = (\{q_0, q_1, q_2\}, \{0,1\}, \{0,1, Z_0\}, \delta, q_0, Z_0, \{q_2\})$$

10/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.2. PDAの形式的な定義

例) $L = \{ ww^R \mid w \in \{0,1\}^* \}$ を受理する PDA P



q_1 に関する δ

- 入力とスタックのトップを比較

$$\delta(q_1, 0, 0) = \{(q_1, \epsilon)\}$$

$$\delta(q_1, 1, 1) = \{(q_1, \epsilon)\}$$

- 入力とスタックのトップがどちらも空になったら ϵ 動作で q_2 に遷移

$$\delta(q_1, \epsilon, Z_0) = \{(q_2, Z_0)\}$$

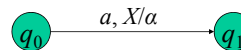
$$P = (\{q_0, q_1, q_2\}, \{0,1\}, \{0,1, Z_0\}, \delta, q_0, Z_0, \{q_2\})$$

11/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.3. PDAの図による表現

- 関数 δ を辺のラベルで表現する



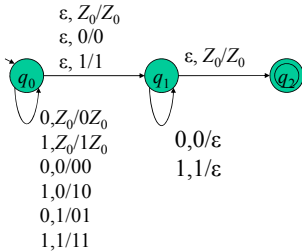
$$\delta(q_0, a, X) = \{ \dots, (q_1, a), \dots \}$$
 の図示

12/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.3. PDAの図による表現

例) $L = \{ ww^R \mid w \in \{0,1\}^* \}$ を受理する PDA P の δ



13/27

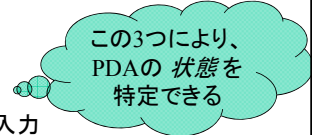
6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.4. PDAの状況の関係

- オートマトンは「状態」だけで特定できた
- PDAでは「状態」+「スタックの文字列」でないと状態が特定できない
- ⇒ δ 記法は適切でない

PDAの状況とは (q, w, γ) 。

- ただし
- $q \in Q$: 状態
 - $w \in \Sigma^*$: 残りの入力
 - $\gamma \in \Gamma^*$: スタックの文字列



状況は時点表示, ID (Instantaneous Description)とも言う ^{14/27}

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.4. PDAの状況の関係

- PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ において $\delta(q, a, X)$ が $\delta(p, \alpha)$ を含むなら、 $w \in \Sigma^*, \beta \in \Gamma^*$ に対して

$$(q, aw, X\beta) \vdash (p, w, \alpha\beta)$$

と書く(P がわかっているならと書く)。

- 0ステップ以上の一般の動作を表現するときは \vdash^* を使う:

1. 状況 I に対し $I \vdash^* I$
2. 状況 I, J, K に対し $I \vdash J$ & $J \vdash^* K$ なら $I \vdash^* K$

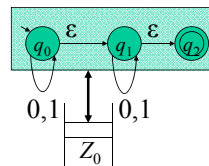
15/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.4. PDAの状況の関係

例) $L = \{ ww^R \mid w \in \{0,1\}^* \}$ を受理する PDA

入力101101に対する状況の遷移:



$$(q_0, 101101, Z_0) \vdash (q_0, 01101, 1Z_0)$$

$$(q_0, 101101, Z_0) \vdash (q_1, 101101, Z_0)$$

$$(q_0, 101101, Z_0) \vdash^* (q_0, 01101, 1Z_0)$$

$$(q_0, 101101, Z_0) \vdash^* (q_1, 101101, Z_0)$$

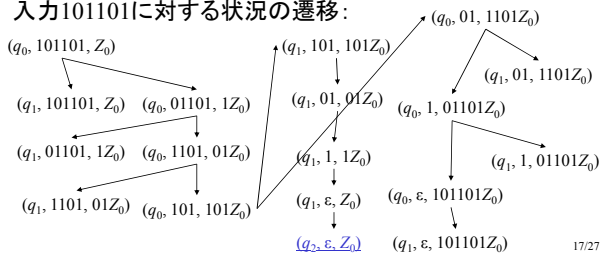
16/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.4. PDAの状況の関係

例) $L = \{ ww^R \mid w \in \{0,1\}^* \}$ を受理する PDA

入力101101に対する状況の遷移:



17/27

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.4. PDAの状況の関係

[定理] PDA P の計算プロセスを考えると、

$$(q, x, \alpha) \vdash^* (p, y, \beta) \text{ なら } (q, xw, \alpha\gamma) \vdash^* (p, yw, \beta\gamma)$$

[定理] PDA P の計算プロセスを考えると、

$$(q, xw, \alpha) \vdash^* (p, yw, \beta) \text{ なら } (q, x, \alpha\gamma) \vdash^* (p, y, \beta\gamma)$$

★『 $(q, x, \alpha\gamma) \vdash^* (p, y, \beta\gamma) \text{ なら } (q, x, \alpha) \vdash^* (p, y, \beta)$ 』ではない

18/27

6.2. PDAの言語

PDAの受理状態:

