

# 2008年度 情報数理学

# 履修にあたって

2008年度  
大学院奇数セメスター(前期)開講

教室: K336→大学院棟D416(次回から)

時限: 火曜日3時限(12:50-14:20)

担当

草苺良至

# 講義予定

○計算機のいろいろな理論モデル → 言語理論

○計算の限界  
○問題の難しさ → 計算量理論

○現実問題と計算 → アルゴリズム論

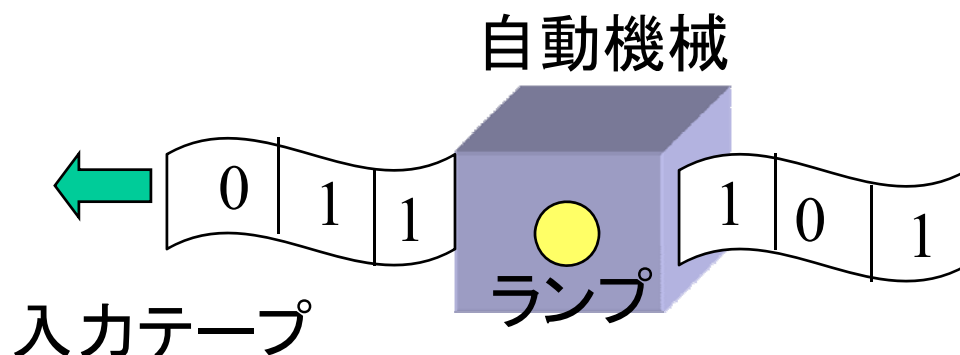
## 参考書

- M. Sipser著、  
「計算理論の基礎」、  
共立出版、1997、ISBN:4-320-02948-8
- 岩間一雄、  
「オートマトン・言語と計算理論」  
コロナ社、2003、ISBN:4-339-01821-X
- 岩間一雄、  
「アルゴリズム理論入門」  
昭晃堂、2001、ISBN:4-7856-3125-2
- ホップクロフト、ウルマン、  
「オートマトン・言語理論・計算論 I,II」  
サイエンス社、1984、ISBN:4-7819-0374-6,4-7819-0432-7
- M.R. Garey and D.S.Johnson,  
"Computers And Intractability:A guide to the Theory of NP-Completeness,"  
Freeman,1979,ISBN:0-7167-1045-5
- V.V.ヴィジラーニ著、浅野 孝夫訳、  
「近似アルゴリズム」、  
シュプリンガー・フェアラーク東京、2002、  
ISBN:4-431-70991-6

# 1. オートマトンと正規表現

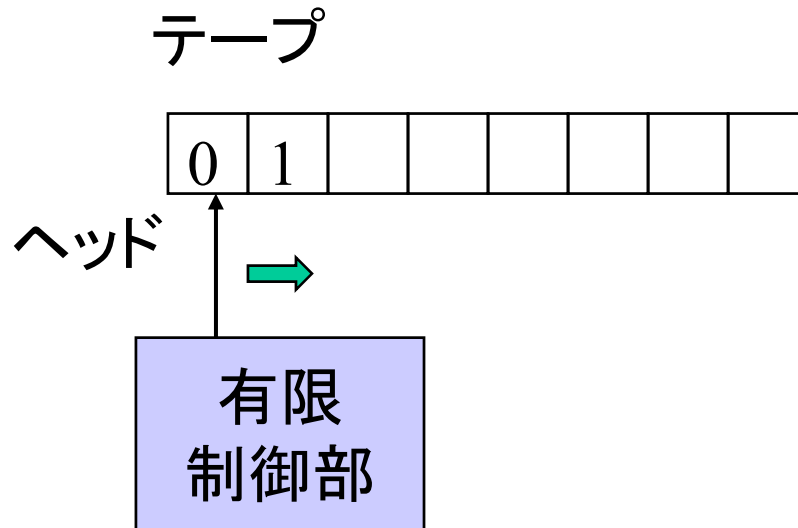
## 1-1. 有限オートマトン

メモリがほとんどなく、  
「はい」と「いいえ」しか答えられない計算機を考える。



入力テープを”一度だけ“走査したあと、  
「はい」ならランプ点灯  
「いいえ」ならランプ消灯。  
このような自動機械を(有限)オートマトンという。

# 有限オートマトンの概略



オートマトンを定める要素

入力テープ

テープに書ける文字

有限制御部

内部状態

初期状態

状態変化

受理かどうかの判断

# 有限オートマトンの数学的定義

定義 : (有限オートマトン)

