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. Pushdown Automaton

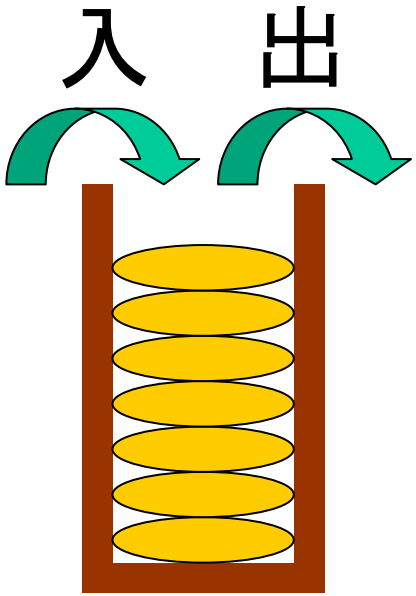
6.1. プッシュダウン・オートマトン (PDA)の定義

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 **I**n **F**irst **O**ut



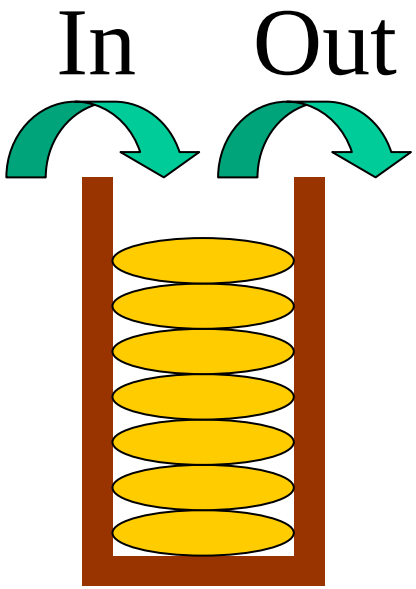
6.1. Definition of Pushdown Automaton (PDA)

Pushdown Automaton (PDA)=

Automaton + Pushdown Stack

Pushdown Stack…

- Trays in Food Court
- (Old-fashioned?) Coin case
- so-called stack

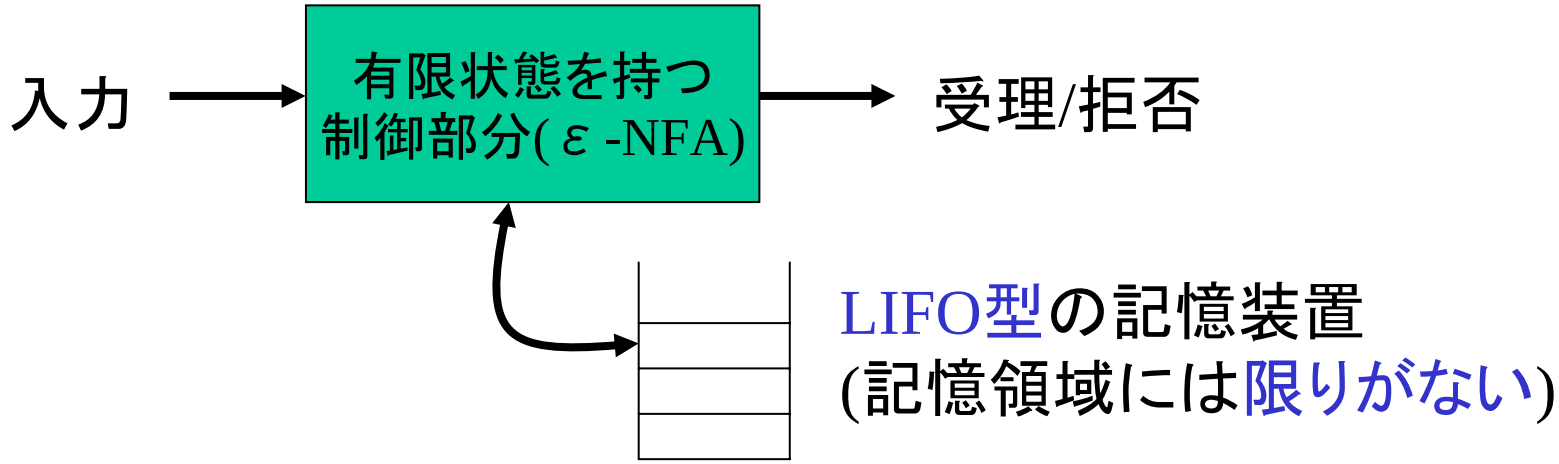


LIFO: Last **I**n **F**irst **O**ut

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

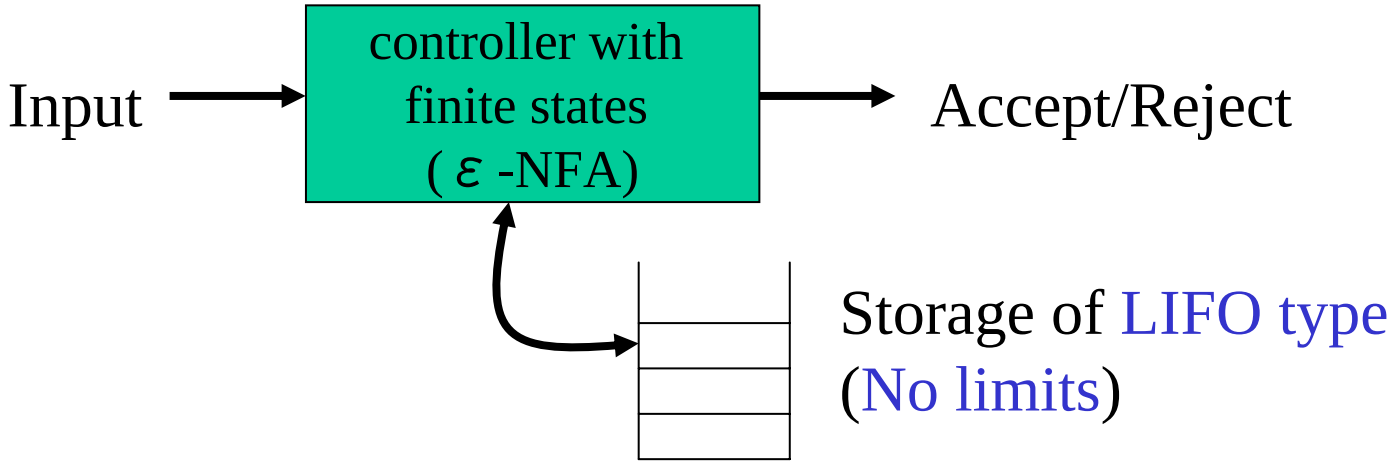
PDA とは ε -NFA が stack を一つ持った機械モデル



6.1. Definition of Pushdown Automaton (PDA)

6.1.1. Intuition

PDA consists of an ε-NFA with one stack

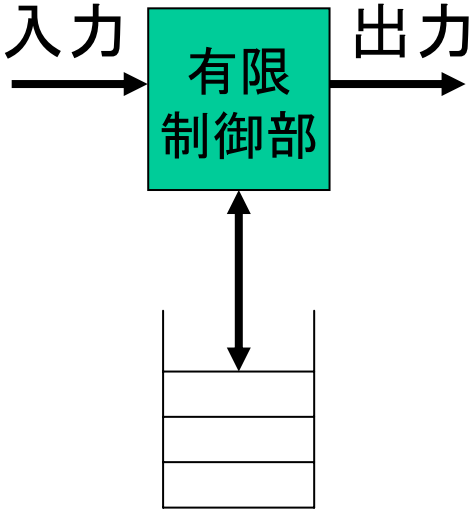


6.1. プッシュダウン・オートマトン (PDA)の定義

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:

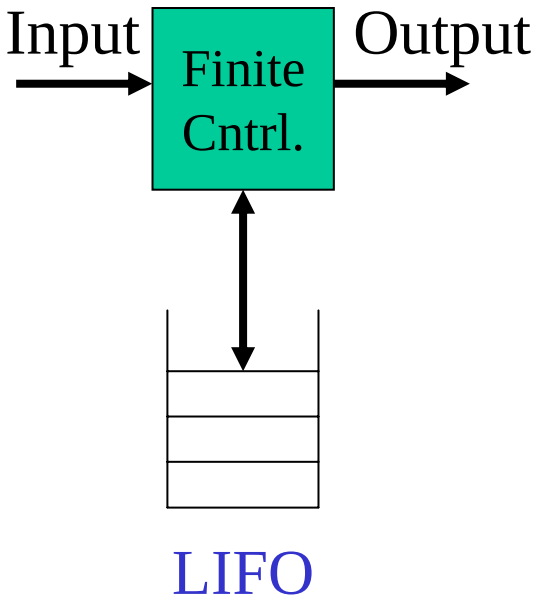
1. 入力を1文字読む
   ε -動作のときは0文字
2. 入力 x とスタックの一番上の文字 y に応じて状態遷移する
3. スタックを操作する:
   1. 一番上の文字を取り出して捨てる
   2. 一番上の文字を書き換える
   3. 2に加えて、いくつかの文字を記憶
4. 入力が終わって受理条件を満たしていたら受理。入力が残っているならステップ1へ

