

プッシュダウンオートマトン

離散数学・オートマトン

2024 年後期

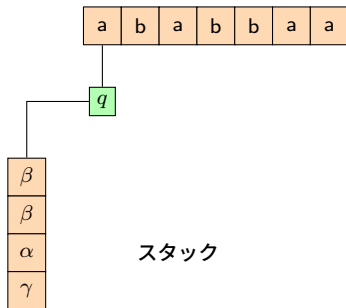
佐賀大学工学部 只木進一

- ① プッシュダウンオートマトン: Pushdown automata
- ② 決定性プッシュダウンオートマトン: Deterministic PDA
- ③ PDA と受理言語: PDA and Their Accepted Languages
- ④ 非決定性プッシュダウンオートマトン: Nondeterministic PDA

プッシュダウンオートマトン: Pushdown automata

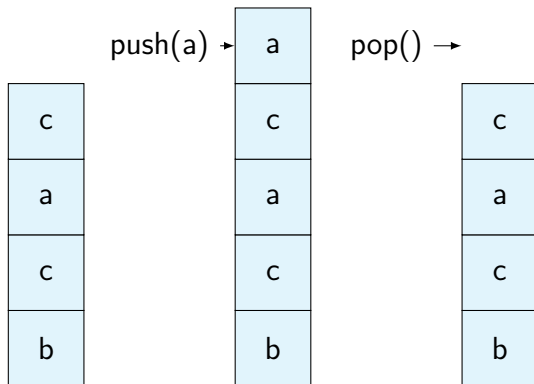
- テープとともに、スタックの文字を読む
- 状態遷移するとともに、スタックへ文字列を書き込む
- スタックという特殊な無限に大きなメモリを持つ機械

テープ



スタック: stacks

- リストのような 1 次元のデータ列
- 先頭に書く (push) ことと、先頭から読む (pop) ことだけが許される
 - 先頭以外のデータは触れない
 - FILO (First-In Last-Out)
 - pop: 先頭を取り出して読む、つまり、先頭の要素はスタックから無くなることに注意
- Python での実装例: 次シート
 - deque を利用
 - append(): 最後に要素を追加
 - pop(): 最後の要素を取り出し、削除



Stack クラス定義

```

1  from collections import deque
2  from typing import TypeVar, Generic
3  T = TypeVar('T')
4  class Stack(Generic[T]):
5      """
6      スタックのクラス
7      """
8      def __init__(self): #コンストラクタ
9          self.elements: deque[T] = deque()
10     def is_empty(self) -> bool: #要素が無いとき True
11         return len(self.elements) == 0
12     def push(self, e: T) -> None: #要素を追加
13         self.elements.append(e)
14     def pop(self) -> T: #末尾の要素を取り出す。要素は削除される
15         return self.elements.pop()
16     def peek(self) -> T: #末尾の要素を調べる
17         return self.elements[-1]
18     def size(self) -> int: #要素数を返す
19         return len(self.elements)
20     def __str__(self) -> str: #文字列化
21         s = '['
22         for x in self.elements:
23             s += str(x) + ','
24         s = s.removesuffix(',')
25         s += ']'
26         return s

```

Stack クラス利用例

```
1 myStack = Stack[str]()
2 myStack.push('a')
3 myStack.push('b')
4 myStack.push('b')
5 print(myStack)
6
7 myStack.pop()
8 myStack.pop()
9 print(myStack)
```

<https://github.com/discrete-math-saga/PDA>

決定性プッシュダウンオートマトン

Deterministic Pushdown Automata

$$M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle \quad (2.1)$$

- Q : 内部状態の集合
- Σ : テープのアルファベット
- Γ : スタックのアルファベット
- $\delta : Q \times \Sigma \times \Gamma \rightarrow Q \times \Gamma^*$: 遷移関数
 - 注意: スタックから1文字読み、文字列を書き込む
 - 文字列は、右側からスタックに書き込む
- $q_0 \in Q$: 初期状態
- $Z_0 \in \Gamma$: スタックの底の記号
- $F \subseteq Q$: 終状態の集合

例 2.1:

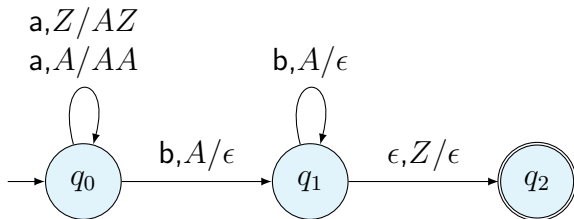
$$Q = \{q_0, q_1, q_2\},$$

$$F = \{q_2\},$$

$$\Sigma = \{a, b\},$$

$$\Gamma = \{A, Z\}.$$

$$\begin{aligned} \delta(q_0, a, Z) &= (q_0, AZ), & \delta(q_0, a, A) &= (q_0, A), & \delta(q_0, b, A) &= (q_0, \epsilon), \\ \delta(q_1, b, A) &= (q_1, \epsilon), & \delta(q_1, \epsilon, Z) &= (q_2, \epsilon). \end{aligned}$$

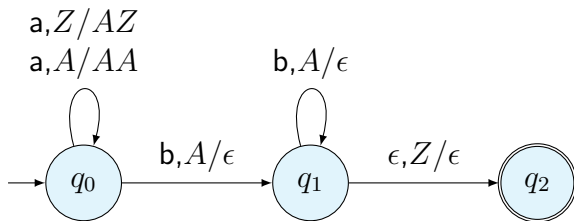


注意: $\delta(q_1, \epsilon, Z)$ は決定的動作であることに注意。スタック文字が Z のときのみ。

動作: (Q, Σ^*, Γ^*) の変化

$(q_0, aaabbb, Z) \vdash (q_0, aabbb, AZ)$ a を読んでいる間は q_0 に留まる
 $\vdash (q_0, abbb, AAZ)$
 $\vdash (q_0, bbb, AAAZ)$ b を読むと q_1 へ遷移
 $\vdash (q_1, bb, AAZ)$
 $\vdash (q_1, b, AZ)$
 $\vdash (q_1, \epsilon, Z)$ 空スタックで q_2 へ遷移
 $\vdash (q_2, \epsilon, \epsilon)$

aabb に対する動作



動作失敗: a と b の数が異なる

$$\begin{aligned}(q_0, aaabb, X) &\vdash (q_0, aabb, AZ) \\ &\vdash (q_0, abb, AAZ) \\ &\vdash (q_0, bb, AAAZ) \\ &\vdash (q_1, b, AAZ) \\ &\vdash (q_1, \epsilon, AZ)\end{aligned}$$

例 2.2:

$$Q = \{q_0, q_1, q_2, q_3\},$$

$$\Sigma = \{a, b, c, d\},$$

$$F = \{q_3\},$$

$$\Gamma = \{A, B, Z\}.$$

$$\delta(q_0, a, Z) = (q_0, AZ),$$

$$\delta(q_0, a, A) = (q_0, AA),$$

$$\delta(q_0, a, B) = (q_0, AB),$$

$$\delta(q_0, c, A) = (q_1, \epsilon),$$

$$\delta(q_0, d, A) = (q_2, \epsilon),$$

$$\delta(q_1, c, A) = (q_1, \epsilon),$$

$$\delta(q_2, d, A) = (q_2, \epsilon),$$

$$\delta(q_1, \epsilon, Z) = (q_3, \epsilon),$$

$$\delta(q_0, b, Z) = (q_0, BZ),$$

$$\delta(q_0, b, A) = (q_0, BA),$$

$$\delta(q_0, b, B) = (q_0, BB),$$

$$\delta(q_0, d, B) = (q_1, \epsilon),$$

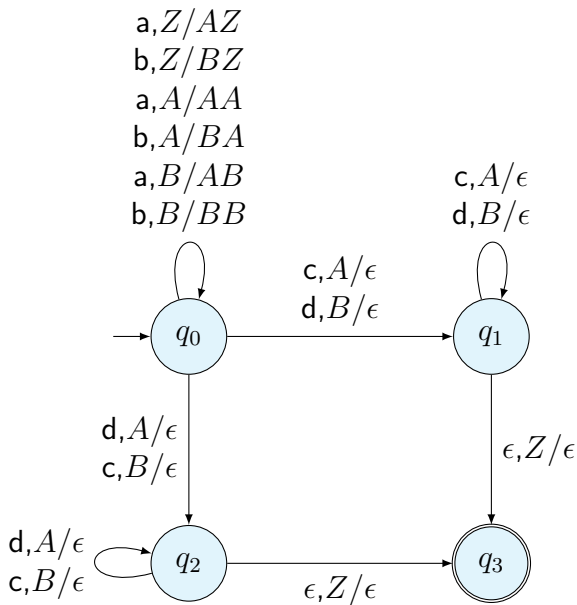
$$\delta(q_0, c, B) = (q_2, \epsilon),$$