有限オートマトンは、 $M = (Q, \Sigma, \delta, q_0, F)$  の5項組で与えられる。  
ここで、

1.  $Q$  は有限集合で、状態を表す。
2.  $\Sigma$  は有限集合で、入力記号の集合を表す。
3.  $\delta$  は  $Q \times \Sigma$  から  $Q$  への写像 ( $\delta: Q \times \Sigma \rightarrow Q$ ) で、状態遷移を表す。 $\delta$  を状態遷移関数という。
4.  $q_0 \in Q$  は、初期状態を表す。
5.  $F \subseteq Q$  は受理状態の集合を表す。

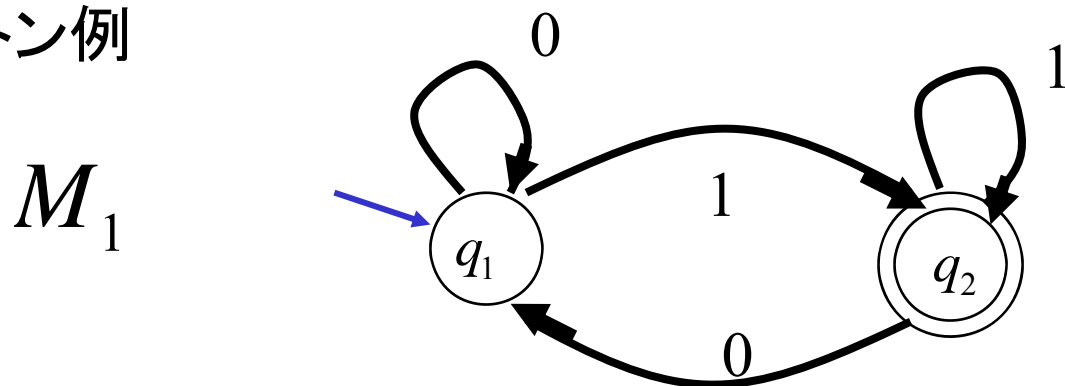
とする。



# 有限オートマトンの図式表現(状態遷移図)

有限オートマトンは、状態遷移図で表現できる。

オートマトン例



このオートマトンの形式的定義(数学的定義)は、

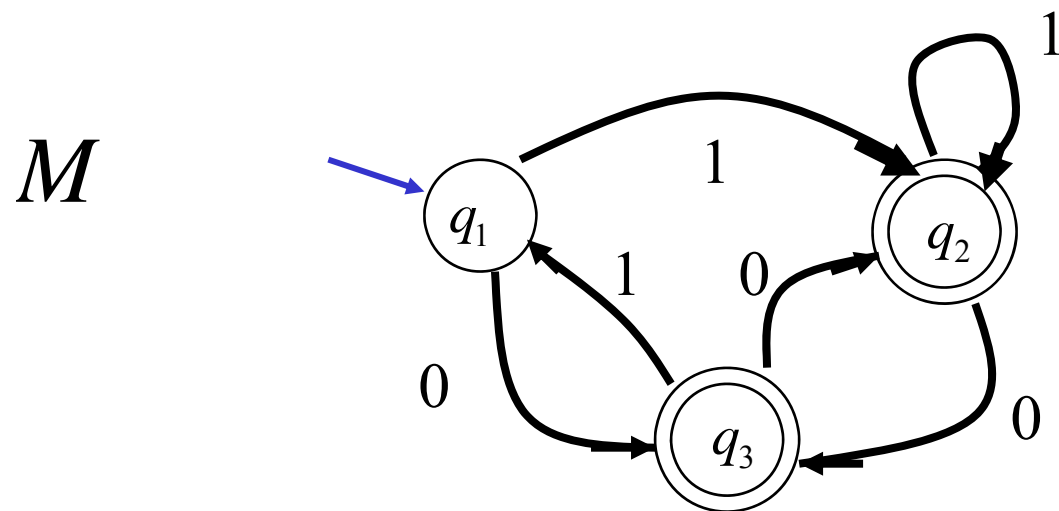
$$M_1 = (\{q_1, q_2\}, \{0, 1\}, \delta, q_1, \{q_2\})$$

