計算理論　第10回
第6章：
プッシュダウンオートマトン (1/2)

基礎工学部情報科学科
中川　博之

本日の概要

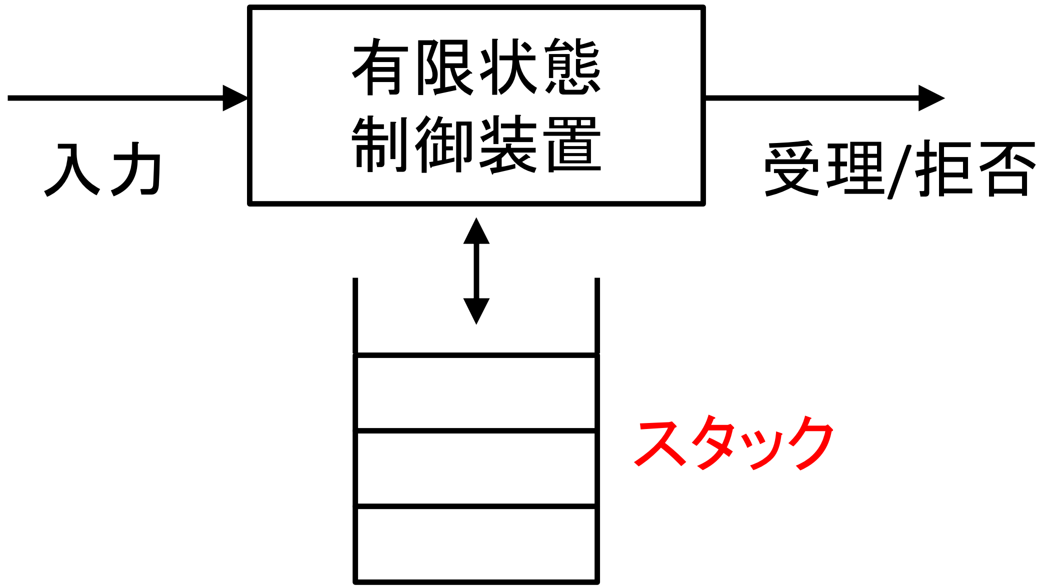
- 第6章：プッシュダウンオートマトン
  - テキスト：p.245～
  - 6.1 プッシュダウンオートマトンの定義
  - 6.2 PDAの言語

- 重要概念
  - 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の定義

直観的な導入

- プッシュダウンオートマトン (PDA):
  - 文脈自由言語を受理するのに適切なオートマトン
  - 「ε-動作可能な非決定性有限オートマトン」に「スタック」を追加したもの
    - スタックにはFILO方式でしかアクセスできない



PDAの特徴

- 文脈自由言語を認識できる
  - 有限オートマトンより強力
- 文脈自由言語より複雑な言語は認識できない
  - 例: $L = \{0^n 1^n 2^n \mid n \geq 1\}$

- PDAが認識する言語のクラス
  = 文脈自由言語(CFL)

PDAの定義

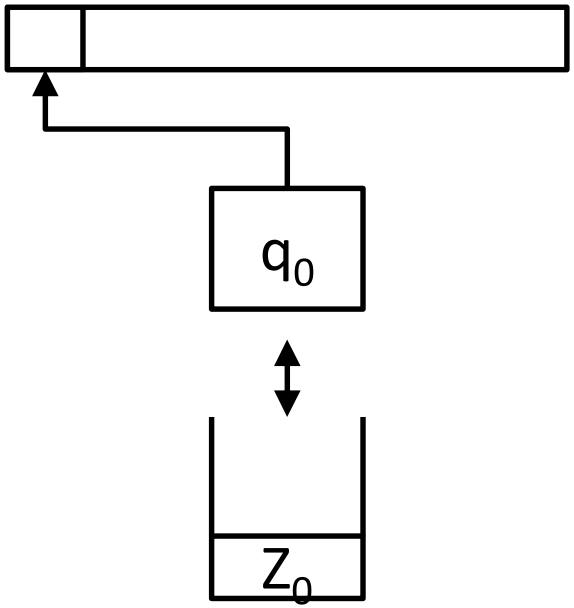
- PDA P = $(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$
  - Q : 状態の有限集合
  - Σ : 入力記号の有限集合
  - Γ : 有限のスタックアルファベット (スタック記号)
    - スタックに記録できる記号の有限集合
  - δ : 遷移関数
  - q_0 : 初期状態 ($\in Q$)
  - Z_0 : 開始記号
    - スタックにこの記号が1つ置かれた状態で動作開始
  - F : 受理状態（最終状態とも呼ぶ）の集合 ($\subseteq Q$)

PDAの遷移関数δ

- $(p, \gamma) \in \delta(q, a, X)$
  - PDAの遷移関数は3入力2出力
  - (集合の)要素記述なのは非決定を許しているため
- 入力
  - $q \in Q$: 現在の状態
  - $a \in (\Sigma \cup \{\varepsilon\})$: 入力記号またはε
  - $X \in \Gamma$: スタック上端のスタック記号
- 出力
  - $p \in Q$: 次の状態
  - $\gamma \in \Gamma^*$: スタック記号の列