$$\delta(q_1, d, B) = (q_1, \epsilon),$$

$$\delta(q_2, c, B) = (q_2, \epsilon),$$

$$\delta(q_2, \epsilon, Z) = (q_3, \epsilon)$$

注意: $\delta(q_1, \epsilon, Z)$ は決定的動作であることに注意。スタック文字が Z のときのみ。



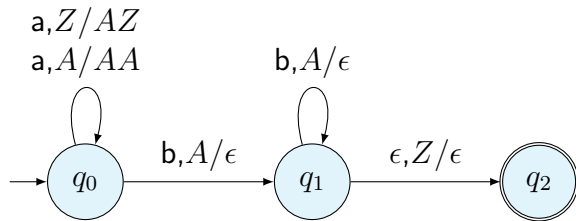
動作例:abaaccdc

$$\begin{aligned}(q_0, abaaddcd, Z) &\vdash (q_0, baaddcd, AZ) \\ &\vdash (q_0, aaddcd, BAZ) \\ &\vdash (q_0, addcd, ABAZ) \\ &\vdash (q_0, ddcd, AABAZ) \\ &\vdash (q_2, dcd, ABAZ) \\ &\vdash (q_2, cd, BAZ) \\ &\vdash (q_2, d, AZ) \\ &\vdash (q_2, \epsilon, Z) \\ &\vdash (q_3, \epsilon, \epsilon)\end{aligned}$$

動作例: ababdcdc

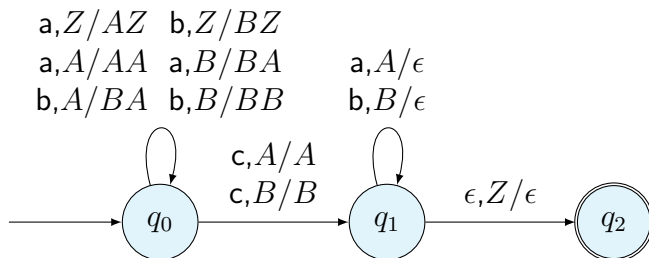
$$\begin{aligned}(q_0, ababdcdc, Z) &\vdash (q_0, babdcdc, AZ) \\ &\vdash (q_0, abdcdc, BAZ) \\ &\vdash (q_0, bdcdc, AB AZ) \\ &\vdash (q_0, dc dc, BAB AZ) \\ &\vdash (q_1, dc, AB AZ) \\ &\vdash (q_1, dc, BAZ) \\ &\vdash (q_1, c, AZ) \\ &\vdash (q_1, \epsilon, Z) \\ &\vdash (q_3, \epsilon, \epsilon)\end{aligned}$$

受理言語: Accepted Languages



- 入力とスタックが空になった時に、終状態に居るか？
- 例 2.1 では、 $\{a^i b^i \mid i \in N\}$ を受理
 - a の数をスタック文字 A で記録
 - テープ上の b とスタック上の A を照合
 - FA では受理できない言語: 任意の数の a の「数」を記録
- 例 2.2 の受理言語は？
- 次の例 3.1 では、 $\{wcw^R \mid w \in (a+b)^*\}$ を受理

例 3.1:



$$Q = \{q_0, q_1, q_2\},$$

$$\Sigma = \{a, b, c\},$$

$$F = \{q_2\},$$

$$\Gamma = \{A, B, Z\}.$$

$$\delta(q_0, a, Z) = (q_0, AZ), \quad \delta(q_0, a, A) = (q_0, AA), \quad \delta(q_0, a, B) = (q_0, AB),$$

$$\delta(q_0, b, Z) = (q_0, BZ), \quad \delta(q_0, b, A) = (q_0, BA), \quad \delta(q_0, b, B) = (q_0, BB),$$

$$\delta(q_0, c, A) = (q_1, A), \quad \delta(q_0, c, B) = (q_1, B),$$

$$\delta(q_1, a, A) = (q_1, \epsilon), \quad \delta(q_1, b, B) = (q_1, \epsilon), \quad \delta(q_1, \epsilon, Z) = (q_2, \epsilon).$$

