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)

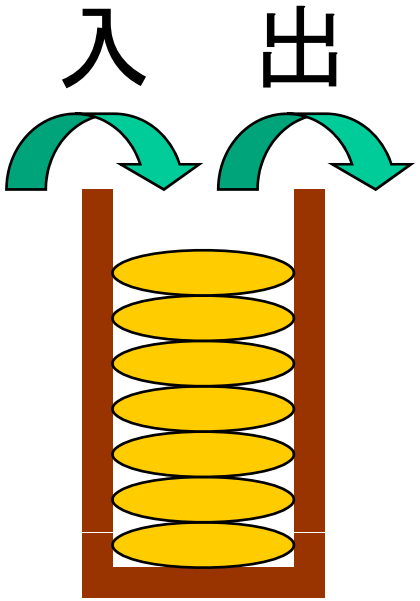
6.1. プッシュダウン・オートマトン (PDA)の定義

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

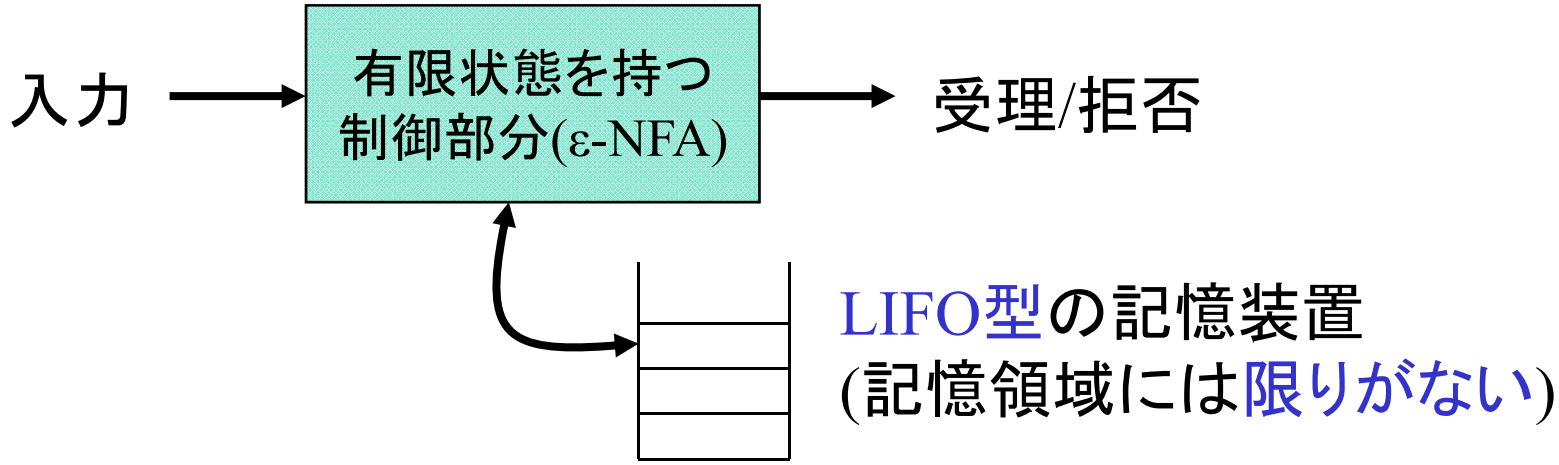
LIFO: Last In First Out