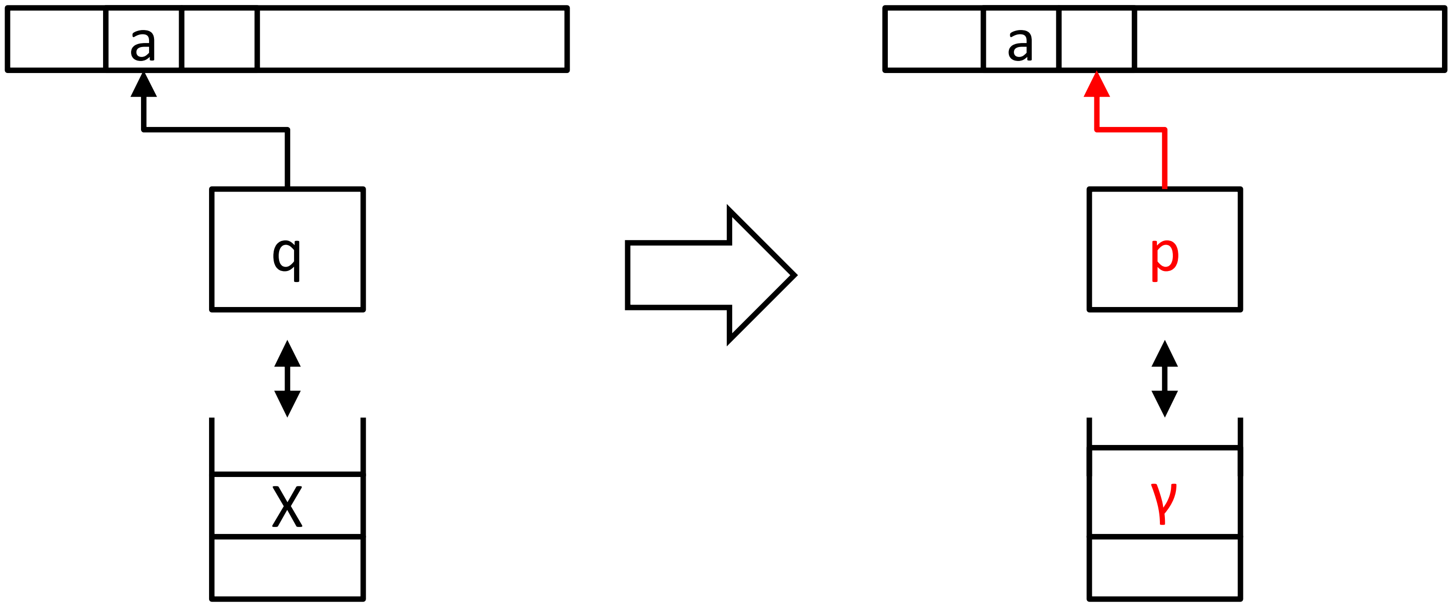
PDAの振舞い

- 初期状態
  - 状態 : q_0
  - スタック : Z_0



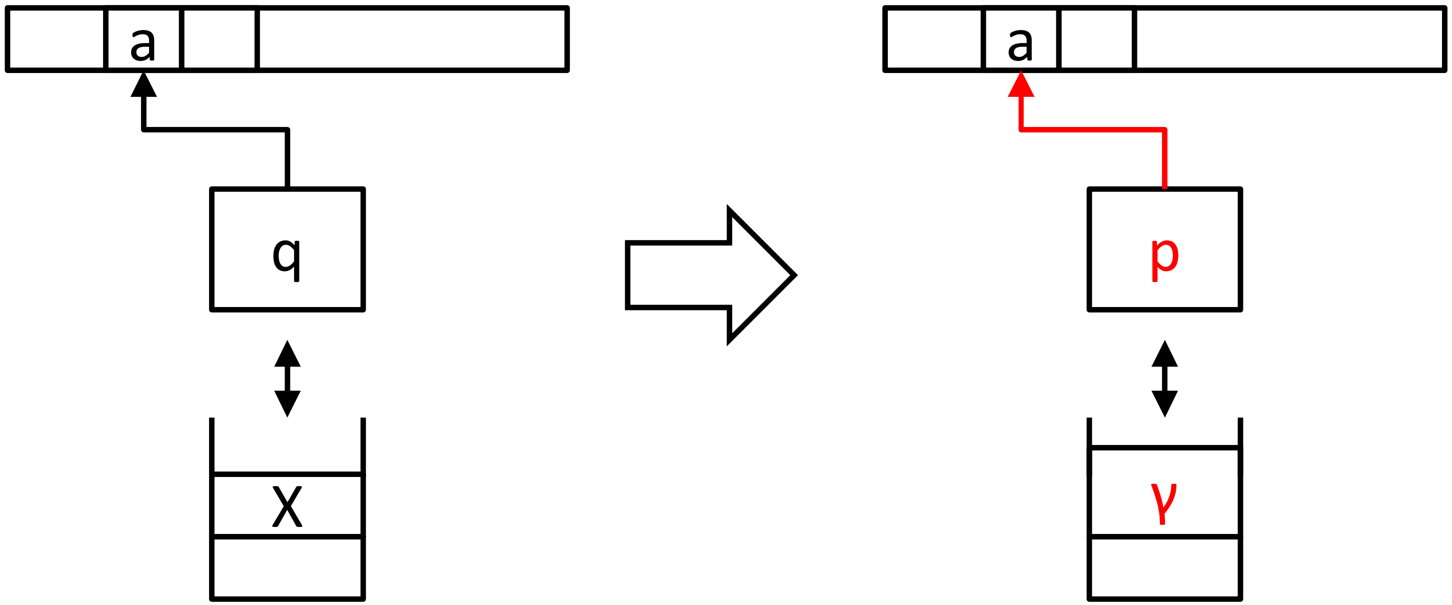
PDAの振舞い：状態遷移(1)

- 遷移：(p, γ)∈δ(q, a, X)　ただしa ≠ ε
  - 状態：q, 入力：a, スタックの上端X のとき
  - 次の状態をpにする
  - 入力記号をひとつ読み進める
  - スタックからXをpopして，γをpush