であり、 $\delta$  は次の状態遷移表により定義される。

$\delta$	0	1
$q_1$	$q_1$	$q_2$
$q_2$	$q_1$	$q_2$

# 練習

次のオートマトンの数学的表現を与えよ。



## 1-2. 言語

ここで、計算機で扱える対象について再考する。

計算機が扱える対象は、 $\{0,1\}$ で表された**数**と考えがちである。  
しかし、 $\{0,1\}$ の並びを一種の**言語**とみなすこともできる。

以下では、言語の数学的定義を与える。

定義：（言語）

任意の有限集合を**アルファベット**という。

アルファベットの要素を**文字**という。

アルファベットの任意の列を**文字列**という。

文字列の集合を、（アルファベット上の）**言語**という。

# 言語の例1

アルファベット例:

$$\Sigma_1 = \{a,b,c,d,\dots,z,(\text{スペース}),(\text{ピリオド})\}$$

$\Sigma_1$  上の文字列例:

a   aa   ab   book

$\Sigma_1$  上の言語例:

$$L_1 = \{w \mid w \text{ は } a \text{ で始まる文字列}\}$$

$$= \{a, aa, ab, ac, ad, \dots, a \text{ }, a., aaa, \dots\}$$

$$L_2 = \{\text{this, that, is, a, pen, this is a pen., that is a pen.}\}$$

$$L_3 = \{\text{全ての英単語}\}$$

$$L_4 = \{(\Sigma_1 \text{ 以外の記号を無視した) 全ての英文}\}$$

## 言語の例2

アルファベット例:

$$\Sigma_2 = \{0, 1\}$$

$\Sigma_2$  上の文字列例:

0    00    001    100010001111110111

$\Sigma_2$  上の言語例:

$$\begin{aligned} L_3 &= \{w \mid w \text{は} 1 \text{で終わる文字列}\} \\ &= \{1, 01, 11, 001, 011, 101, 111, \dots\} \end{aligned}$$

$$\begin{aligned} L_4 &= \{w \mid w \text{は} 1 \text{が奇数個である文字列}\} \\ &= \{1, 01, 10, 001, 010, 100, 111, 0000001000101, \dots\} \end{aligned}$$