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

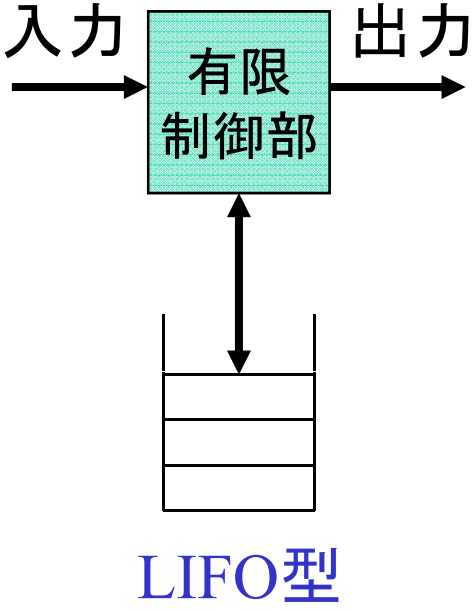


6.1. プッシュダウン・オートマトン (PDA)の定義

受理条件:
1. 受理状態
2. スタックが空

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



LIFO型

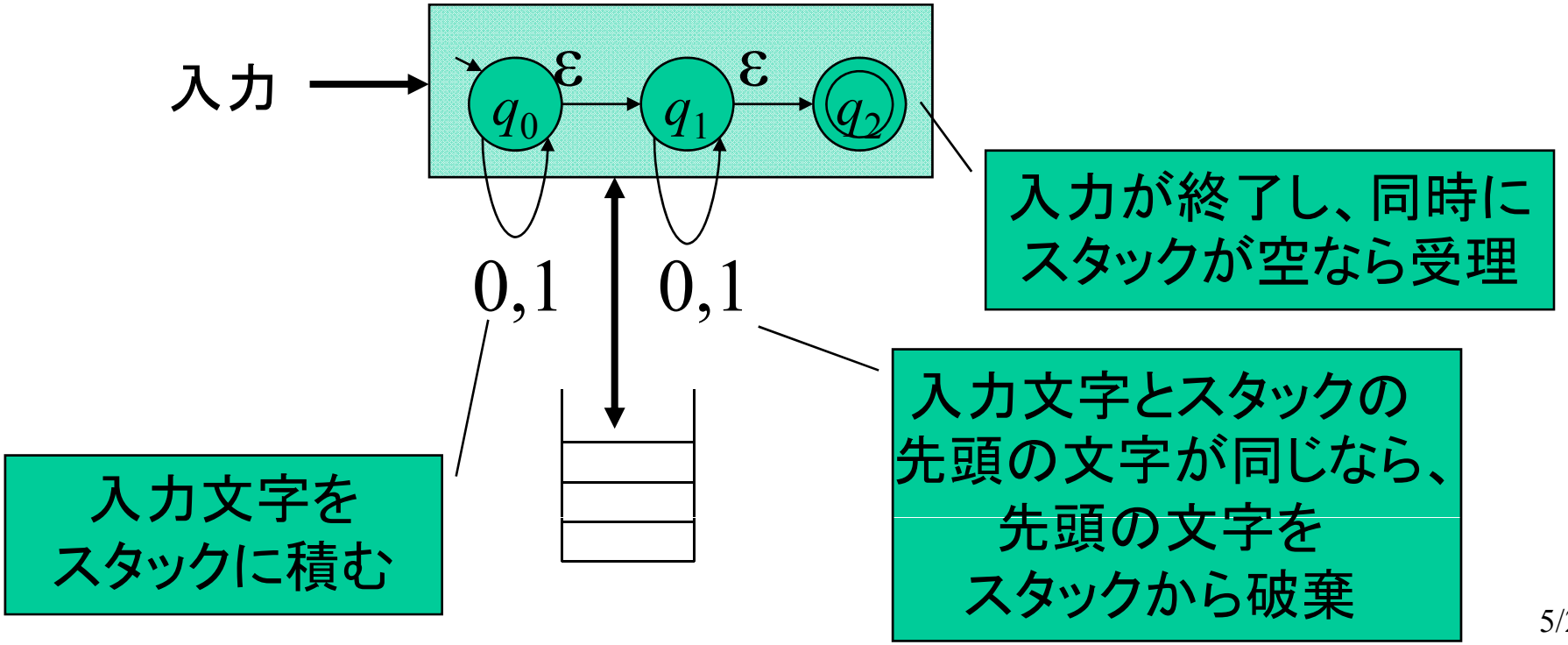
動作プロセス:

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

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

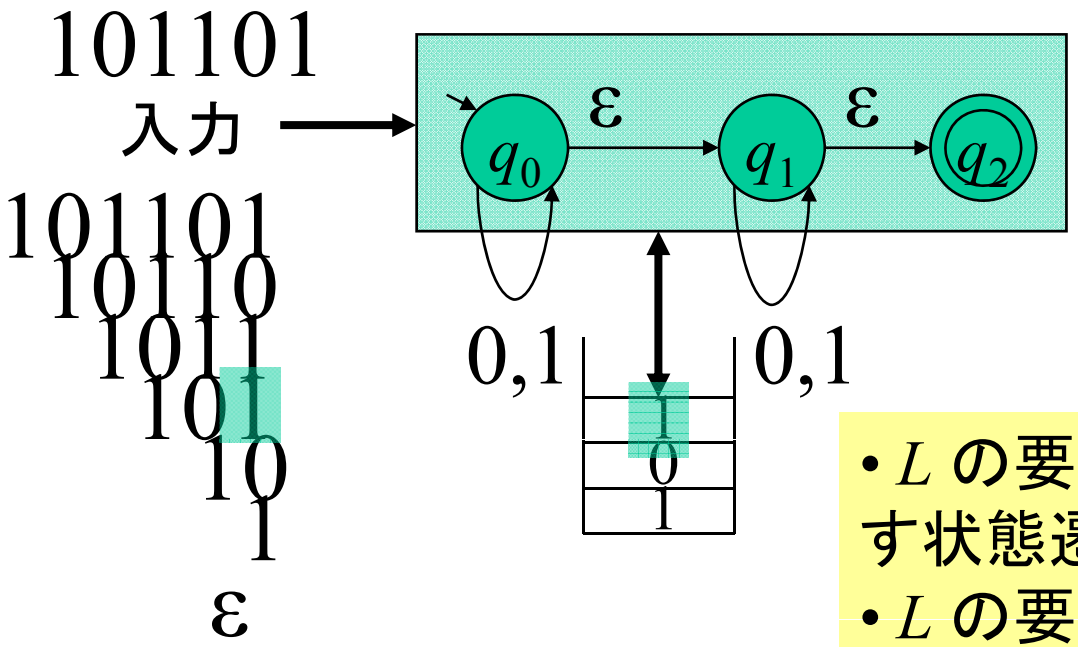
例) $L = \{ ww^R \mid w \in \{0,1\}^* \}$ を受理する PDA



6.1. 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

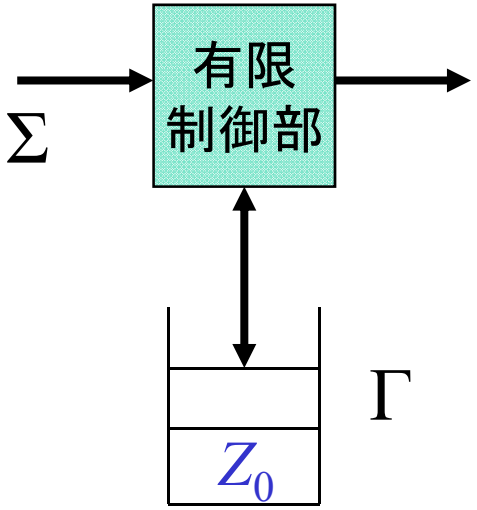
例) $L = \{ ww^R \mid w \in \{0,1\}^* \}$ を受理する PDA



入力が終わり、スタックも空になるので受理

- L の要素なら、受理条件を満たす状態遷移が存在する
- L の要素でないなら、どんな遷移をしても受理条件を満たせない

6.1. プッシュダウン・オートマトン (PDA)の定義

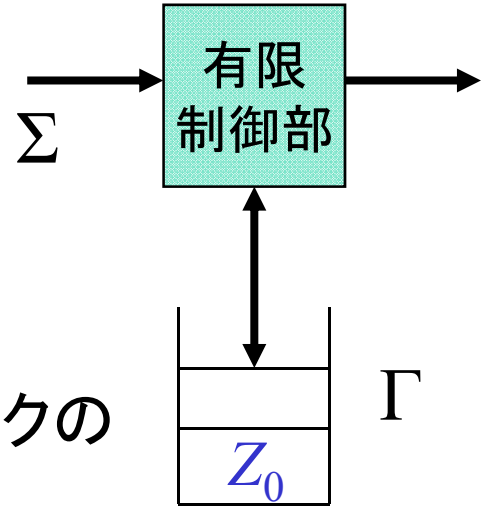


6.1.2. PDA の形式的定義

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つ入っているとする。(スタックの[底]を判定するための便宜上の文字)