動作例

$$\begin{aligned}
 (q_0, \text{abaacaaba}, Z) &\vdash (q_0, \text{baacaaba}, AZ) \\
 &\vdash (q_0, \text{aacaaba}, BAZ) \\
 &\vdash (q_0, \text{acaaba}, AB AZ) \\
 &\vdash (q_0, \text{caaba}, AAB AZ) \\
 &\vdash (q_1, \text{aaba}, AAB AZ) \\
 &\vdash (q_1, \text{aba}, AAB AZ) \\
 &\vdash (q_1, \text{ba}, BAZ) \\
 &\vdash (q_1, \text{a}, AZ) \\
 &\vdash (q_1, \epsilon, Z) \\
 &\vdash (q_2, \epsilon, \epsilon)
 \end{aligned}$$

動作失敗例

$$\begin{aligned}(q_0, \text{abaaaaba}, Z) &\vdash (q_0, \text{baaaaba}, AZ) \\ &\vdash (q_0, \text{aaaaba}, BAZ) \\ &\vdash (q_0, \text{aaaba}, AB AZ) \\ &\vdash (q_0, \text{aaba}, AA B AZ) \\ &\vdash (q_0, \text{aba}, AAA B AZ) \\ &\vdash (q_0, \text{ba}, AAAA B AZ) \\ &\vdash (q_0, \text{a}, BAAAA B AZ) \\ &\vdash (q_0, \epsilon, ABAAAA B AZ)\end{aligned}$$

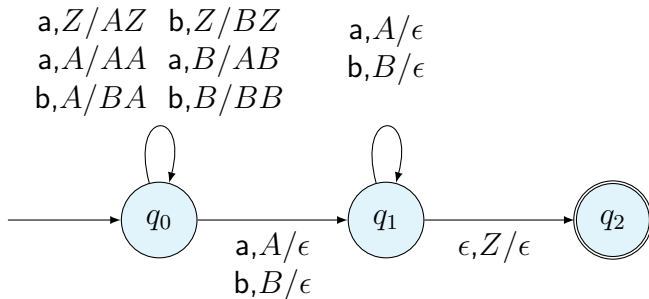
非決定性プッシュダウンオートマトン

Non-deterministic PDA

$$M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle \quad (4.1)$$

- Q : 内部状態の集合
- Σ : テープのアルファベット
- Γ : スタックのアルファベット
- $\delta : Q \times \Sigma \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$: 遷移関数
- $q_0 \in Q$: 初期状態
- $Z_0 \in \Gamma$: スタックの底の記号
- $F \subseteq Q$: 終状態の集合

例 4.1:



$$Q = \{q_0, q_1, q_2\},$$
$$\Sigma = \{a, b, c\},$$

$$F = \{q_2\},$$
$$\Gamma = \{A, B, Z\}.$$

$$\delta(q_0, a, Z) = \{(q_0, AZ)\}$$

$$\delta(q_0, a, B) = \{(q_0, AB)\}$$

$$\delta(q_0, b, A) = \{(q_0, BA)\}$$

$$\delta(q_1, a, A) = \{(q_1, \epsilon)\}$$

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

$$\delta(q_0, a, A) = \{(q_0, AA), (q_1, \epsilon)\}$$

$$\delta(q_0, b, Z) = \{(q_0, BZ)\}$$

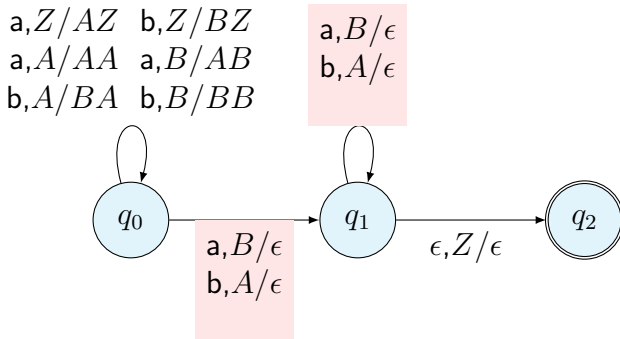
$$\delta(q_0, b, B) = \{(q_0, BB), (q_1, \epsilon)\}$$