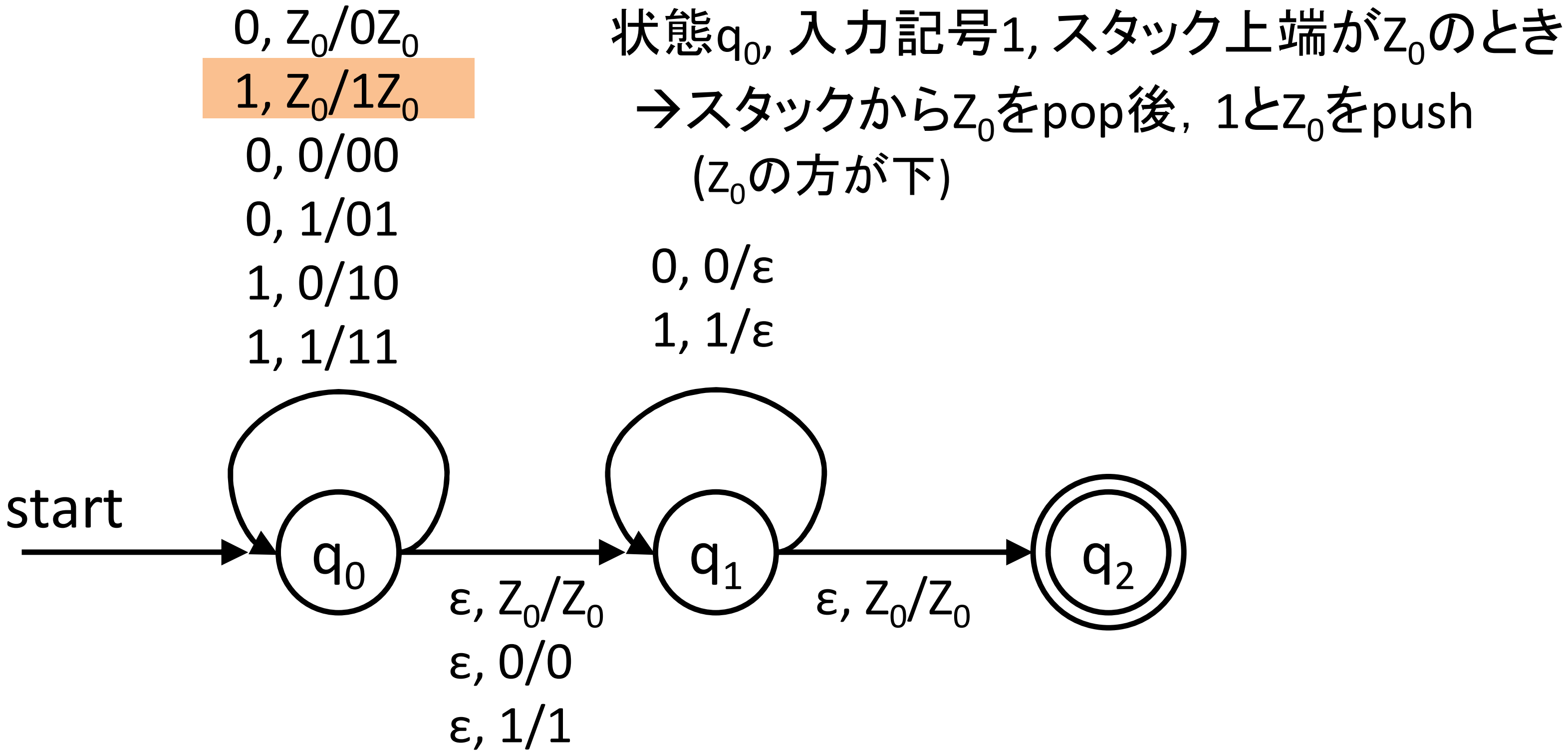
PDAの振舞い：状態遷移(2)

- 遷移：$(p, \gamma) \in \delta(q, \varepsilon, X)$
  - 状態：q, スタックの上端Xのとき
  - 次の状態をpにする
  - 入力記号を読み進めない
  - スタックからXをpopして，γをpush



PDAの図表現

- 各状態を円で表現，受理状態は2重円
- 状態qからpへの辺のラベルが “a, X/γ”
  → 遷移 $(p, \gamma) \in \delta(q, a, X)$ を表す



PDAの設計例（例6.1, 例6.2）

- 言語 L_{wwr} = {ww^R | wは(0+1)*の中の列}
- PDAの設計方針
  - w中の記号を読んでいる間はスタックにプッシュ
  - w^{R}の記号を読むたびにスタック上端の記号と比較
    - 一致すればその記号を取り除く
    - 入力記号をすべて読み終えたら受理状態に移動
  - 受理状態を含めて3つの状態を用意
- 気になる点
  - どこからw^{R}に移るのかが分からない

    →非決定性なので気にしなくて良い

PDAの例：詳細設計

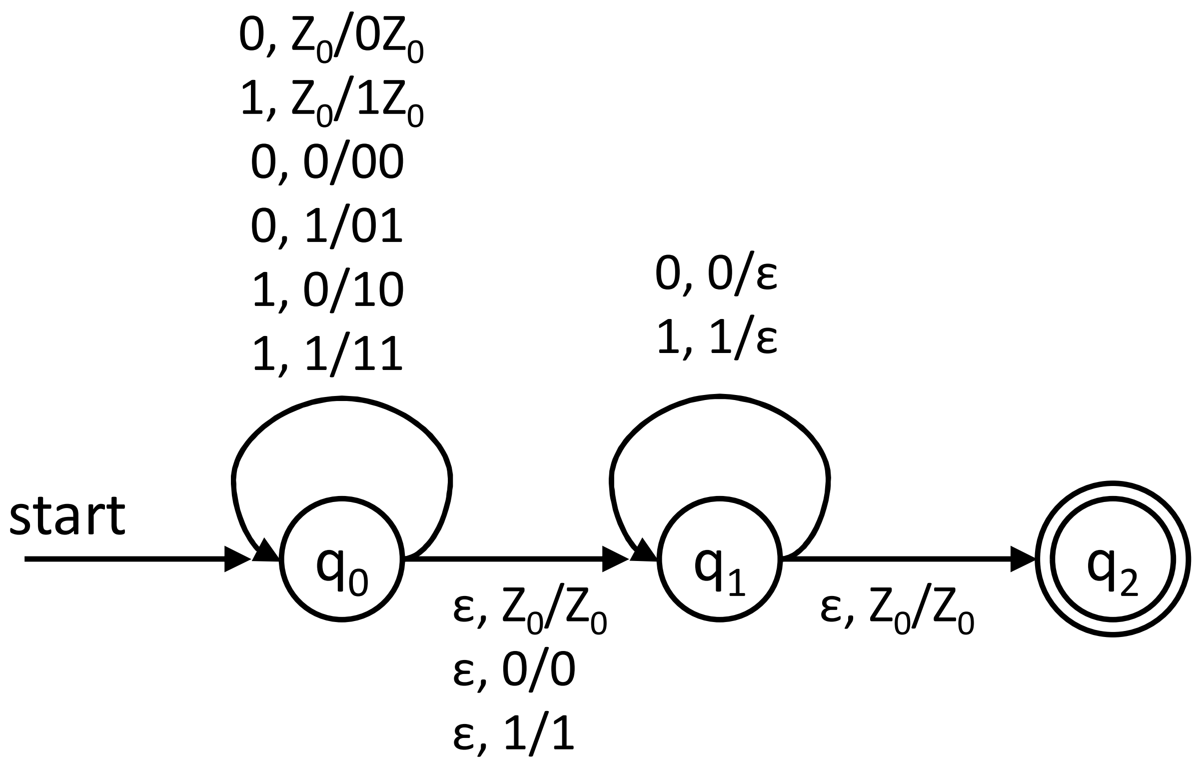
- スタック記号Z_0：スタックの下端を表現
  - 入力を読み終えた時点で，何も残っていないと受理状態に移動できないため
- $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$
  - $Q = \{q_0, q_1, q_2\}$
    - q_0：スタックプッシュ，q_1：照合，q_2：受理状態
  - $\Sigma = \{0, 1\}$
  - $\Gamma = \{0, 1, Z_0\}$
  - $F = \{q_2\}$