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

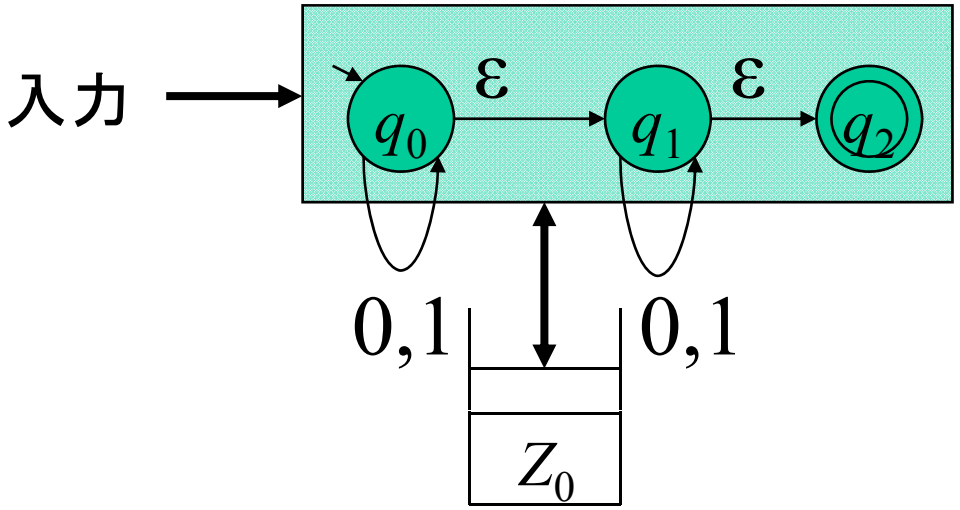
- δは「現在の状態」「入力1文字」「スタックのトップの文字X」が与えられ、
- 「次の状態」「Xを置き換える文字列Y」を返す関数
  - 「入力1文字」はεも可
  - Yは次が可:
    - Y=ε; Xの破棄
    - Yが1文字; Xの置換
    - Yが2文字以上; 置換＋追加
- δは非決定的なので、

  同じ入力に対する行き先が複数ありえる

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

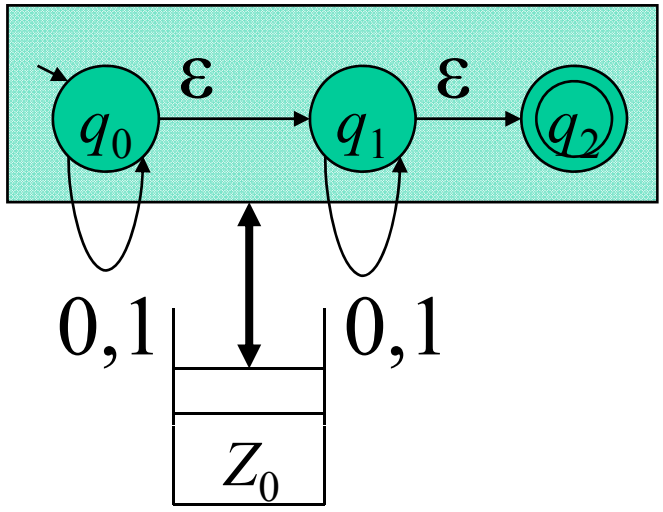


$$P = (\{q_0, q_1, q_2\}, \{0,1\}, \{0,1,Z_0\}, \delta, q_0, Z_0, \{q_2\})$$

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

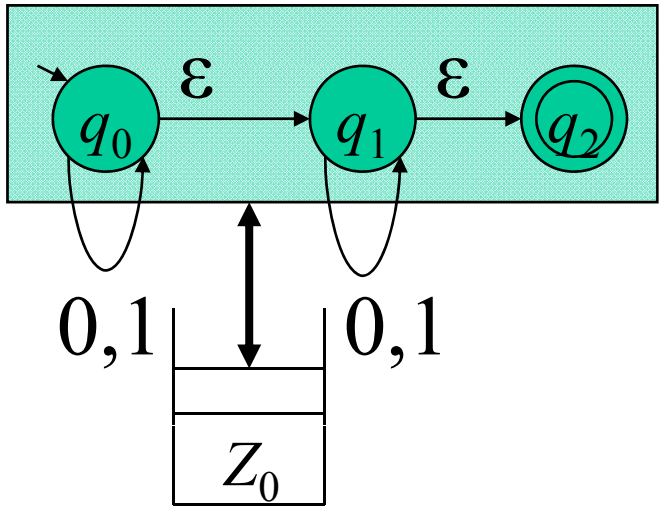
$$\delta(q_0,\varepsilon,Z_0)=\{(q_1,Z_0)\},\delta(q_0,\varepsilon,0)=\{(q_1,0)\},\delta(q_0,\varepsilon,1)=\{(q_1,1)\}$$

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

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

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

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

- 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

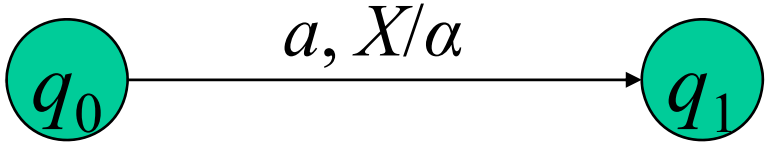
$$\delta(q_1, \varepsilon, Z_0) = \{(q_2, Z_0)\}$$