# 言語に関する諸概念1

ここでは、**文字列**に関する諸概念の定義を与える。

定義：（文字列関連）

**文字列の長さ**：

文字列 $w$ に含まれる文字数を、文字列 $w$ の**長さ**といい、 $|w|$  という記号で表す。

**空列**：

長さが0の文字列を**空列**といい、記号 $\varepsilon$ で表す。

**連結**：

文字列 $x$ の後ろに文字列 $y$ を繋げてえられる文字列を $x$ と $y$ の**連結**といい次のような記号で表す。

$$xy \quad x \circ y \quad x^k = \underbrace{xx \cdots x}_k$$

## 例

$\Sigma_2 = \{0,1\}$  上の文字列を考える。

$w = 01, x = 011, y = 01011$  とする。

このとき、次式が成り立つ。

$$|w| = 2, |x| = 3, |y| = 5$$

$$w^0 = x^0 = y^0 = \varepsilon$$

$$|\varepsilon| = 0$$

$$y = wx \quad y \neq xw$$

$$w^2 = 0101, w^3 = 010101$$

文字列の連結演算は、  
交換不可

## 言語に関する諸概念2

ここでは、**言語**に関する諸概念の定義を与える。

**定義** : (言語関連)

$A$  と  $B$  を言語とする。

**言語の和集合(和集合演算)** :

$$A \cup B = \{x \mid x \in A \text{ または } x \in B\}$$

**言語の連結(連結演算)** :

$$A \circ B = AB = \{xy \mid x \in A \text{ かつ } y \in B\}$$

**言語の閉包(スター演算)** :

$$A^* = \{x_1 x_2 \cdots x_k \mid k \geq 0 \text{ かつ } x_1, x_2, \dots, x_k \in A\}$$