6.1. Definition of Pushdown Automaton (PDA)

It accepts when
1. accepting state
2. empty stack

6.1.1. Intuition

PDA consists of an ε-NFA with <u>one stack</u>



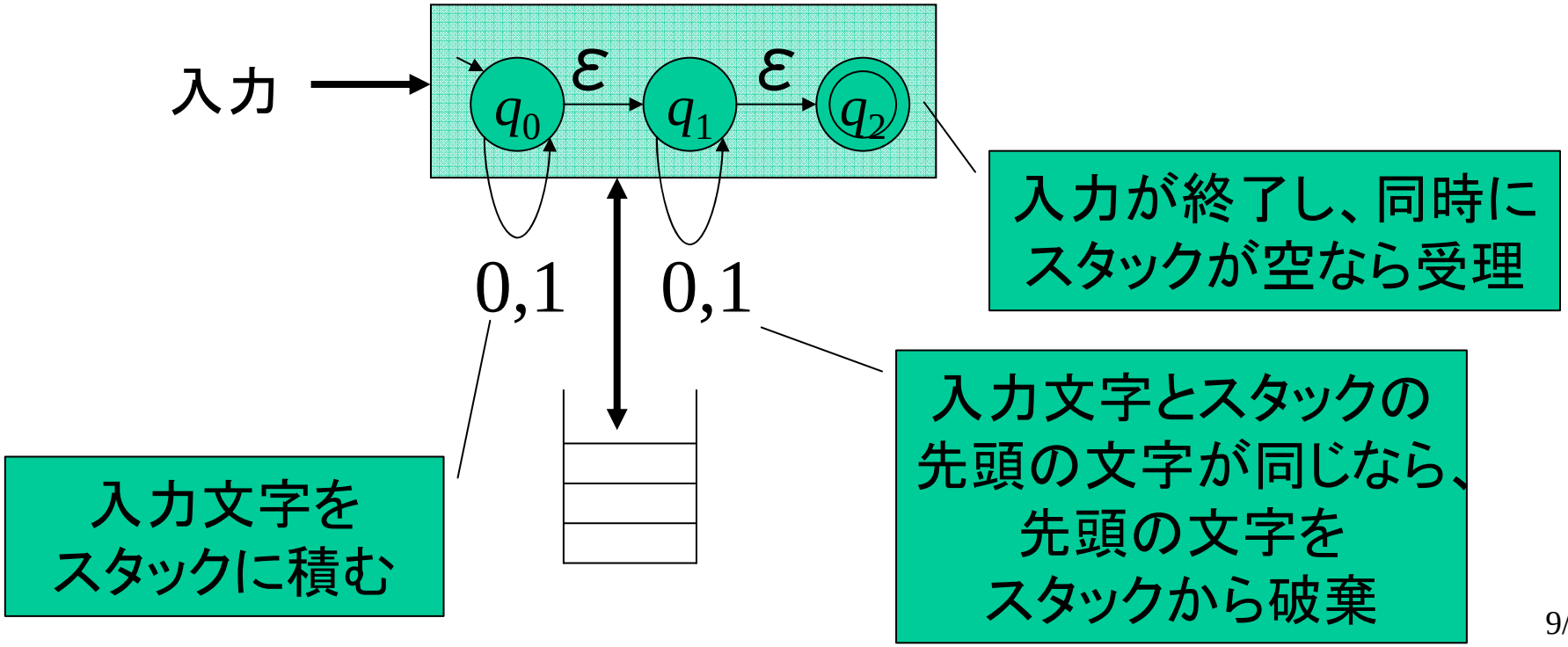
Computation process:

1. Read one letter *x* from input
   0 letter if ε-transition
2. Change the states according to input *x* and letter *y* on top of stack
3. Operation on the letter *y*:
   1. just remove *y*
   2. replace *y* by another letter *y'*
   3. replace *y* by another word *y'*
4. If input ends and <u>accept condition</u> is satisfied, accept. If input does not end, go to step 1.

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

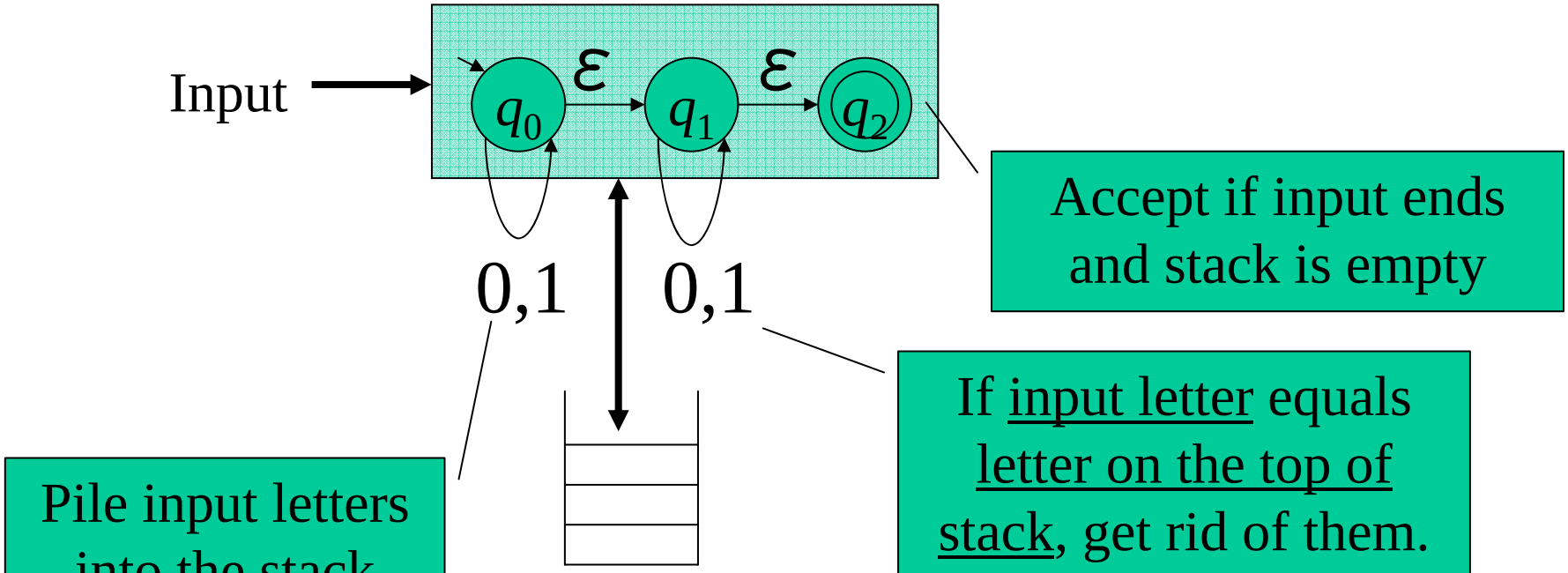
例) $L = \{ ww^R \mid w \in \{0,1\}^* \}$ を受理する PDA



6.1. Definition of Pushdown Automaton (PDA)

6.1.1. Intuition

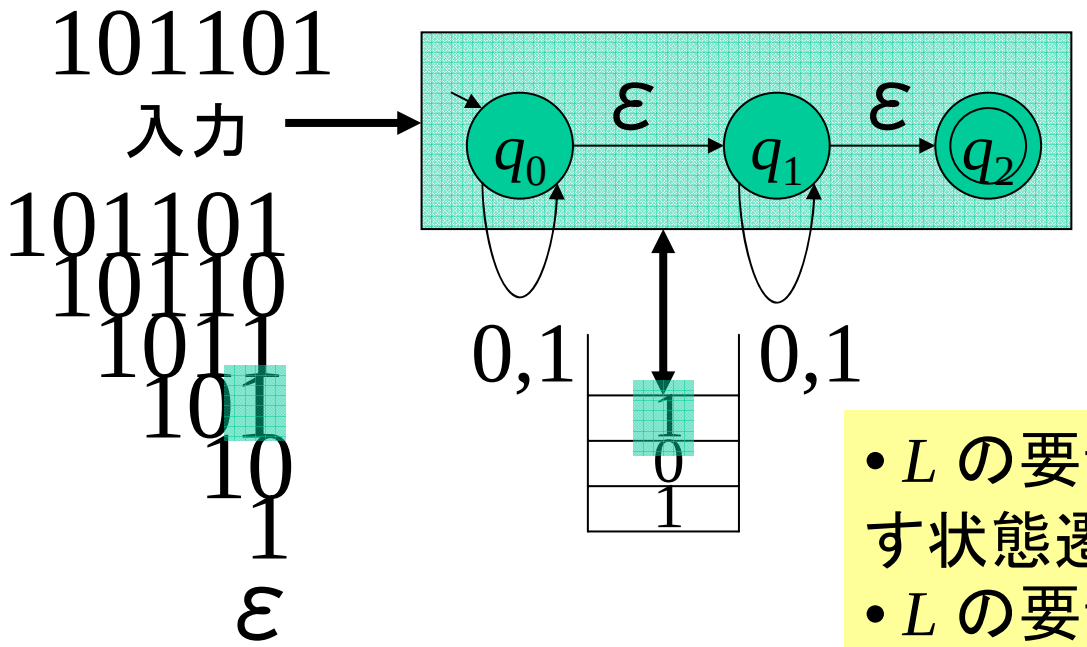
Ex) PDA accepting $L = \{ ww^R \mid w \in \{0,1\}^* \}$