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

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

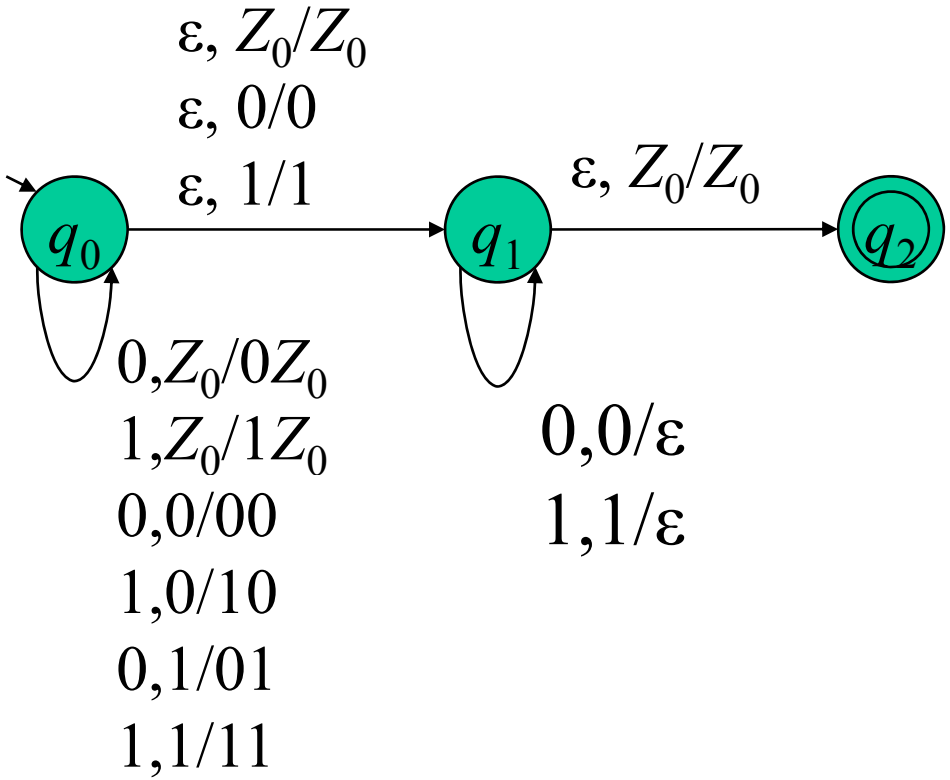


$\delta(q_0, a, X) = \{ \ldots, (q_1, \alpha), \ldots \}$ の図示

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



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

⇒ $\hat{\delta}$記法は適切でない

この3つにより、PDAの *状態* を特定できる

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

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(p,α)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$\vdash$と書く)。

PDAの1ステップを表現

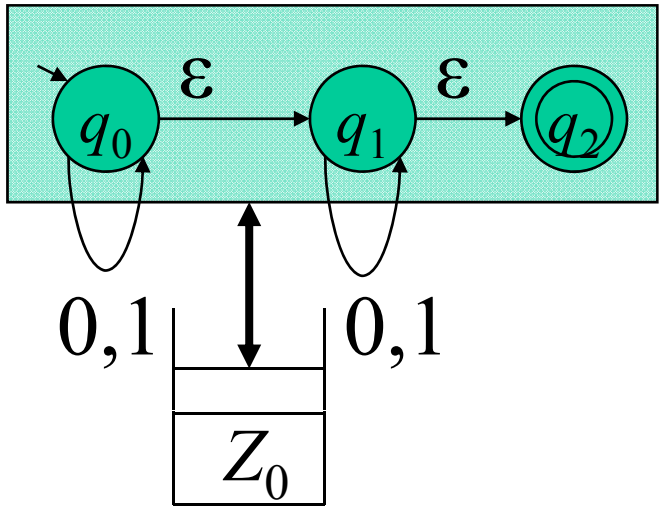
– 0ステップ以上の一般の動作を表現するときは$\vdash^*$を使う:

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 $I \vdash^* K$

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

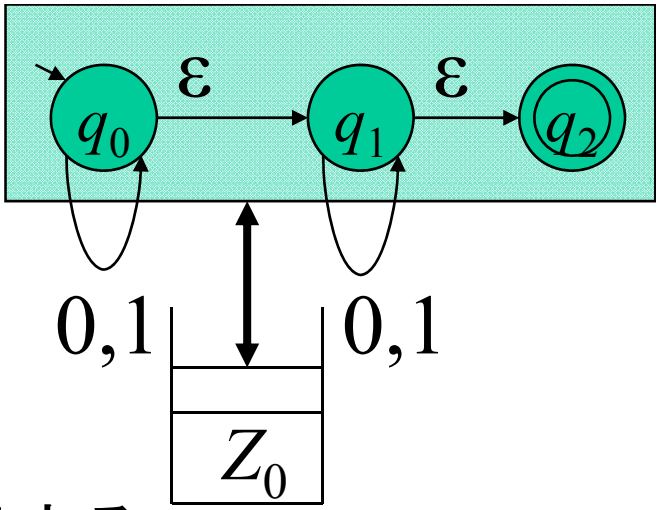


入力101101に対する状況の遷移:

$(q_0, 101101, Z_0) \vdash (q_0, 01101, 1Z_0)$
$(q_0, 101101, Z_0) \vdash (q_1, 101101, Z_0)$

$(q_0, 101101, Z_0)$ → $(q_0, 01101, 1Z_0)$
$(q_0, 101101, Z_0)$ → $(q_1, 101101, Z_0)$

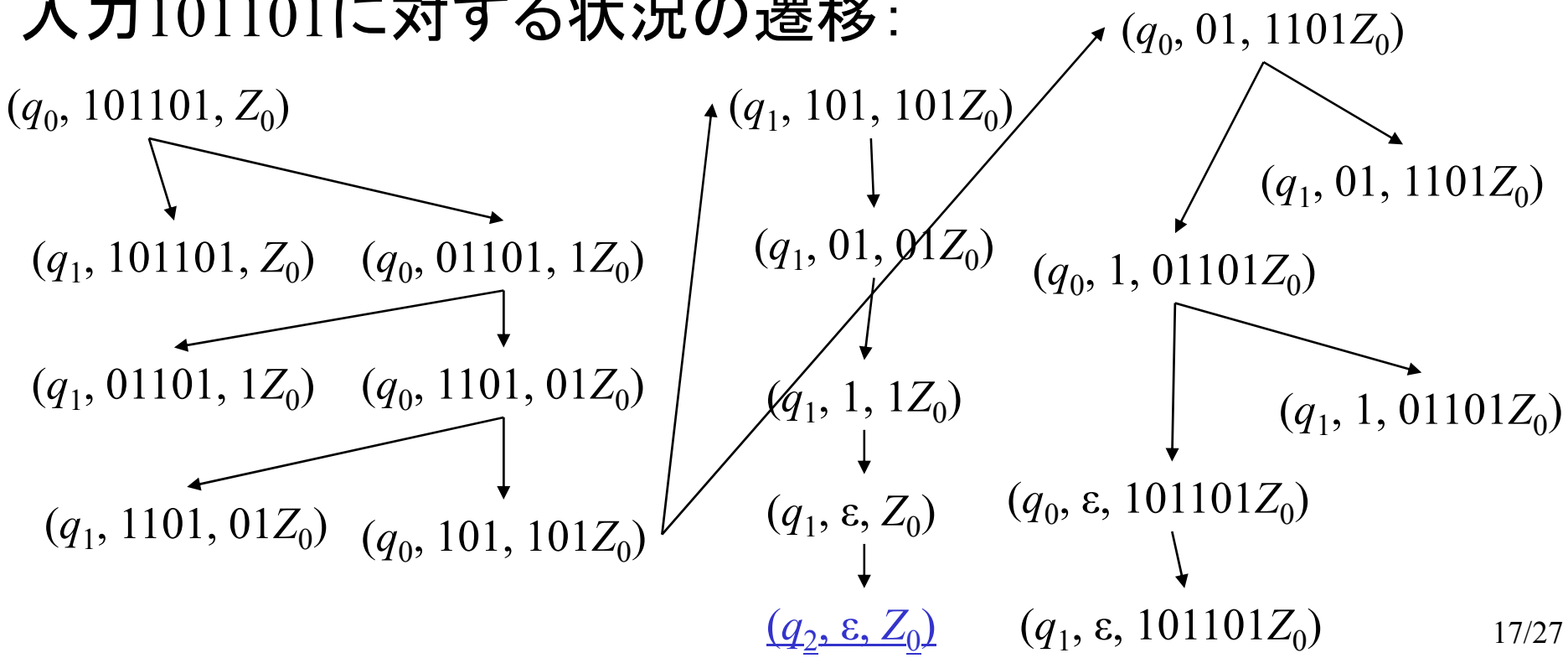
6.1. プッシュダウン・オ
(PDA)の定義



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:



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、

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

[定理] PDA P の計算プロセスを考えると、

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

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

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、

$L(P) = \{ w \mid$ ある $q_f \in F$ に対し $(q_0,w,Z_0) \vdash^* (q_f,\varepsilon,\alpha) \}$

$N(P) = \{ w \mid$ ある $q \in Q$ に対し $(q_0,w,Z_0) \vdash^* (q,\varepsilon,\varepsilon) \}$

- $N(P)$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)

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 がいつでも残る P に対しては $N(P) = \Phi$

[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。

6.2. PDAの言語

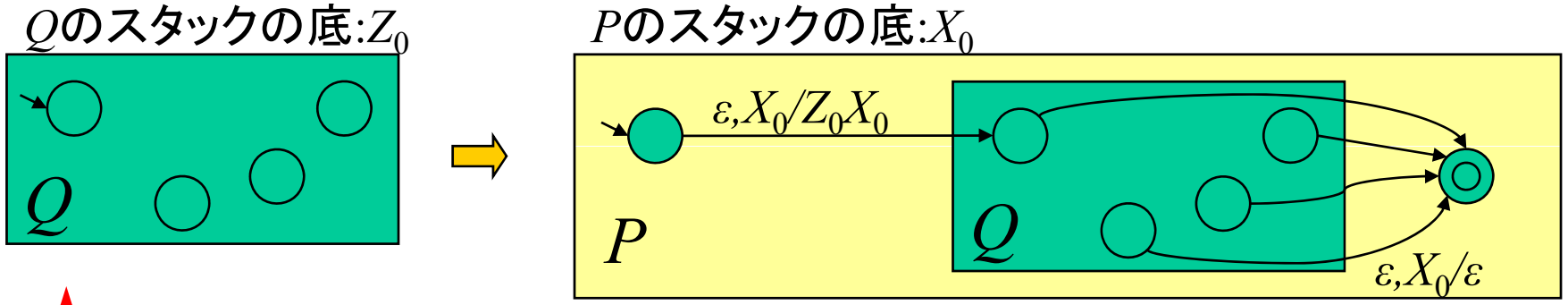
[定理]

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の入力が終了かつスタックが空

のときのみPが受理状態でかつ入力が終了

6.2. PDAの言語

[定理]

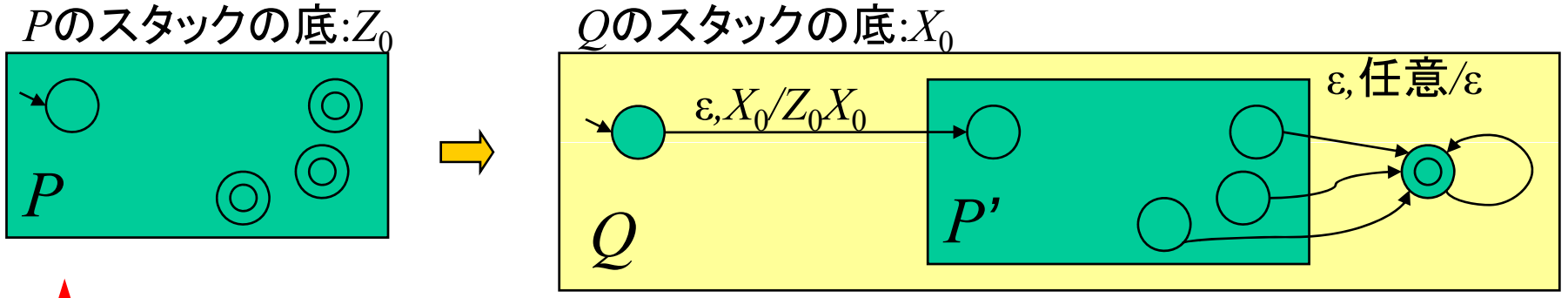
1. PDA Q に対して $N(Q) = L(P)$ を満たす PDA P が存在。
2. PDA P に対して $L(P) = N(Q)$ を満たす PDA Q が存在。

[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が受理状態でかつ入力が終了のときのみ
Qの入力が終了かつスタックが空