PDAの例：詳細設計

- δは以下の通り
  - $\delta(q_0, 0, Z_0)=\{(q_0, 0Z_0)\}$, $\delta(q_0, 1, Z_0)=\{(q_0, 1Z_0)\}$
  - $\delta(q_0, 0, 0)=\{(q_0, \boxed{})\}$, $\delta(q_0, 0, 1)=\{(q_0, \boxed{})\}$,
    $\delta(q_0, 1, 0)=\{(q_0, \boxed{})\}$, $\delta(q_0, 1, 1)=\{(q_0, \boxed{})\}$
  - $\delta(q_0, \varepsilon, Z_0)=\{(q_1, \boxed{})\}$, $\delta(q_0, \varepsilon, 0)=\{(q_1, \boxed{})\}$,
    $\delta(q_0, \varepsilon, 1) = \{(q_1, \boxed{})\}$
  - $\delta(q_1, 0, 0)=\{(q_1, \boxed{})\}$, $\delta(q_1, 1, 1)=\{(q_1, \boxed{})\}$
  - $\delta(q_1, \varepsilon, Z_0)=\{(q_2, \boxed{})\}$

言語 L_{wwr}を受理するPDA

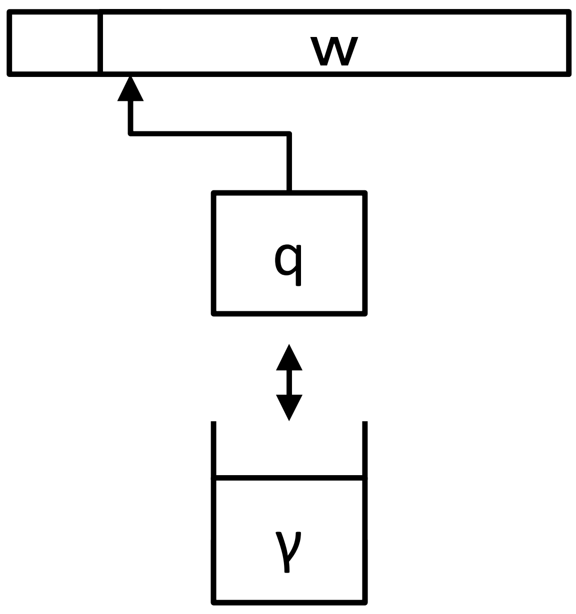
- 言語 L_{wwr} = {ww^R | wは(0+1)*の中の列}



※<u>受理状態でスタックにZ_0が残るが問題ない</u>

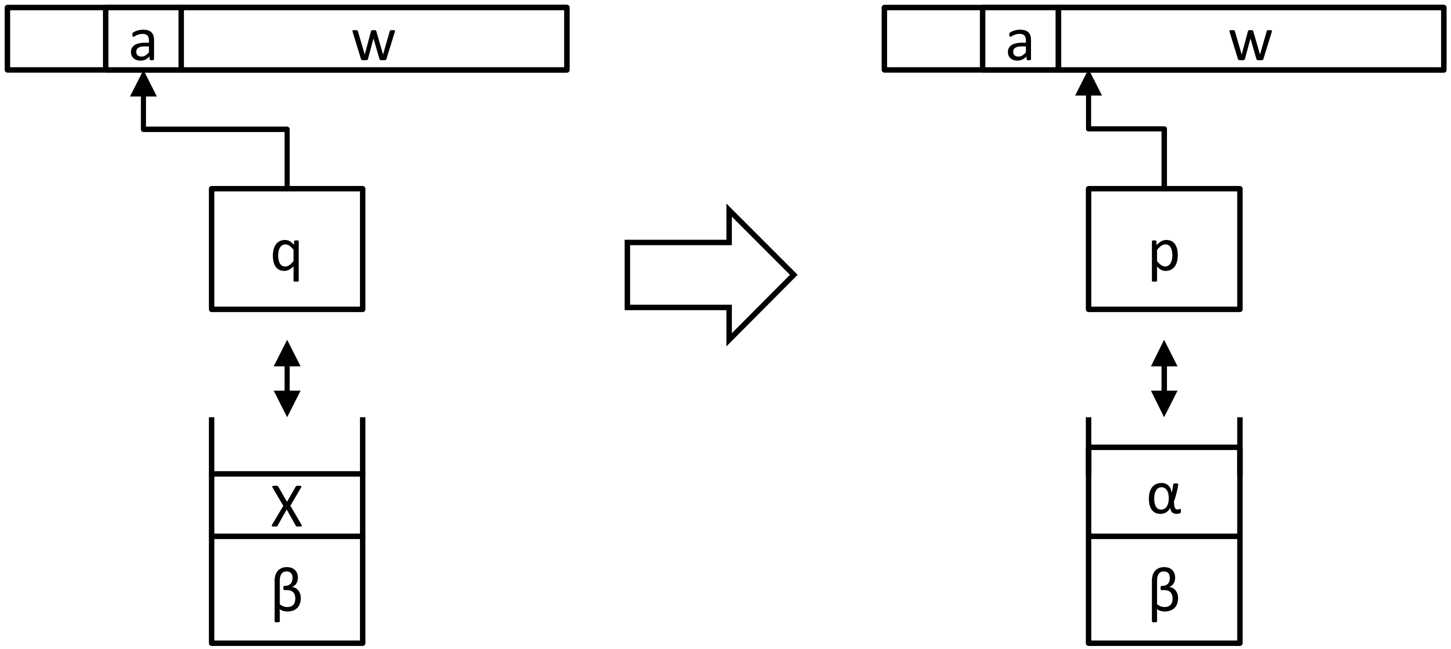
PDAの時点表示

- 時点表示:現在の状態をもれなく表現したもの
- 3つ組で表現:(q, w, γ)
  - q:現在の状態
  - w:入力の残り
  - γ:スタックの内容



時点表示の変化の関係

- $(q, aw, X\beta) \vdash_P (p, w, \alpha\beta)$
  - $(p, \alpha) \in \delta(q, a, X)$に対する遷移
    - aはεの場合もある
  - Pが文脈から明らかなとき，$\vdash_P$ を$\vdash$と表記する