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



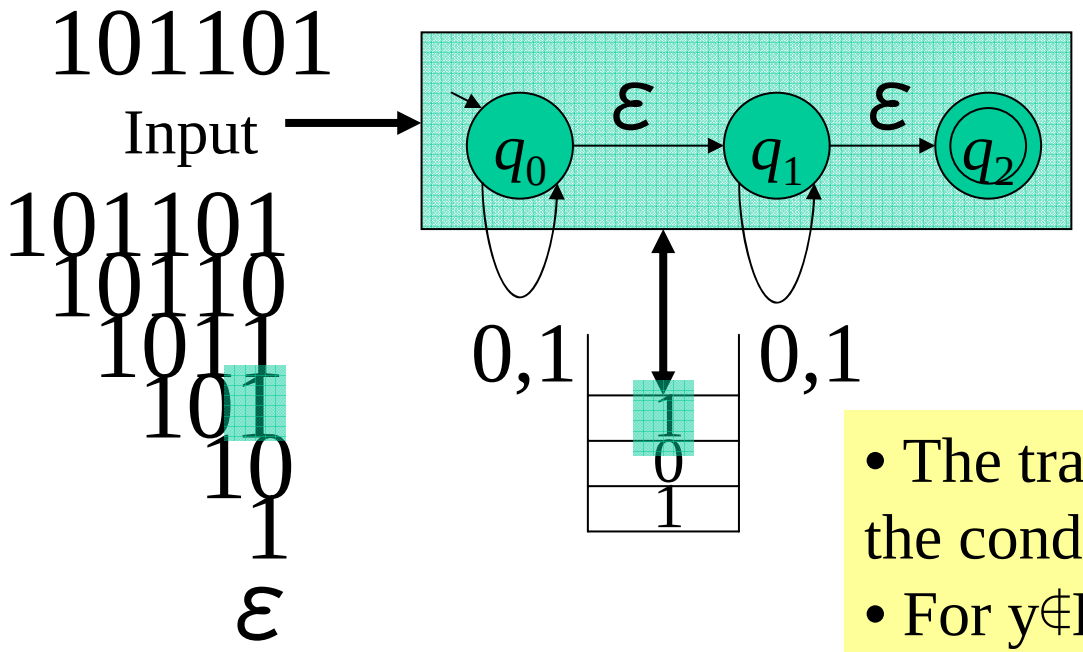
入力が終わり、スタックも空になるので受理

- 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. Definition of Pushdown Automaton (PDA)

6.1.1. Intuition

Ex) PDA accepting $L = \{ ww^R \mid w \in \{0,1\}^* \}$

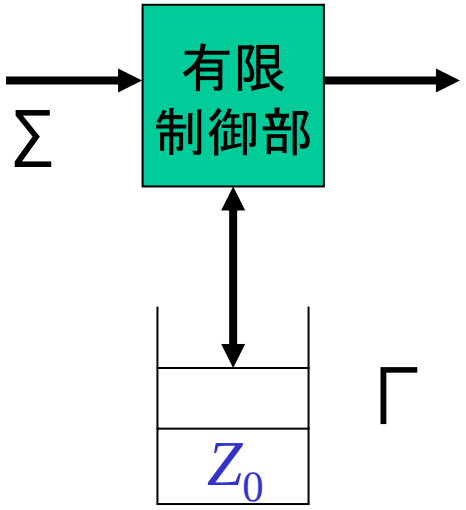


101101 can make
- input ends
- stack empty.

Hence it is accepted.

- The transitions **exist** that satisfy the conditions for $y \in L$.
- For $y \notin L$, **any transitions** do not satisfy the conditions.

6.1. プッシュダウン・オートマトン (PDA)の定義

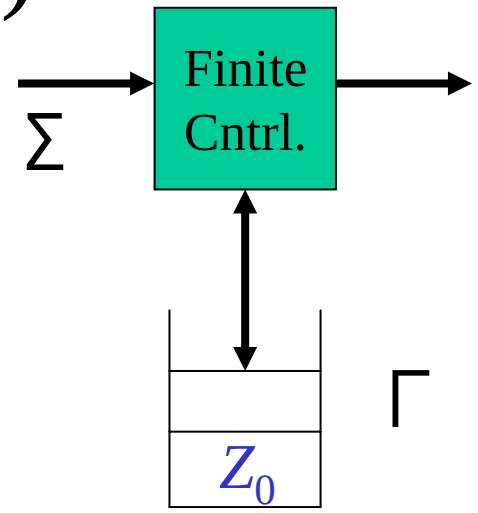


6.1.2. PDA の形式的定義

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. Definition of Pushdown Automaton (PDA)

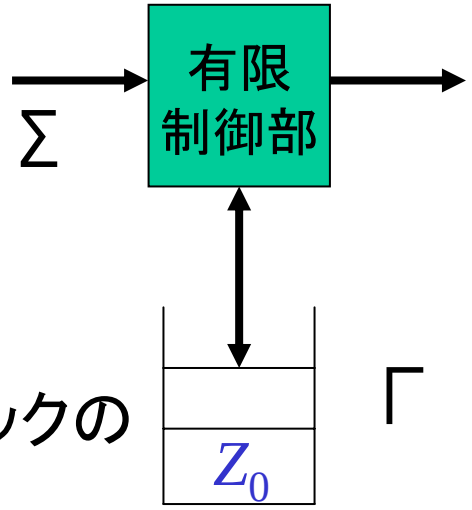


6.1.2. Formal Definition of a PDA

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

- Q: Finite set of states
- Σ : Input alphabet
- q_0: Initial state $q_0 \in Q$
- F: Accepting states $F \subseteq Q$
- Γ : Stack alphabet; set of letters which can be in the stack
- Z_0: Start symbol $Z_0 \in \Gamma$; the stack is initialized by this symbol. (That describes the 'bottom' of the stack.)

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

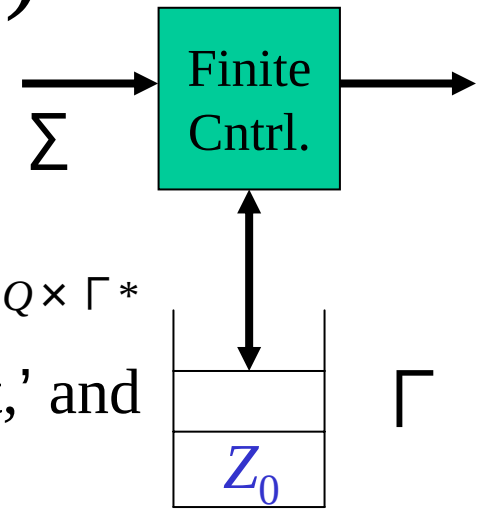
PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

関数 $\delta : Q \times (\Sigma \cup \{ \varepsilon \}) \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$

- 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は右が可:
    - $Y = \varepsilon$; 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. Definition of Pushdown Automaton (PDA)



6.1.2. Formal Definition of a PDA

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

Trans. function $\delta : Q \times (\Sigma \cup \{ \varepsilon \}) \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$

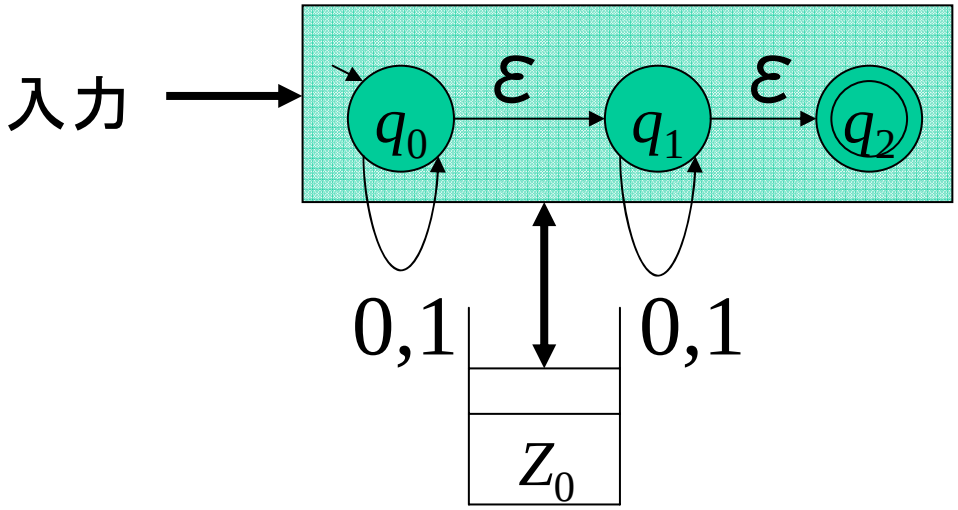
- δ receives 'current state,' 'current input,' and 'the letter X at the top of the state,'
- and returns 'next state' and 'the word Y' which replace X.
  - 'current input' can be ε
  - Y is one of:
    - $Y = \varepsilon$; 'remove X'
    - Y is a letter; 'replace X by Y'
    - Y is a word ; replace + add.
- δ is nondeterministic,

and hence there can be many transitions for one input 3-tuple.