$$\delta(q_1, b, B) = \{(q_1, \epsilon)\}$$

動作 (受理した例)

$$\begin{aligned}(q_0, abaaaaba, Z) &\vdash (q_0, baaaaba, AZ) \\ &\vdash (q_0, aaaaba, BAZ) \\ &\vdash (q_0, aaaba, ABABZ) \\ &\vdash (q_0, aaba, AABABZ) \\ &\vdash (q_1, aba, AABABZ) \\ &\vdash (q_1, ba, BAZ) \\ &\vdash (q_1, a, AZ) \\ &\vdash (q_1, \epsilon, Z) \\ &\vdash (q_1, \epsilon, \epsilon)\end{aligned}$$

例 4.2:



$$Q = \{q_0, q_1\},$$

$$\Sigma = \{a, b, c\},$$

$$F = \{q_1\},$$

$$\Gamma = \{A, B, Z\}.$$

$$\begin{aligned}\delta(q_0, a, Z) &= \{(q_0, AZ)\} \\ \delta(q_0, a, B) &= \{(q_0, AB), (q_1, \epsilon)\} \\ \delta(q_0, b, A) &= \{(q_0, BA), (q_1, \epsilon)\} \\ \delta(q_1, a, B) &= \{(q_1, \epsilon)\} \\ \delta(q_1, \epsilon, Z) &= \{(q_2, \epsilon)\}\end{aligned}$$

$$\begin{aligned}\delta(q_0, a, A) &= \{(q_0, AA)\} \\ \delta(q_0, b, Z) &= \{(q_0, BZ)\} \\ \delta(q_0, b, B) &= \{(q_0, BB)\} \\ \delta(q_1, b, A) &= \{(q_1, \epsilon)\}\end{aligned}$$

動作 (受理した例)

$$\begin{aligned}(q_0, \text{abaabbab}, Z) &\vdash (q_0, \text{baabbab}, AZ) \\ &\vdash (q_0, \text{aabbab}, BAZ) \\ &\vdash (q_0, \text{abbab}, AB AZ) \\ &\vdash (q_0, \text{bbab}, AAB AZ) \\ &\vdash (q_1, \text{bab}, AB AZ) \\ &\vdash (q_1, \text{ab}, BAZ) \\ &\vdash (q_1, \text{b}, AZ) \\ &\vdash (q_1, \epsilon, Z) \\ &\vdash (q_1, \epsilon, \epsilon)\end{aligned}$$

- 最初の例では、 $\{ww^R | w \in (a + b)^*\}$ を受理。折り返しの文字 c は不要
- 二番目の例では、 w^R において a と b を入れ替えた文字列を受理

PDAの受理言語: Accepted Languages of PDA

- PDAの受理言語は、正規表現では表せないもの
 - 前半と後半の文字数が同じ、前後を反転などは正規表現では表せない
 - $L_{01} = \{0^n 1^n \mid n \in N\}$ は正規表現で表せない: 反復補題
- スタックを使うことで、前半の文字列を覚えることができる
 - 長さに制限なし
- 再帰関数の実装にはスタックが必要

回文の集合は正規言語ではない:

A set of palindromes is not a regular language

- 0 と 1 からなる回文全体 L が正規言語であると仮定
- 反復補題に現れる n に対して、 $w = xyz = 0^n 1^n 1^n 0^n$ を考える
- $|xy| \leq n$ より、 $y = 0^k (1 \leq k \leq n)$

$$xy^0z = xz = 0^{n-k} 1^n 1^n 0^n \notin L$$

- L は正規言語ではない

回文 (palindrome) 受理する Python コード: 再帰

```
1 def palindrome(inputStr:str) -> bool:
2     if len(inputStr) <= 1:
3         return True
4     if inputStr[0] != inputStr[-1]:
5         return False
6     return palindrome(inputStr[1:-1])
```


回文 (palindrome) 受理する Python コード: 非再帰

```
1  def palindrome2(inputStr:str) -> bool:
2      myStack:Stack[str] = Stack()
3      n = len(inputStr)
4      m = n//2
5      for i in range(m): #前半をスタックに積む
6          myStack.push(inputStr[i])
7      if n % 2 !=0: #入力の長さが奇数の場合
8          m += 1
9      for j in range(m,n): #後半と照合
10         c = myStack.pop()
11         if c != inputStr[j]:
12             return False
13     return True
```