関係$\vdash_P{}^*$

テキストでは $\overset{*}{\underset{P}{\vdash}}$

- $\vdash_P{}^*$: PDAの0回以上の動作を表現
  - Pが文脈から明らかなとき, $\vdash_P{}^*$ を$\vdash^*$と表記する

I, J, Kを時点表示とするとき

- 基礎: 任意の I に対して, $I \vdash^* I$
- 帰納: $I \vdash^* K$ かつ $K \vdash^* J$ ならば $I \vdash^* J$

6.2 PDAの言語

2種類の受理

- PDAには2通りの受理がある
  - 入力を最後まで読み取って受理状態に入る
    →「最終状態による受理」
  - スタックを空にするような入力列
    →「空スタックによる受理」
- この2通りの受理は同等
  → 都合の良い方を用いれば良い

最終状態による受理

- PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$
- Pが最終状態で受理する言語 L(P)

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

- 解釈
  - 初期状態から0回以上の遷移により，入力をすべて読み終えた時点で受理状態に至る
  - そのときのスタックの内容αは何でも良い

空スタックによる受理

- PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0)$
- Pが空スタックで受理する言語 N(P)

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

- 解釈
  - 入力をすべて読み終えた時点でスタックが空になる
  - そのときの状態qは何でも良い
  - 教科書の定義は誤り (Fは不要)

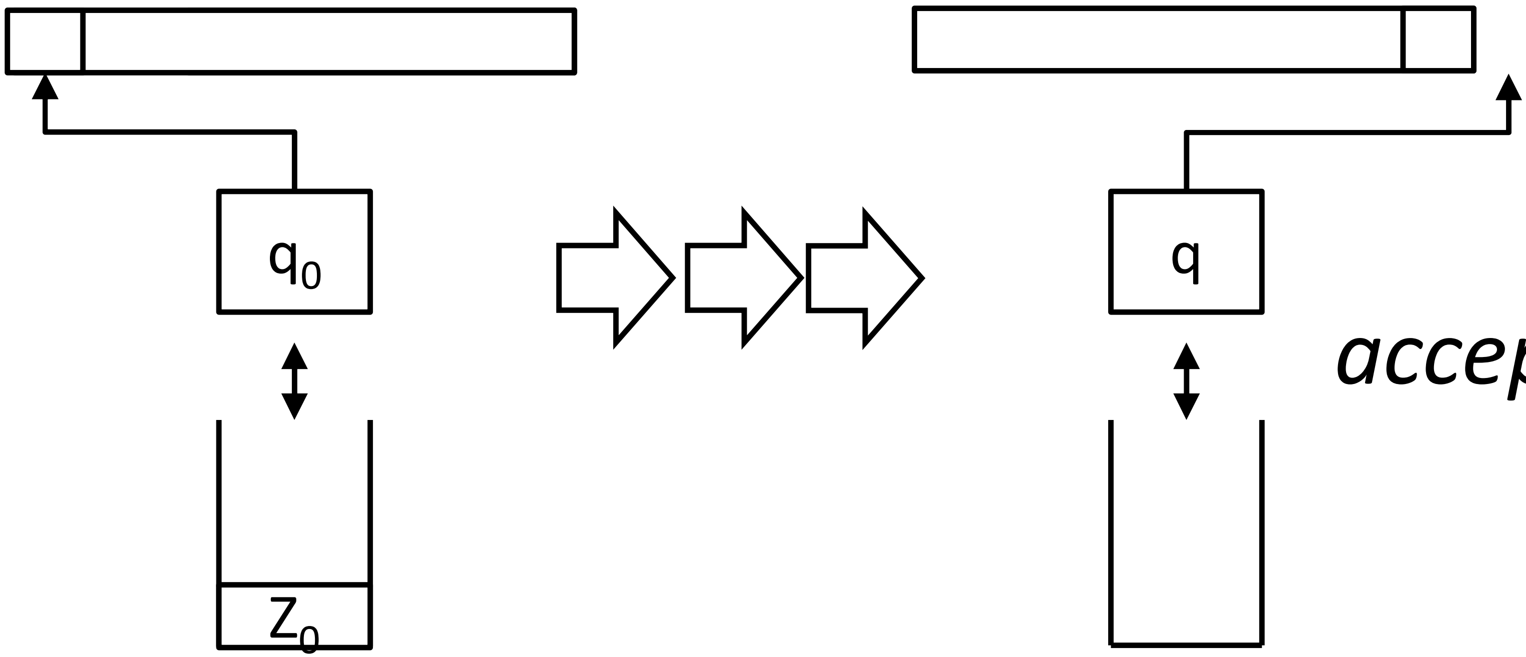
空スタックから最終状態へ

- 空スタック受理PDA ならば 最終状態受理PDA を示す
- 任意の空スタック受理PDA
  $P_N = (Q, \Sigma, \Gamma, \delta, q_0, Z_0)$
  – Fは定義していない
- P_Nの動作を模倣する最終状態受理PDA P_Fを作る
  – 任意のP_Nに対し $N(P_N)=L(P_F)$となるP_Fが存在
  – 同じ言語を受理

まずP_Nの動作を観察

- $P_N = (Q, \Sigma, \Gamma, \delta, q_0, Z_0)$
  - 動作開始時：状態 q_0，スタック Z_0
  - 受理時：スタック 空