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

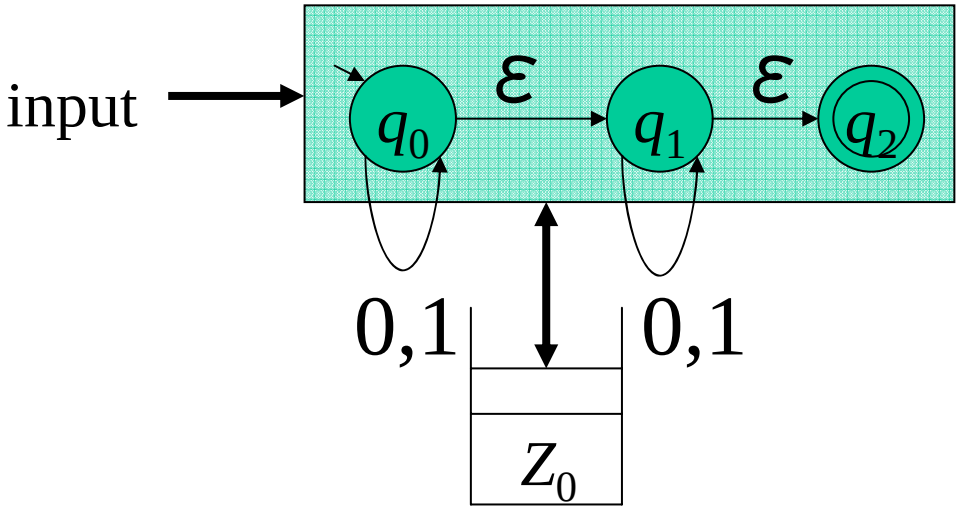


$$P = (\{q_0,q_1,q_2\}, \{0,1\}, \{0,1,Z_0\}, \delta, q_0, Z_0, \{q_2\})$$

6.1. Definition of Pushdown Automaton (PDA)

6.1.2. Formal Definition of a PDA

Ex) PDA P accepting $L = \{ ww^R \mid w \in \{0,1\}^* \}$

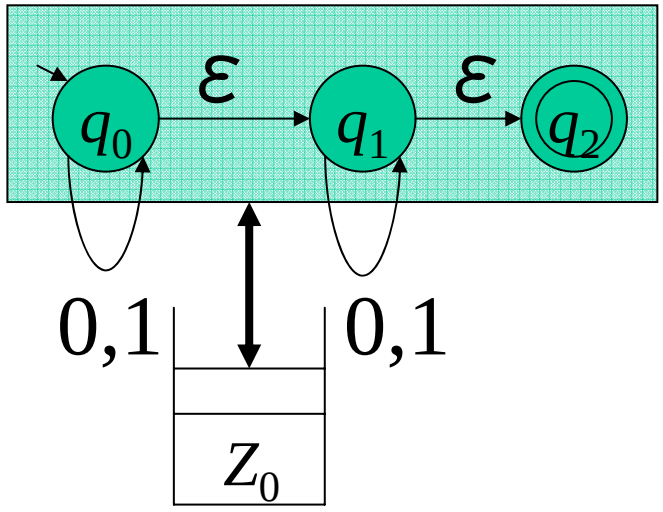


$$P = (\{q_0,q_1,q_2\}, \{0,1\}, \{0,1,Z_0\}, \delta, q_0, Z_0, \{q_2\})$$

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_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

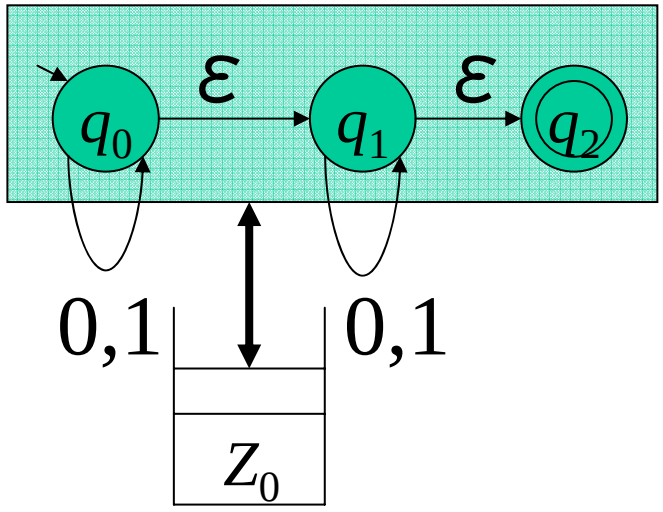
$$\delta(q_0,\varepsilon,Z_0)=\{(q_1,Z_0)\},\delta(q_0,\varepsilon,0)=\{(q_1,0)\},\delta(q_0,\varepsilon,1)=\{(q_1,1)\}$$

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

6.1. Definition of Pushdown Automaton (PDA)

6.1.2. Formal Definition of a PDA

Ex) PDA P accepting $L = \{ ww^R \mid w \in \{0,1\}^* \}$



δ for q_0

- Push the input to the stack

$$\delta(q_0,0,Z_0)=\{(q_0,0Z_0)\},\delta(q_0,1,Z_0)=\{(q_0,1Z_0)\}$$
$$\delta(q_0,0,0)=\{(q_0,00)\},\delta(q_0,1,0)=\{(q_0,10)\}$$
$$\delta(q_0,0,1)=\{(q_0,01)\},\delta(q_0,1,1)=\{(q_0,11)\}$$

- Trans. to q_1 by ε -transition

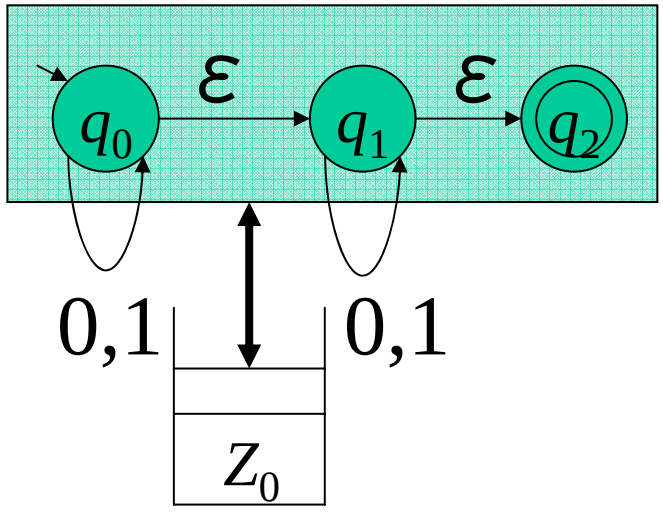
$$\delta(q_0,\varepsilon,Z_0)=\{(q_1,Z_0)\},\delta(q_0,\varepsilon,0)=\{(q_1,0)\},\delta(q_0,\varepsilon,1)=\{(q_1,1)\}$$

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

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

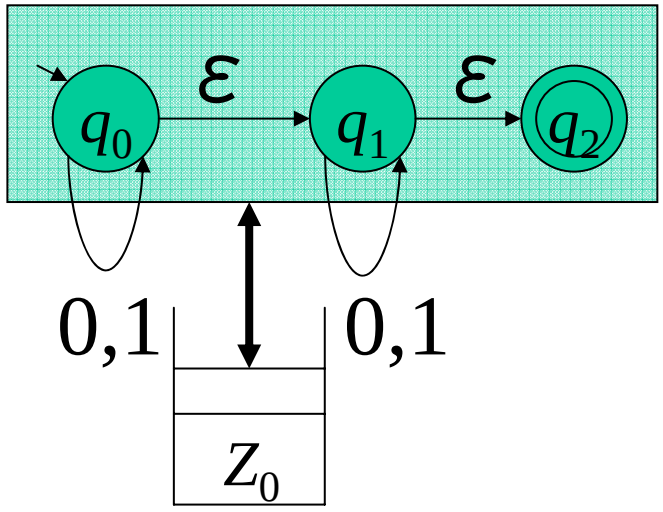
$$\delta(q_1, \varepsilon, Z_0) = \{(q_2, Z_0)\}$$

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

6.1. Definition of Pushdown Automaton (PDA)

6.1.2. Formal Definition of a PDA

Ex) PDA P accepting $L = \{ ww^R \mid w \in \{0,1\}^* \}$



δ for q_1

- Compare input with the top of stack

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

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

- When both of them become empty, trans to q_2 by ε -transition.