## 例

$\Sigma_2 = \{0,1\}$  上の言語を考える。

$L_1 = \{10,1\}$ ,  $L_2 = \{011,11\}$  とする。

このとき、次式が成り立つ。

$$L_1 \cup L_2 = \{10,1,011,11\}$$

$$L_1 \circ L_2 = \{10011,1011,111\}$$

$$L_1^0 = \{\varepsilon\}, \quad L_1^1 = L_1 = \{10,1\},$$

$$L_1^2 = L_1 L_1 = \{1010,101,110,11\}$$

$$L_1^* = \{\varepsilon, 10, 1, 1010, 101, 110, 11, 101010, \dots\}$$

# 要素の無い言語と空列だけの言語

要素の無い言語と空列だけの言語は異なる。

$$L_1 = \{\} = \phi, L_2 = \{\varepsilon\} \quad \text{とする。}$$

このとき、

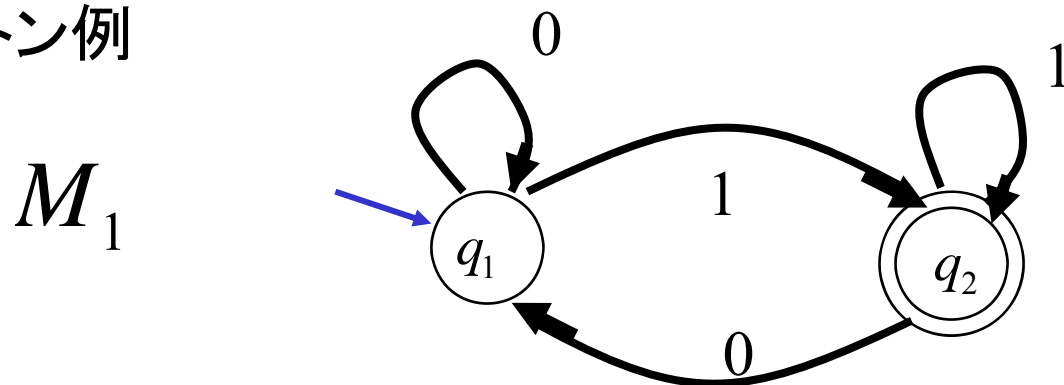
$$L_1 \neq L_2$$

である。

# オートマトンと言語

オートマトンによって受理される入力の集合は、  
入力記号  $\Sigma$  上の言語になっている。

オートマトン例



このオートマトン  $M_1$  で受理される言語を  
 $L(M_1)$  と書く。

例えば、

$L(M_1) = \{w \mid w \text{ は } 1 \text{ で終わる文字列}\}$   
である。

## 練習

次の言語を受理するオートマトン  $M$  を作成せよ。

$$L(M) = \{w \mid w \text{ は } 0 \text{ で終わる文字列}\}$$

オートマトンは、状態遷移図および、形式的定義の両方で示す事。

## 1-3. 非決定性(有限)オートマトン

オートマトンでは、入力記号にしたがって、状態遷移は一意に定められていた。

この制限を緩和した計算機モデルが考えられる。



**非決定性オートマトン**とは、同じ入力に対して複数の遷移をゆるす”オートマトン”である。

これに対して、同じ入力に対して、一つの遷移しかおこなえない”オートマトン”を**決定性オートマトン**という。

# オートマトンの略記

決定性オートマトンは、英語では、  
Deterministic Finite Automaton  
であり、

**DFA**

と略記される。

非決定性オートマトンは、英語では、  
Non-determinisc Finite Automaton  
であり、

**NFA**

と略記される。

# NFAの形式的定義

定義：（非決定性オートマトン）

非決定性有限オートマトンは、 $N = (Q, \Sigma, \delta', q_0, F)$  の5項組で与えられる。ここで、

1.  $Q$  は有限集合で、状態を表す。
2.  $\Sigma$  は有限集合で、入力記号の集合を表す。
3.  $\delta'$  は  $Q \times \Sigma$  から  $\mathcal{P}(Q)$  への写像

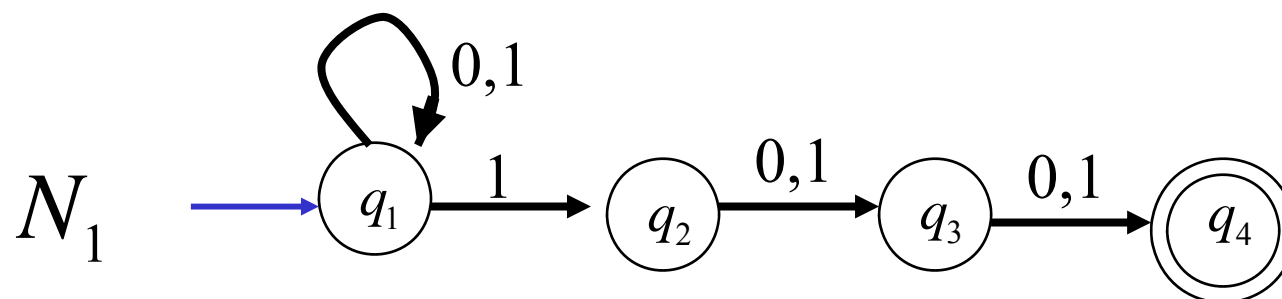
$$\delta': Q \times \Sigma \rightarrow \mathcal{P}(Q)$$

で、状態遷移を表す。 $\delta$  を状態遷移関数という。

4.  $q_0 \in Q$  は、初期状態を表す。
5.  $F \subseteq Q$  は受理状態の集合を表す。

とする。

# NFAの状態遷移図



このオートマトンの形式的定義(数学的定義)は、

$$N_1 = (\{q_1, q_2, q_3, q_4\}, \{0, 1\}, \delta, q_1, \{q_4\})$$

であり、 $\delta$  は次の状態遷移表により定義される。

$\delta$	0	1
$q_1$	$\{q_1\}$	$\{q_1, q_2\}$
$q_2$	$\{q_3\}$	$\{q_3\}$
$q_3$	$\{q_4\}$	$\{q_4\}$
$q_4$	$\phi$	$\phi$



このオートマトン  $N_1$  で受理される言語  $L(N_1)$  は、

$$L(N_1) = \left\{ w \mid \begin{array}{l} w \text{ は最後から 3 文字目が,} \\ 1 \text{ である文字列} \end{array} \right\}$$