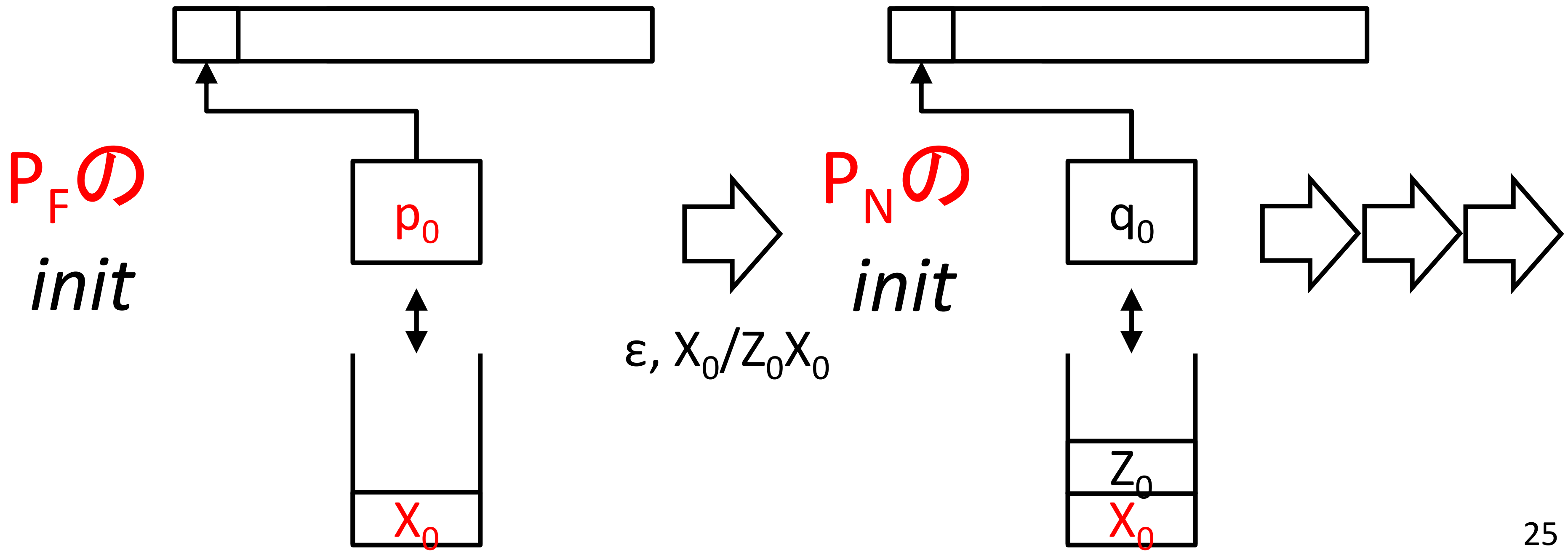
init



accept

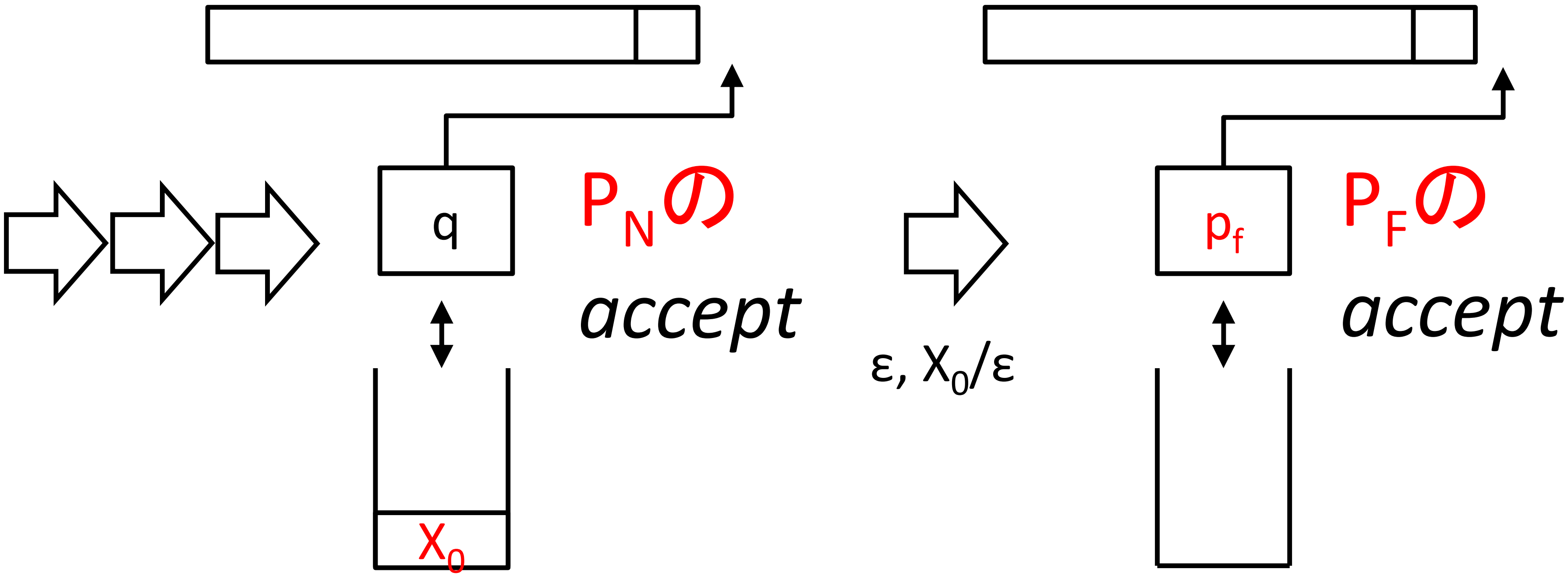
P_NからP_Fへの変換

- 最初にスタックのZ_0の下にX_0を置いておく
  - 空スタックを検知するため
- その後はP_Nの動作を模倣
  - P_Nがスタック内のX_0にアクセスすることはない



P_NからP_Fへの変換

- スタック上端がX_0になると受理状態p_fに遷移
  - 空スタックを検知するため
- $P_F = (Q \cup \{p_0, p_f\}, \Sigma, \Gamma \cup \{X_0\}, \delta_F, p_0, X_0, \{p_f\})$