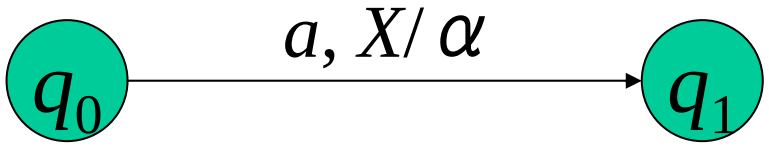
$$\delta(q_1, \varepsilon, Z_0) = \{(q_2, Z_0)\}$$

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

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

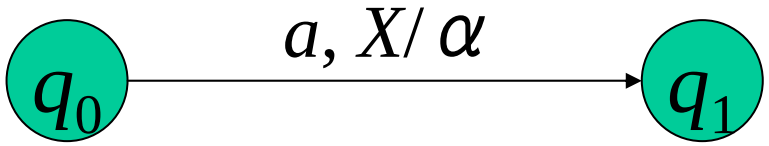


$\delta(q_0, a, X) = \{\ \ldots, (q_1, \alpha\ ), \ldots\}$ の図示

6.1. Definition of Pushdown Automaton (PDA)

6.1.3. Description of PDA by Diagram

- Function δ is described by a label on the edge

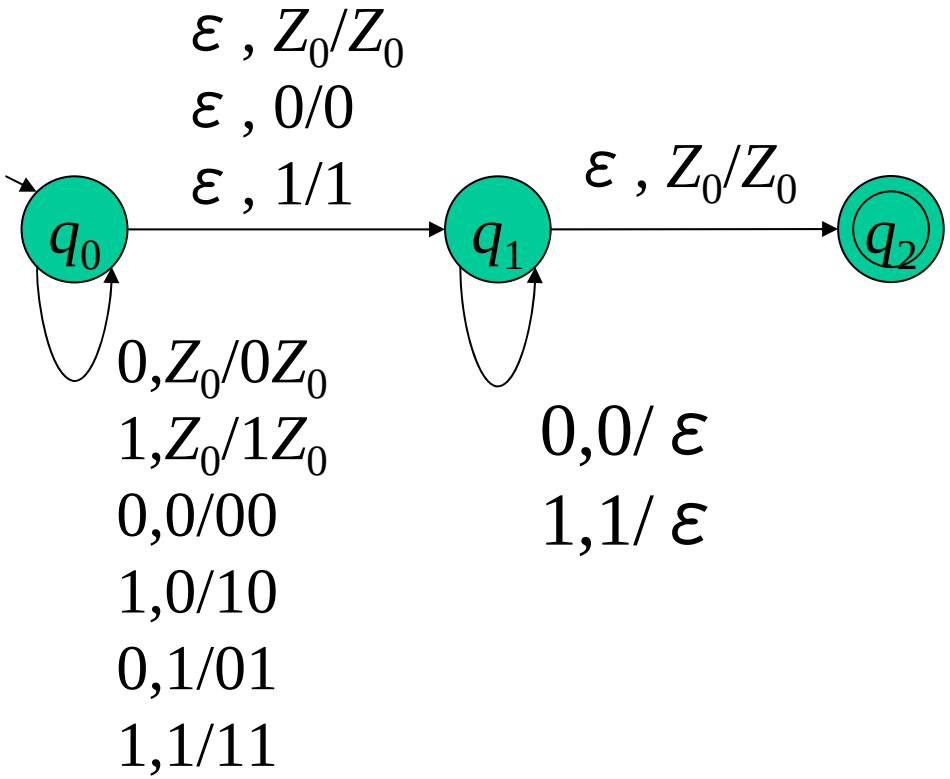


$$\delta(q_0, a, X) = \{\ \ldots, (q_1, \alpha), \ldots\}$$

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

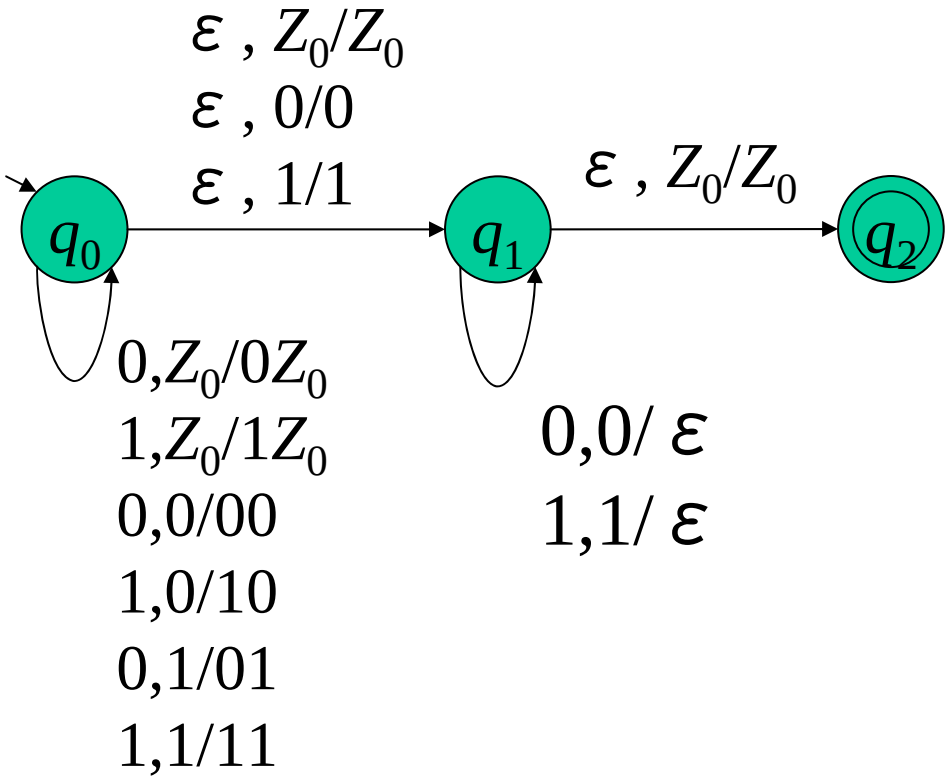
例) $L = \{ ww^R \mid w \in \{0,1\}^* \}$ を受理する PDA P の δ



6.1. Definition of Pushdown Automaton (PDA)

6.1.3. Description of PDA by Diagram

Ex) PDA P accepting $L = \{ ww^R \mid w \in \{0,1\}^* \}$



6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.4. PDAの時点表示

- 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. Definition of Pushdown Automaton (PDA)

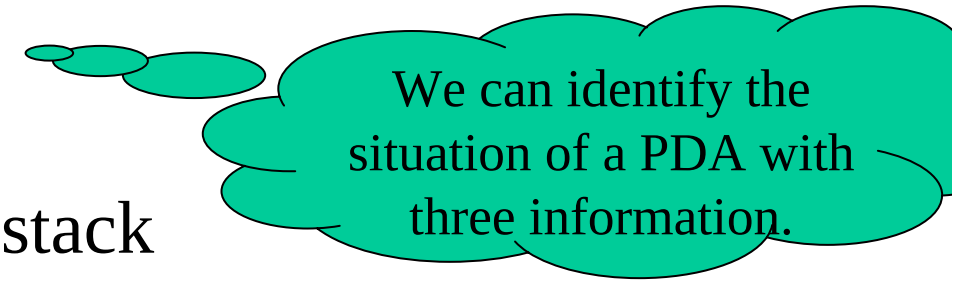
6.1.4. Instantaneous Description of PDA

- Automaton can be identified by a state
- PDA cannot be identified by a state without the string in the stack.

⇒ $\hat{\delta}$ notation is not suitable.

An instantaneous description of a PDA is (q, w, γ), where

- $q \in Q$: state
- $w \in \Sigma^*$: unread input
- $\gamma \in \Gamma^*$: string in the stack



Instantaneous Description is abbreviated by ID.

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.4. PDAの時点表示

- 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

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

PDAの1ステップを表現

と書く(Pがわかっているなら$\vdash$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 $I \vdash^* K$

6.1. Definition of Pushdown Automaton (PDA)

6.1.4. Instantaneous Description of PDA

- For a PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, if $\delta(q,a,X)$ contains (p, α), for $w \in \Sigma^*$, $\beta \in \Gamma^*$, we denote by

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

  …which describes 1 step of the PDA

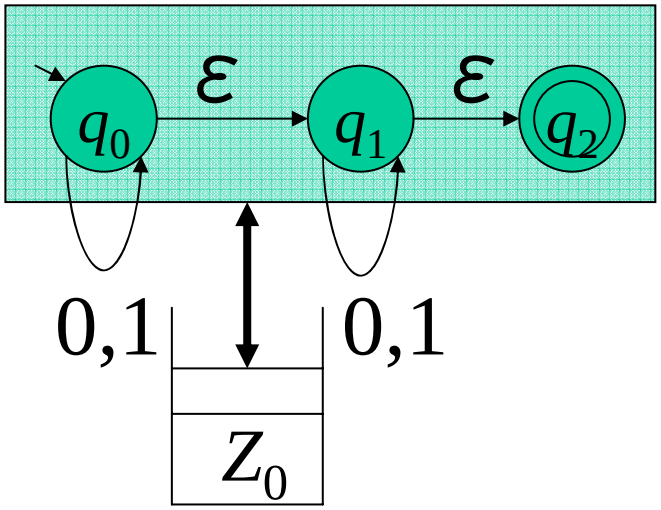
  (P would be omitted if it is clear).