である。

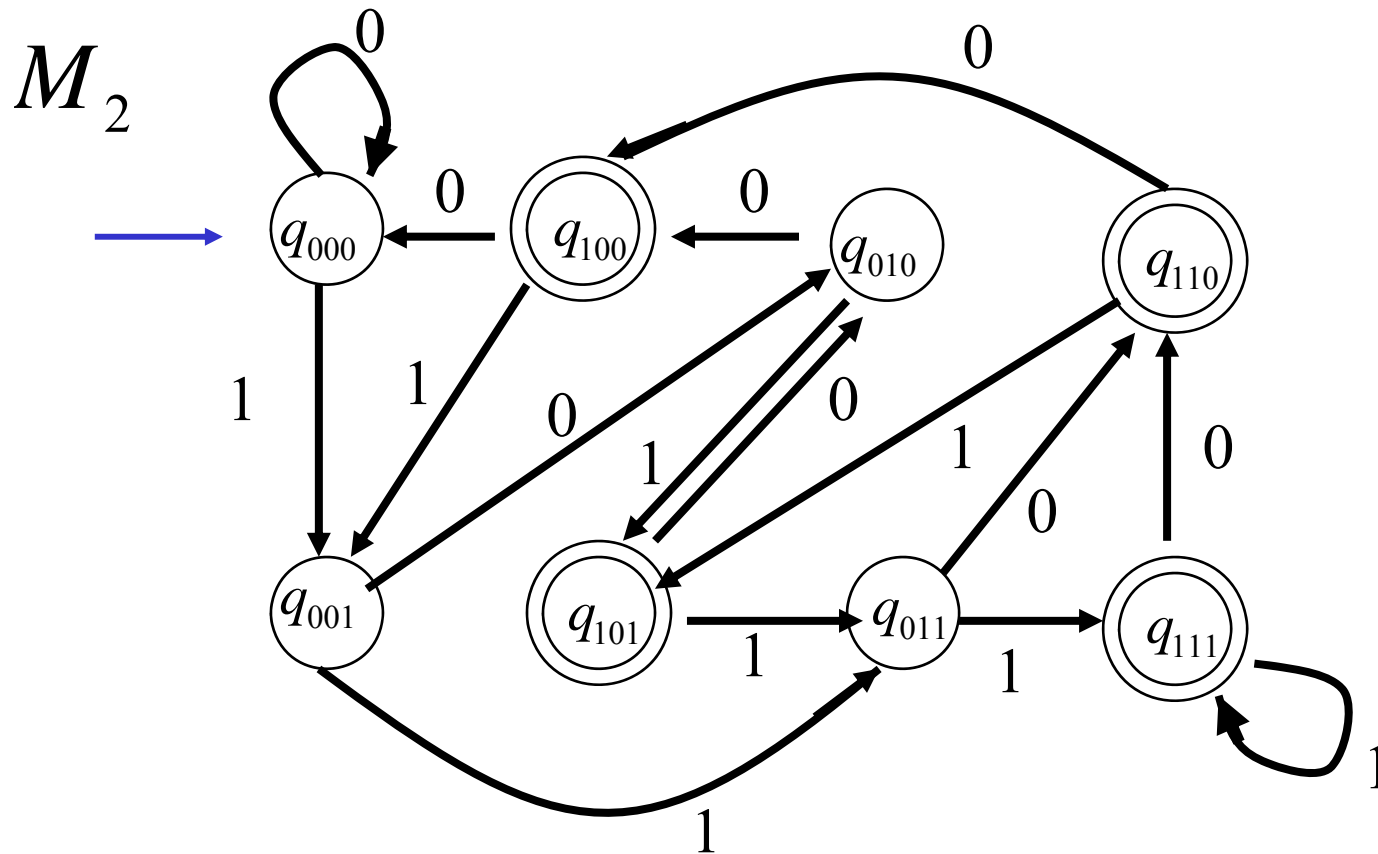
**性質** : (決定性オートマトンと非決定性オートマトン)

実は、非決定性オートマトンが受理する言語と同じ言語を受理する決定性オートマトンが常に存在する。

モデル自体の能力に差がない。  
あとで、証明する。

言語  $\left\{ w \mid \begin{array}{l} w \text{は最後から 3 文字目が,} \\ 1 \text{ である文字列} \end{array} \right\}$  を受理する