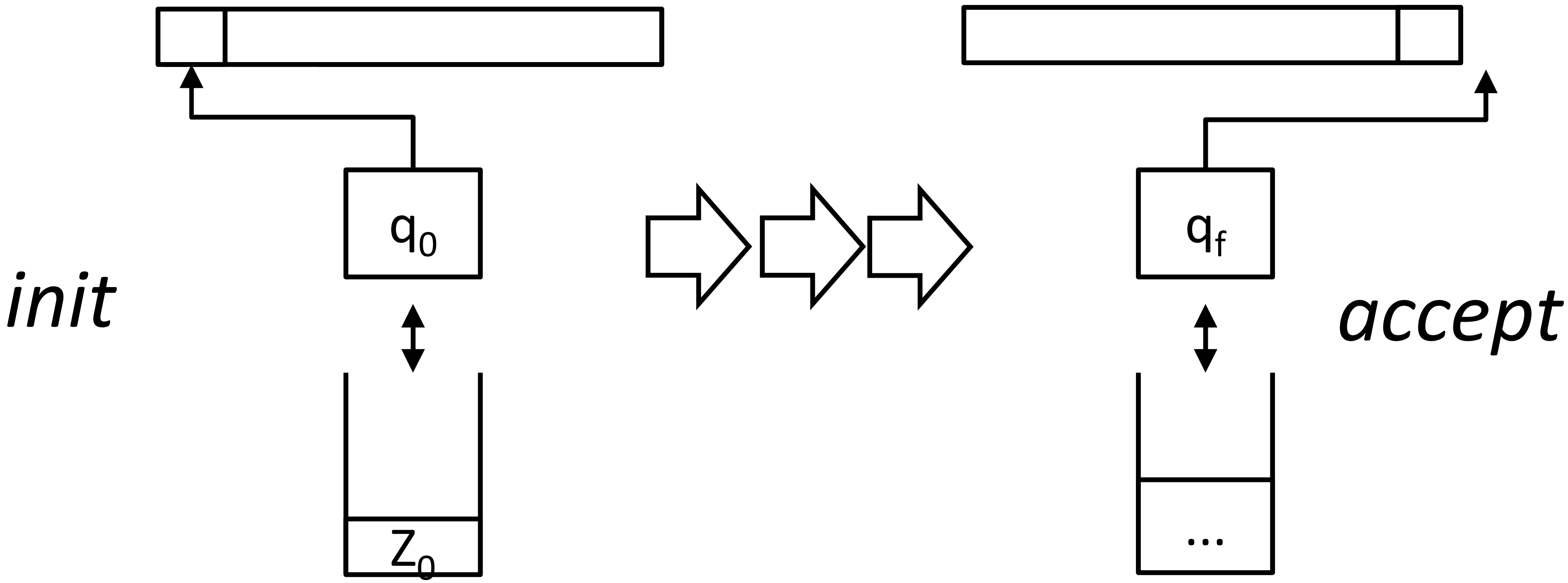
最終状態から空スタックへ

- 最終状態受理PDA ならば 空スタック受理PDAを示す
- 任意の最終状態受理PDA

  $P_F = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$
- P_Fの動作を模倣する空スタック受理PDA P_Nを作る
  - 任意のP_Fに対し $L(P_F)=N(P_N)$ となるP_Nが存在
  - 同じ言語を受理

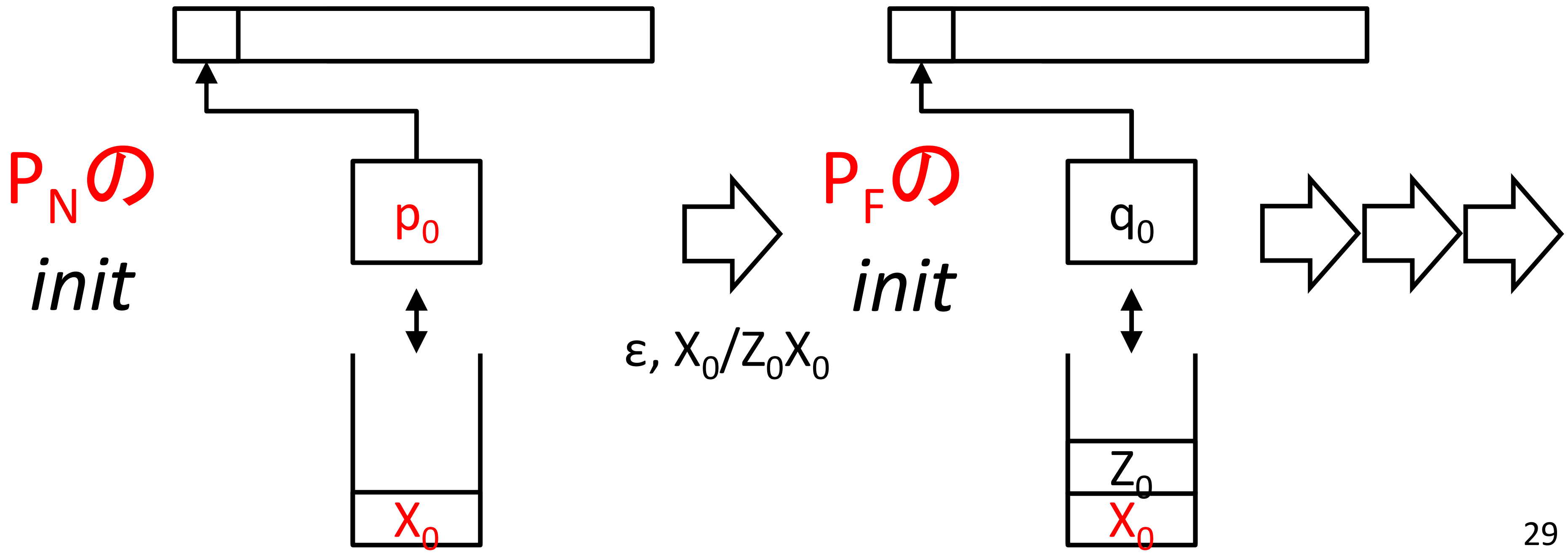
まずP_Fの動作を観察

- $P_F = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$
  - 動作開始時：状態 q_0，スタック Z_0
  - 受理時：状態 $q_f \in F$
  - スタックは空とは限らない