- We also use $\vdash^*$ for general transitions with 0 or more steps:
  1. For any ID I, we denote by $I \vdash^* I$
  2. For IDs I,J,K *with* $I \vdash J$ & $J \vdash^* K$, $I \vdash^* K$

6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.4. PDA 時点表示

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:

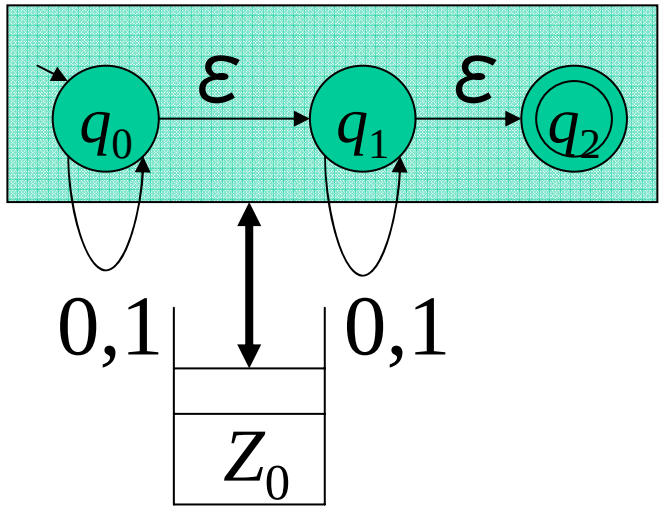
$(q_0, 101101, Z_0) \vdash (q_0, 01101, 1Z_0)$
$(q_0, 101101, Z_0) \vdash (q_1, 101101, Z_0)$

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

6.1. Definition of Pushdown Automaton (PDA)

6.1.4. Instantaneous Description of PDA

Ex) PDA accepting $L = \{ ww^R \mid w \in \{0,1\}^* \}$



Transitions of IDs for input 101101 :

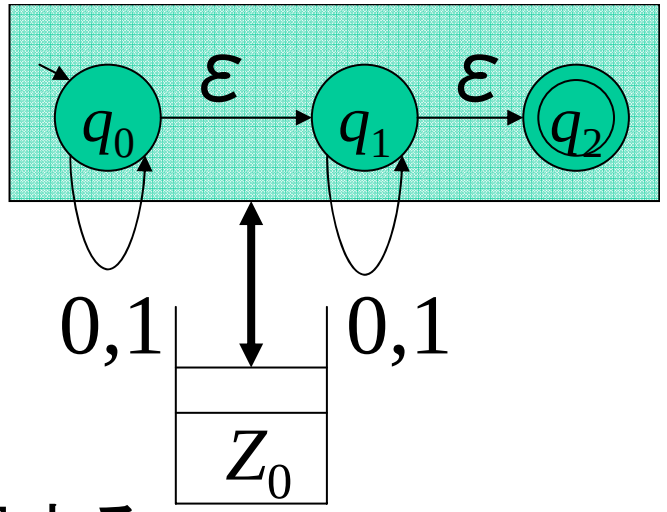
$(q_0, 101101, Z_0) \vdash (q_0, 01101, 1Z_0)$

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

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

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

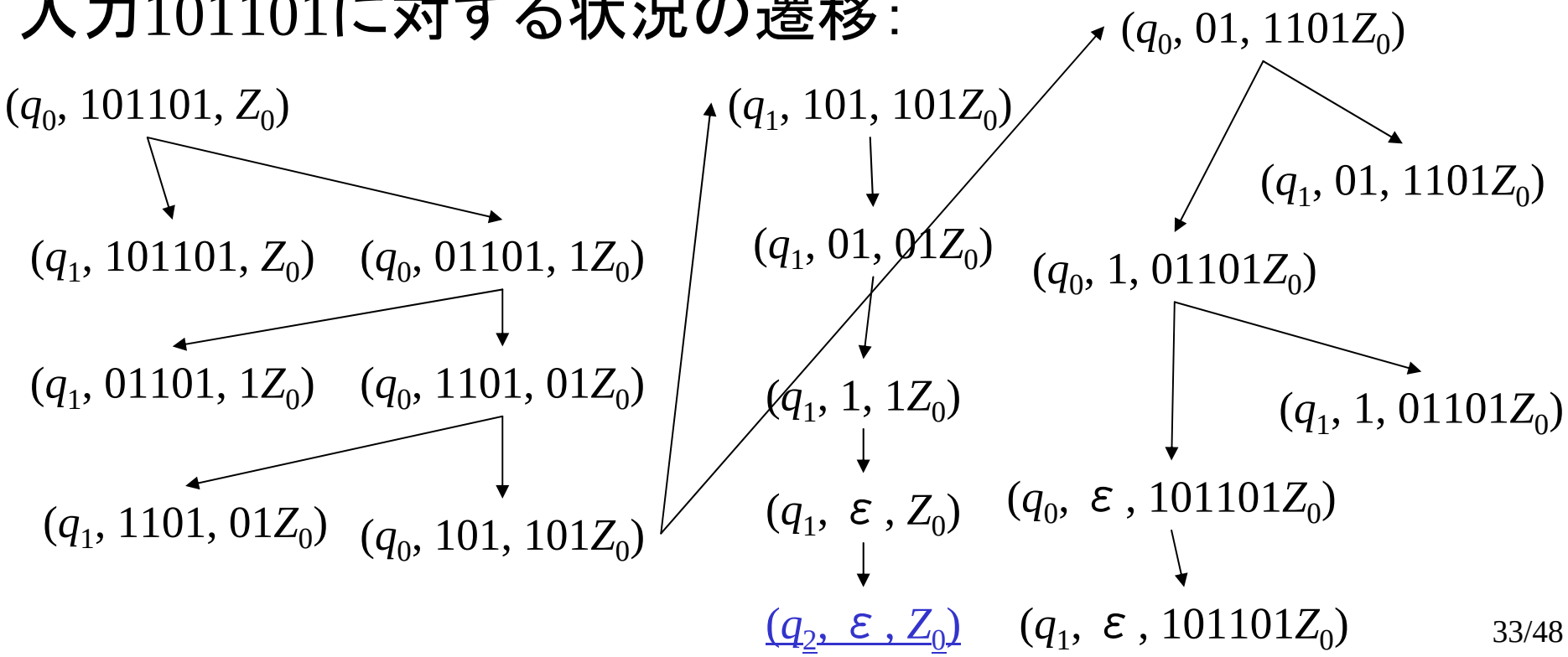
6.1. プッシュダウン・オ
(PDA)の定義



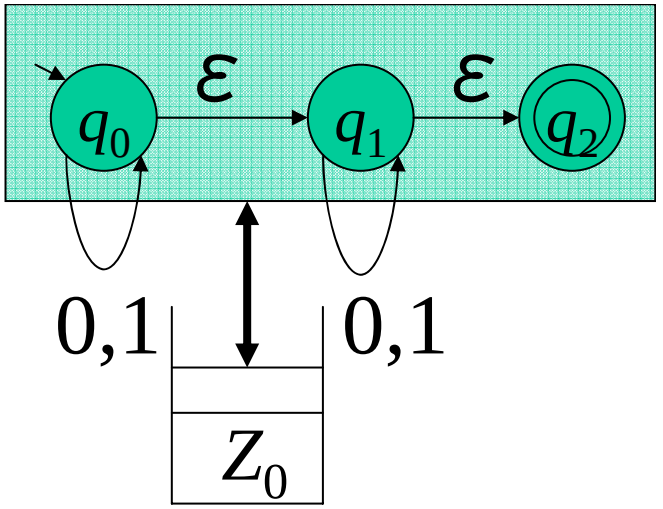
6.1.4. PDA 時点表示

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:



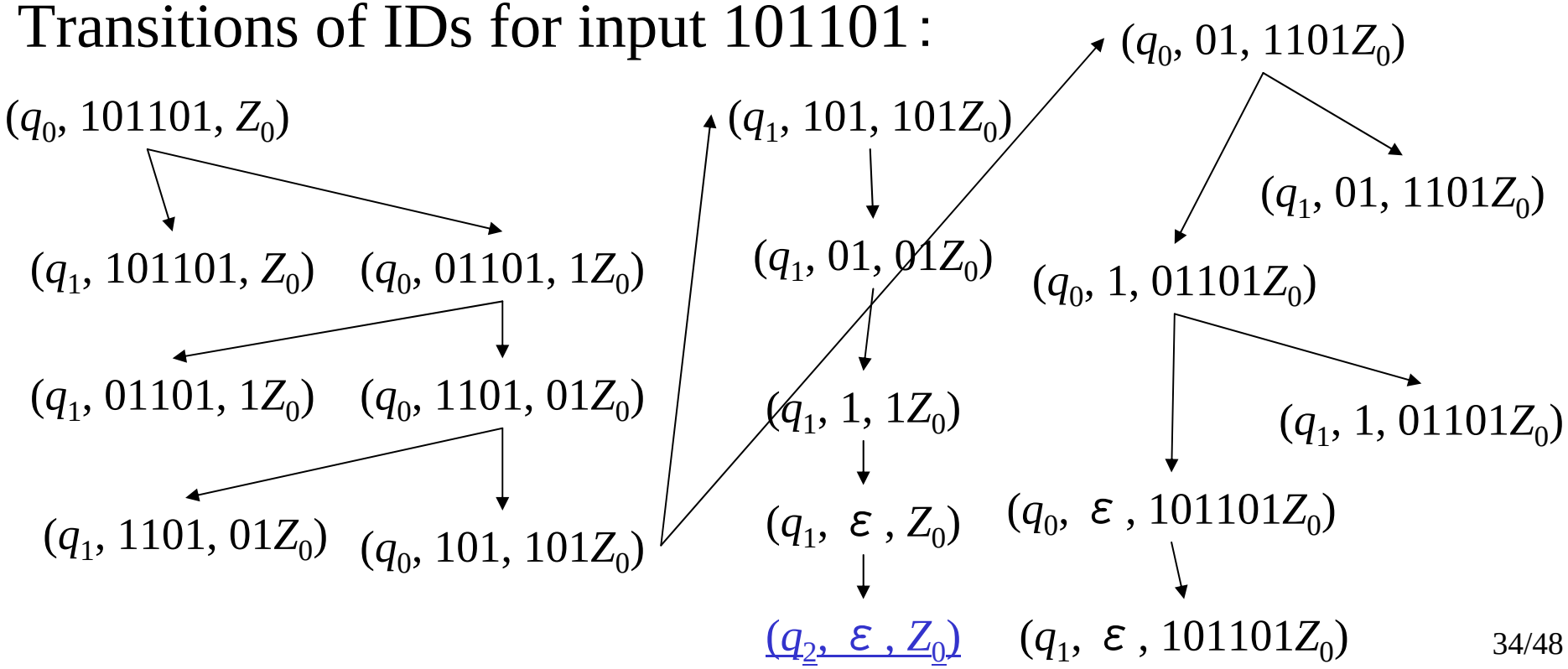
6.1. Definition of Pu Automaton (PD



6.1.4. ID of PDA

Ex) PDA accepting $L = \{ ww^R \mid w \in \{0,1\}^* \}$

Transitions of IDs for input 101101 :



6.1. プッシュダウン・オートマトン (PDA)の定義

6.1.4. 時点表示

[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 $(q,xw,\alpha\ \gamma) \vdash^* (p,yw,\beta\ \gamma)$

[定理] PDA P の計算プロセスを考えると、

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

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

6.1. Definition of Pushdown Automaton (PDA)

6.1.4. Instantaneous Description of PDA

[Theorem] By the process of transitions of a PDA P,

$(q,x,\alpha) \vdash^* (p,y,\beta)$ implies $(q,xw,\alpha\ \gamma) \vdash^* (p,yw,\beta\ \gamma)$

[Theorem] By the process of transitions of a PDA P,

$(q,xw,\alpha) \vdash^* (p,yw,\beta)$ implies $(q,x,\alpha\ \gamma) \vdash^* (p,y,\beta\ \gamma)$

★ '$(q,x,\alpha\ \gamma) \vdash^* (p,y,\beta\ \gamma)$ implies $(q,x,\alpha) \vdash^* (p,y,\beta)$' is not true!

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. Language by PDA

When input ends, PDA accepts if…:

1. it is in an accepting state
2. its stack is empty

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

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

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

- When we consider $N(P)$, F is ignored, and hence we sometimes denote the PDA by 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, Z_0)$.
- On p.258 in the textbook in Japanese edition, $N(P)$ is defined by $q \in F$, which should by a typo.

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

- P を固定すると一般に $L(P) \neq N(P)$

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. Language by PDA

Two languages defined by a PDA P:

1. $L(P)$: it is in accepting state if input ends.
2. $N(P)$: it has an empty stack if input ends.

- For some fixed P, $L(P) \neq N(P)$

Ex) For P such that Z_0 is always at the bottom of the stack, $N(P) = \Phi$

[Theorem]

For any PDA Q, there is a PDA P with $N(Q) = L(P)$.

For any PDA P, there is a PDA Q with $L(P) = N(Q)$.

6.2. PDAの言語

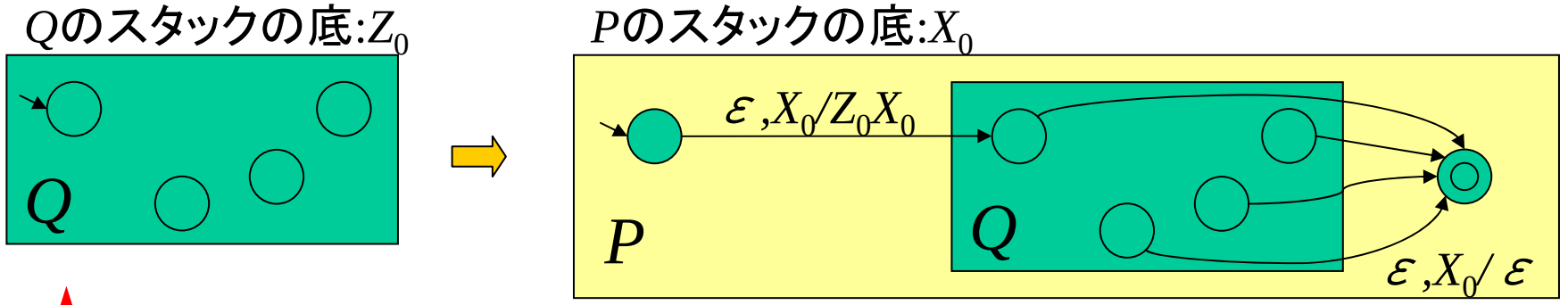
[定理]

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. Language by PDA

[Theorem]

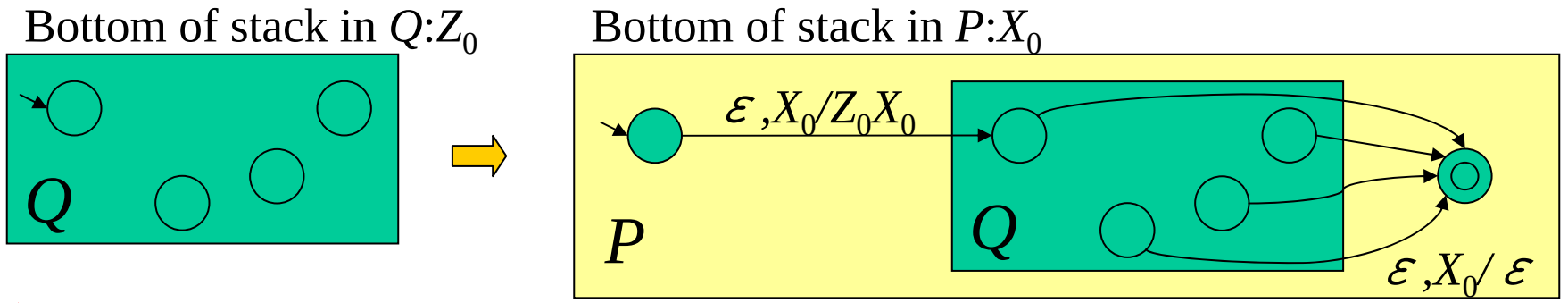
For any PDA Q, there is a PDA P with $N(Q) = L(P)$.

For any PDA P, there is a PDA Q with $L(P) = N(Q)$.

[Proof(Sketch)] For given Q, we construct P that satisfies the condition as follows:

Idea: First, push the special symbol X_0 at the bottom of the stack…

stack is empty in Q = X_0 is the top of the stack in P



★ when input ends, Q has empty stack if and only if P is accepting state.

6.2. PDAの言語

[定理]

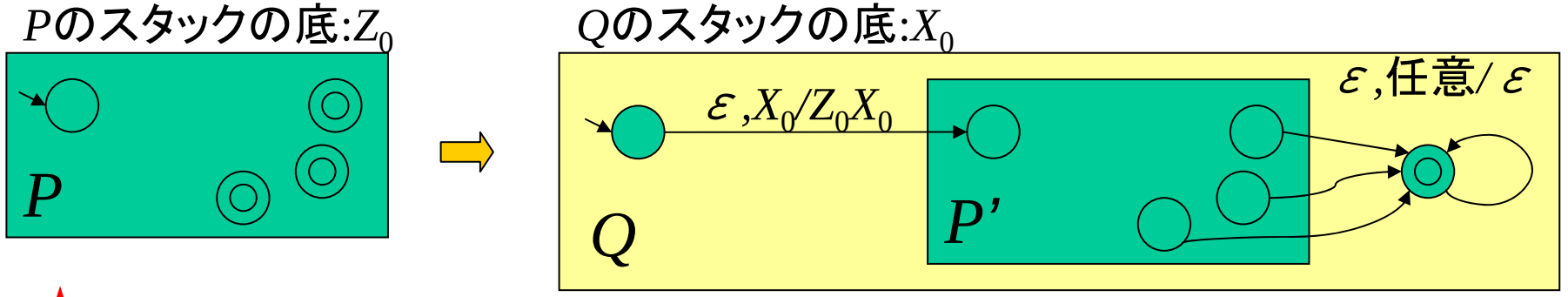
1. PDA Q に対して $N(Q) = L(P)$ を満たす PDA P が存在。
2. PDA P に対して $L(P) = N(Q)$ を満たす PDA Q が存在。

[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

6.2. Language by PDA

[Theorem]

For any PDA Q, there is a PDA P with $N(Q) = L(P)$.