1. 入力を読み終わったときに**受理状態**にある
2. 入力を読み終わったときに**スタックが空**

与えられたPDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ に対して、

$$L(P) = \{ w \mid \text{ある } q_f \in F \text{ に対し } (q_0, w, Z_0) \xrightarrow{*} (q_f, \epsilon, \alpha) \}$$

$$N(P) = \{ w \mid \text{ある } q \in Q \text{ に対し } (q_0, w, Z_0) \xrightarrow{*} (q, \epsilon, \epsilon) \}$$

- $N(P)$ を考えるときは F は関係ないので、6つ組 $(Q, \Sigma, \Gamma, \delta, q_0, Z_0)$ で記述することもある
- 日本語テキストの $N(P)$ の定義では $q \in F$ となっているが間違い (258ページ)

19/27

6.2. PDAの言語

PDA P によって定義される2種類の言語:

1. $L(P)$: 入力を読み終わったときに**受理状態**
2. $N(P)$: 入力を読み終わったときに**スタックが空**

➤ 一般に $L(P) \neq N(P)$

例) スタックに最後に Z_0 がいても残る P に対しては $N(P) = \emptyset$

[定理]

PDA Q に対して $N(Q) = L(P)$ を満たす PDA P が存在。
 PDA P に対して $L(P) = N(Q)$ を満たす PDA Q が存在。

20/27

6.2. PDAの言語

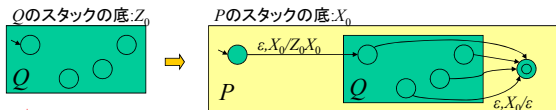
[定理]

1. PDA Q に対して $N(Q) = L(P)$ を満たす PDA P が存在。
2. PDA P に対して $L(P) = N(Q)$ を満たす PDA Q が存在。

[略証] 与えられた Q から条件を満たす P を以下の通り構成

アイデア: 最初にスタックの底に特別な記号 X_0 をつむ...

「 Q の計算でスタックが空」=「 P の計算でスタックのトップが X_0 」



★ Q の入力終了かつスタックが空のときのみ P が受理状態かつ入力終了

21/27

6.2. PDAの言語

[定理]

1. PDA Q に対して $N(Q) = L(P)$ を満たす PDA P が存在。
2. PDA P に対して $L(P) = N(Q)$ を満たす PDA Q が存在。

[略証] 与えられた P から条件を満たす Q を以下の通り構成

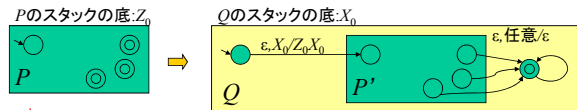
アイデア1: 最初にスタックの底に特別な記号 X_0 をつむ

... P の計算の模倣中に、スタックが空になることはない

アイデア2: P の受理状態から Q の受理状態に ϵ 動作で遷移する

... 入力を消費しないことに注意

アイデア3: Q の受理状態ではスタックの中身をすべて破棄



★ P が受理状態かつ入力終了のときのみ Q の入力終了かつスタックが空

22/27

6.3. PDA とCFGの等価性

主張:

PDAで受理できる言語=CFGで生成できる言語

定理

受理条件は[入力が終わった時点でスタックが空]

1. 任意のCFG G に対し、 $L(G) = N(P)$ を満たす PDA P が存在する。
2. 任意の PDA P に対し、 $N(P) = L(G)$ を満たす CFG G が存在する。

23/27

6.3.1. CFG G に対して $L(G) = N(P)$ を満たす PDA P が存在すること

[定理] 任意のCFG G に対し、 $L(G) = N(P)$ を満たす PDA P が存在する。

[証明のアイデア]

G における最左導出をPDA P のスタックを使って模倣する



24/27

6.3.2. PDA P に対して $N(P)=L(G)$ を満たす CFG G が存在すること

[定理] 任意の PDA P に対し、 $N(P)=L(G)$ を満たす CFG G が存在する。

[証明のアイデア]

- P における[スタック記号を1つ破棄する]動作を文法の1つの導出ルールに対応付ける
- P の[状態 p からスタック記号 X を破棄して状態 q に遷移する]ことを文法 G の非終端記号 $[pXq]$ に対応付ける

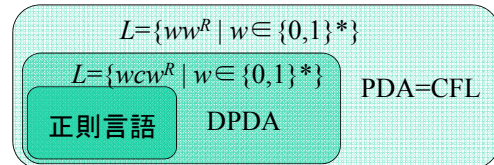
これを一つの記号とみなす

25/27

6.4. 決定性PDA(概要)

決定性PDA=PDAにおいて、非決定性を取り除いたもの。「次の状態」が一意的に決まる。

正則言語、DPDAの言語、PDAの言語の関係:



★DPDAのクラスは実用的クラス(例;LR(k),YACC)_{26/27}

大雑把なまとめ

- TuringのマシンモデルとChomskyの文法

DFA NFA 正則表現(Type 3)

決定性PDA

PDA 文脈自由文法(Type 2)

LBA 文脈依存文法(Type 1)

Turingマシン 制限なし(Type 0)

27/27