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

受理条件は[入力が終わった時点でスタックが空]

定理

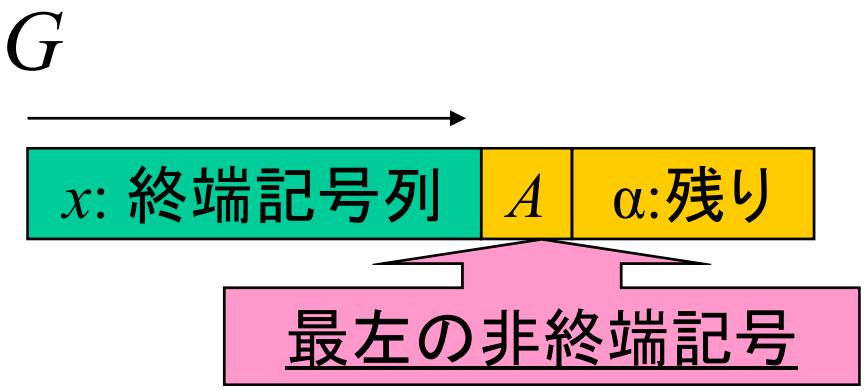
1. 任意のCFG Gに対し、$L(G)=N(P)$ を満たすPDA Pが存在する。
2. 任意の PDA P に対し、$N(P)=L(G)$ を満たすCFG Gが存在する。

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



P

- $A\alpha$をスタックに積む
- Aを書き換えながら入力と比較

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]

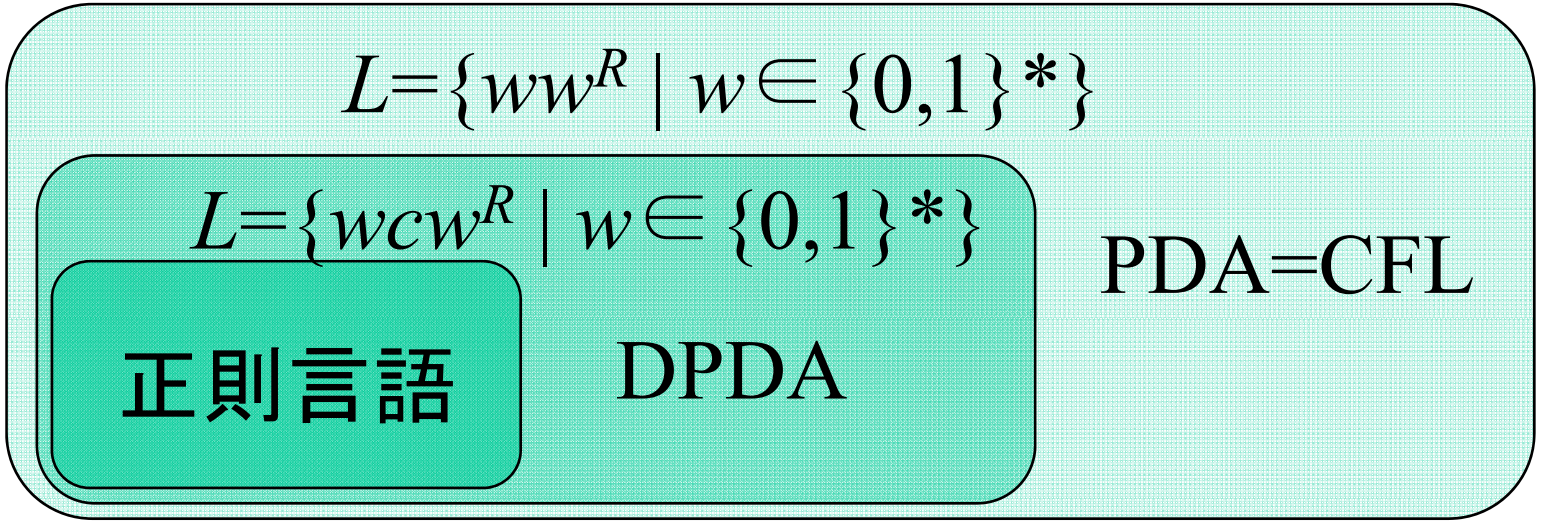
- P における[スタック記号を1つ破棄する]動作を文法の1つの導出ルールに対応付ける
- P の[状態 p からスタック記号 X を破棄して状態 q に遷移する] ことを文法 G の非終端記号 $[pXq]$ に対応付ける

これを一つの記号とみなす

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)

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)