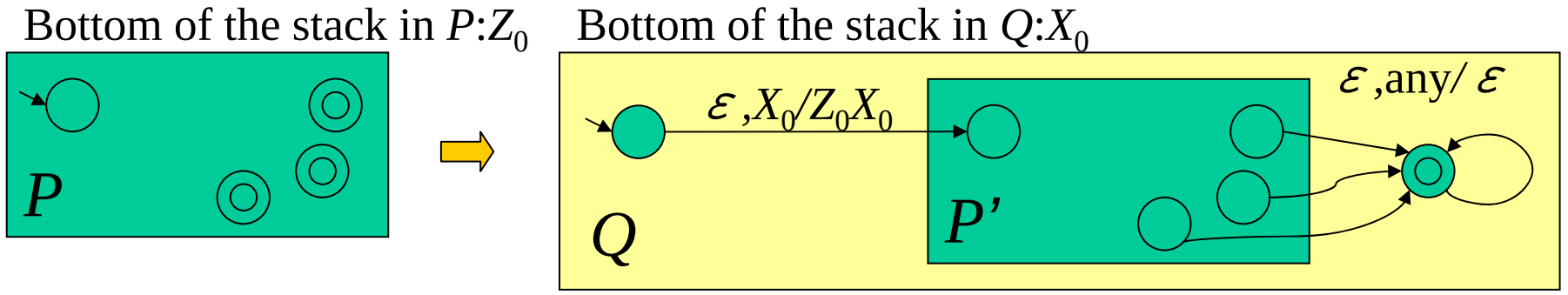
For any PDA P, there is a PDA Q with $L(P) = N(Q)$.

[Proof(Sketch)] For given P, we construct Q that satisfies the condition as follows:

Idea 1: First, push the special symbol X_0 at the bottom of stack … The stack never become empty while the simulation of P.

Idea 2: From an accepting state of P, Q transits to accepting state with ε -transition…remark that no input is consumed.

Idea 3: In the accepting state, Q gets rid of all contents of the stack.



★When input ends, P is accepting state if and only if Q has empty stack.

6.3. PDA とCFGの等価性

[結論]

PDAで受理できる言語=CFGで生成できる言語

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。

証明は省略
(難しくはないが、複雑)

6.3. Equivalence of PDA and CFG

[Fact]

The class of languages accepted by a PDA = the class of languages generated by a CFG

Theorem

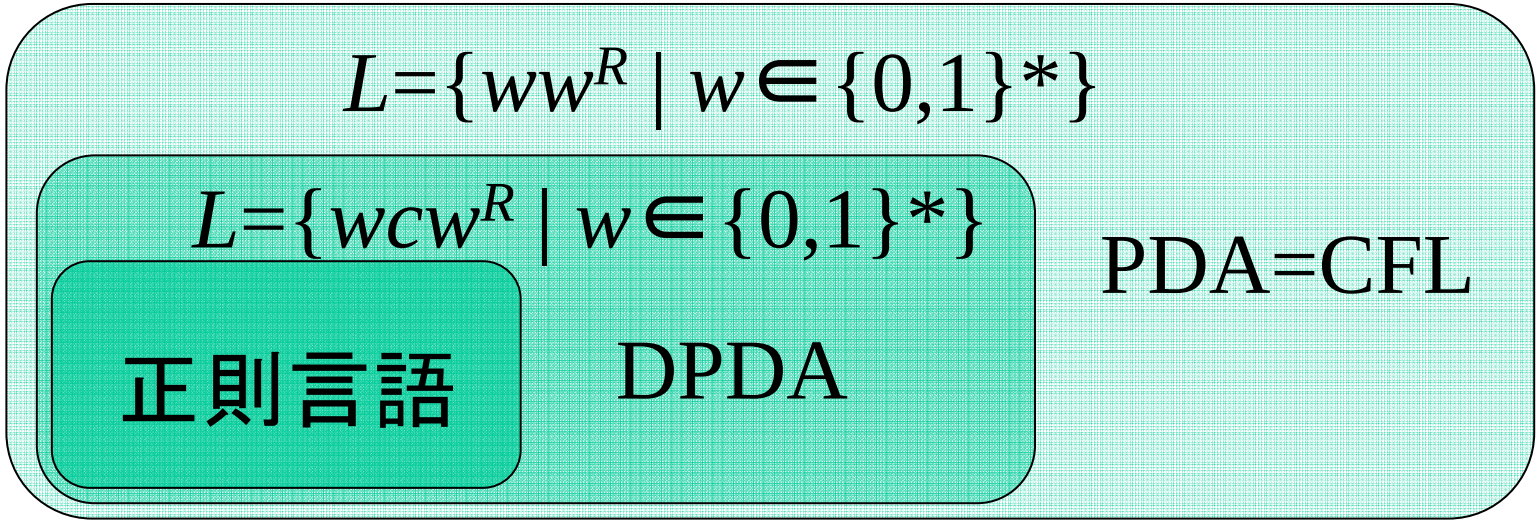
1. For any CFG G, there is a PDA P such that $L(G)=N(P)$.
2. For any PDA P, there is a CFG G such that $N(P)=L(G)$.

Proof is omitted.
(Not difficult, but complicated.)

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:

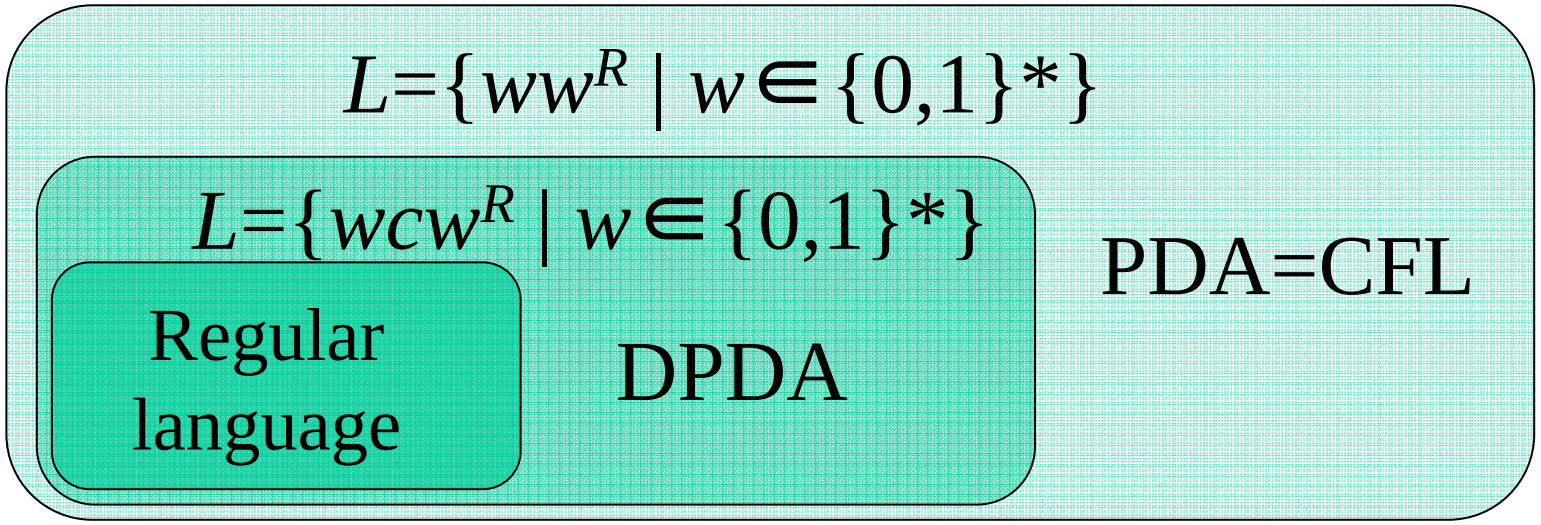


★DPDAのクラスは実用的クラス(例;LR(k),YACC)

6.4. Deterministic PDA (Summary)

Deterministic PDA: PDA whose transition is deterministic.
Namely, 'next state' is always determined uniquely.

Diagram for regular language, language by DPDA, and language by PDA:



★The class of languages accepted by DPDA is practically used. (ex.;LR(k),YACC)