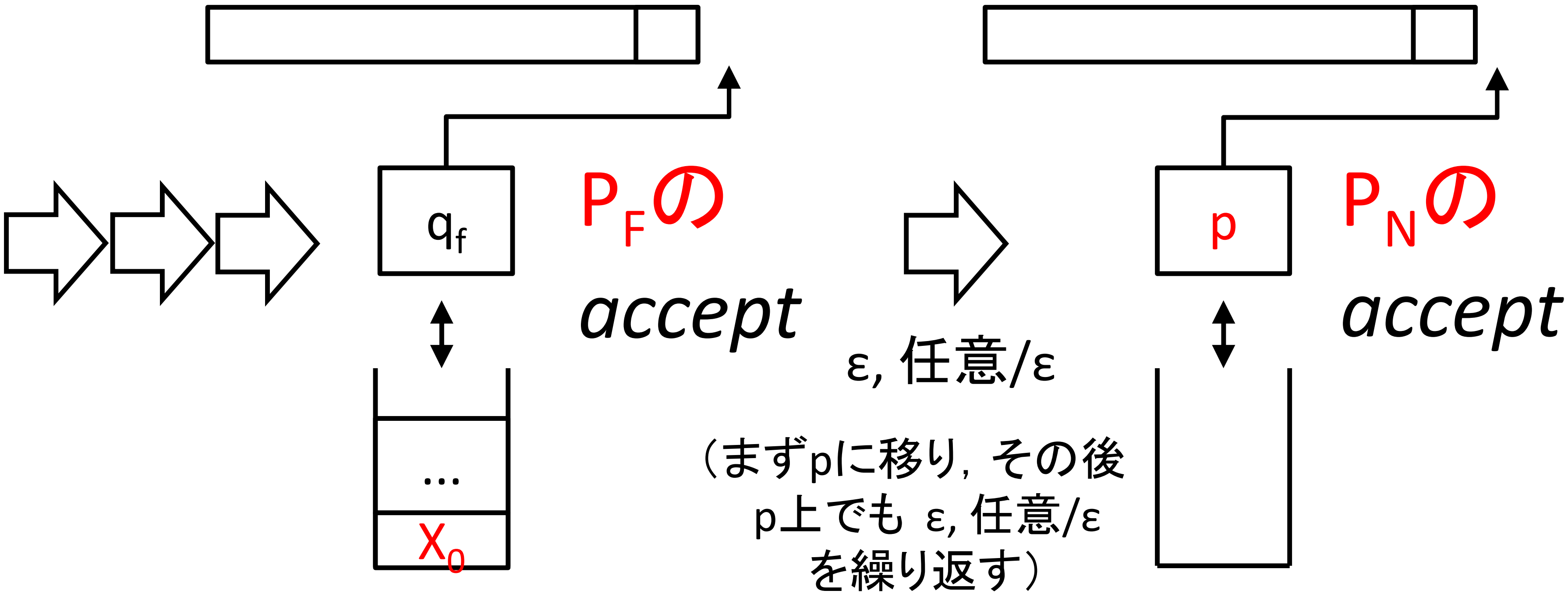
P_FからのP_Nへの変換

- 最初にスタックのZ_0の下にX_0を置いておく
  - 空スタックを検知するため
- その後はP_Fの動作を模倣
  - P_Fがスタック内のX_0にアクセスすることはない



P_FからへP_Nの変換

- P_Fの受理状態p_fに移ると，特別な状態pに遷移
  - その後スタックをポップし続け，空にして受理
- $P_N = (Q \cup \{p_0, p\}, \Sigma, \Gamma \cup \{X_0\}, \delta_N, p_0, X_0)$



P_F とP_Nの関係

- 空スタック受理PDA ならば 最終状態受理PDA
- 最終状態受理PDA ならば 空スタック受理PDA

より,

空スタック受理PDA ⇔ 最終状態受理PDA

- PDAの定義にどちらを用いても同じ
  - 使いやすい方を選べばよい

ミニレポート：10-1

- テキスト p.251　図6.2のPDA
  - PDAが入力 0110 を受理する動作を順を追って示せ
    - 各遷移ごとにPDAの図を示せ
  - PDAの図は右下図のように描け

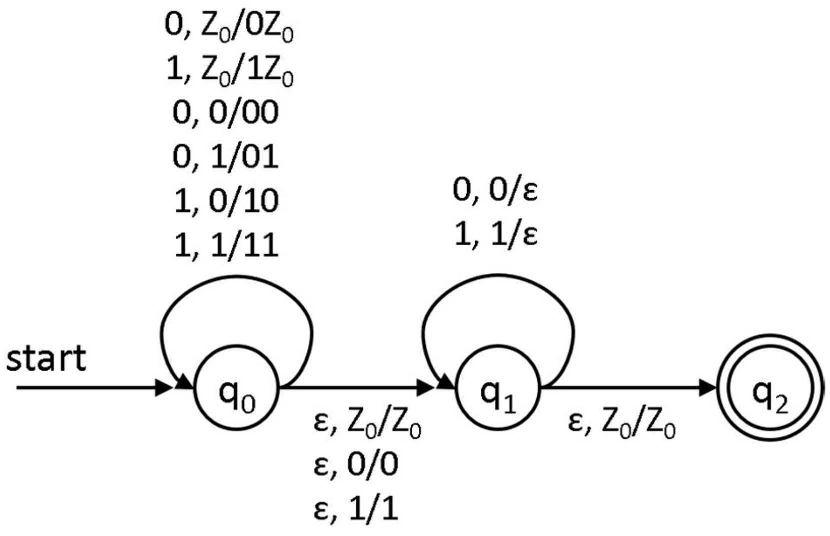
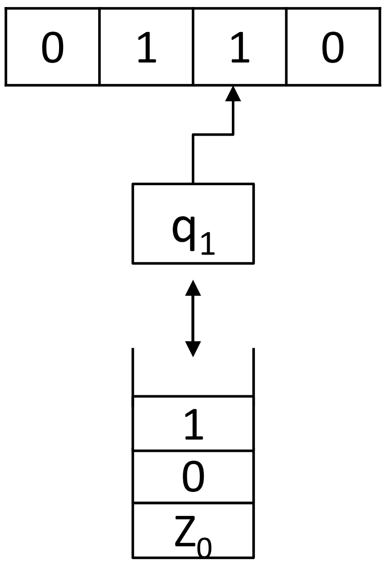


図6.2



ミニレポート

ミニレポート：10-2

- テキストp263 問6.2.1 a) (一部変更)
- 次の言語を受理するPDAを設計せよ.
  - ただし空スタック受理のPDAとせよ
- a) $\{0^n1^n \mid n \geqq 1\}$