DFA  $M_2$  を示す。



# 練習

$\Sigma = \{0, 1\}$  上の

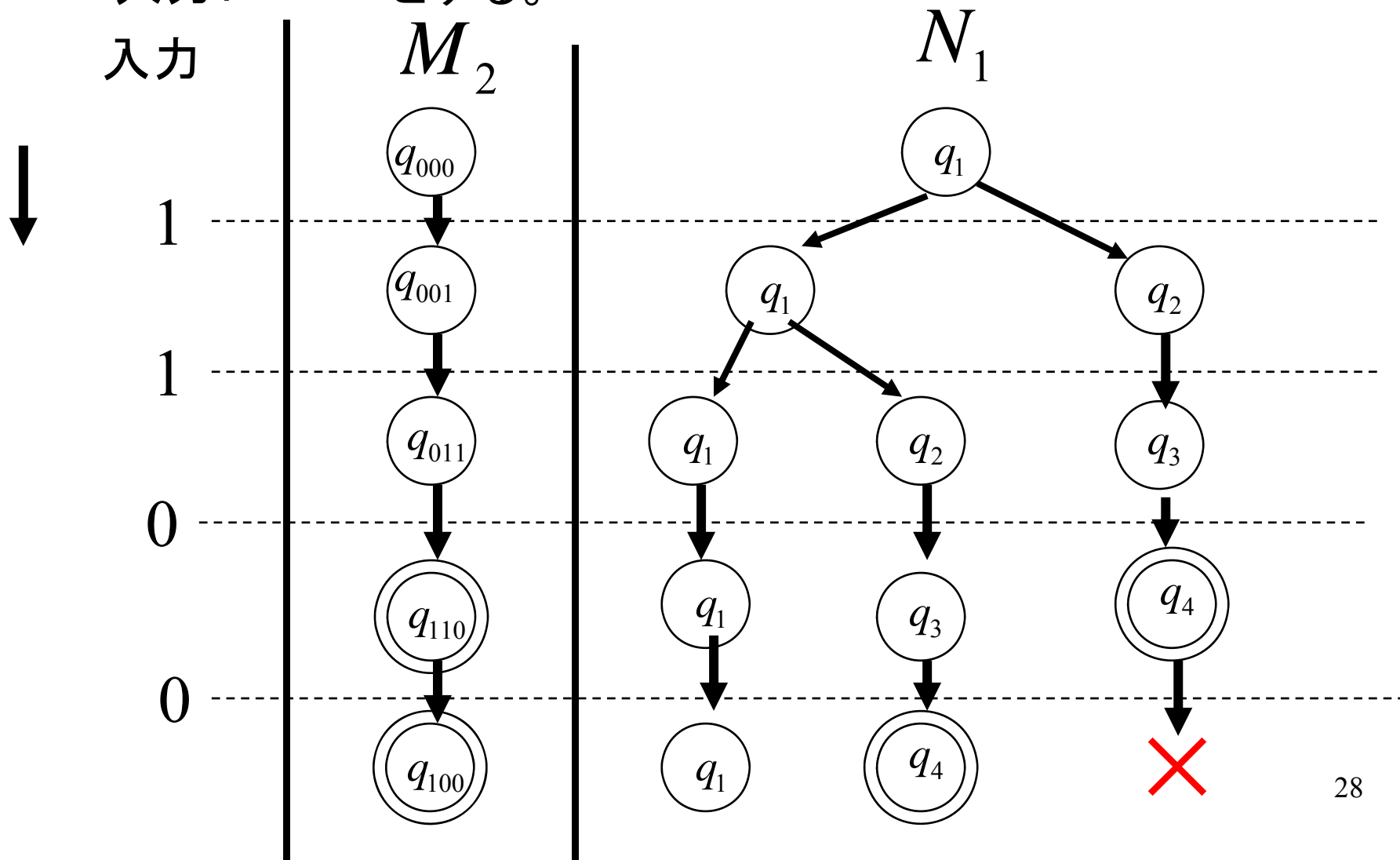
言語  $\left\{ w \left| \begin{array}{l} w \text{は最後から2文字目が,} \\ 1 \text{である文字列} \end{array} \right. \right\}$

を受理する非決定性オートマトンと決定性オートマトンを示せ。

# DFAとNFAの状態遷移

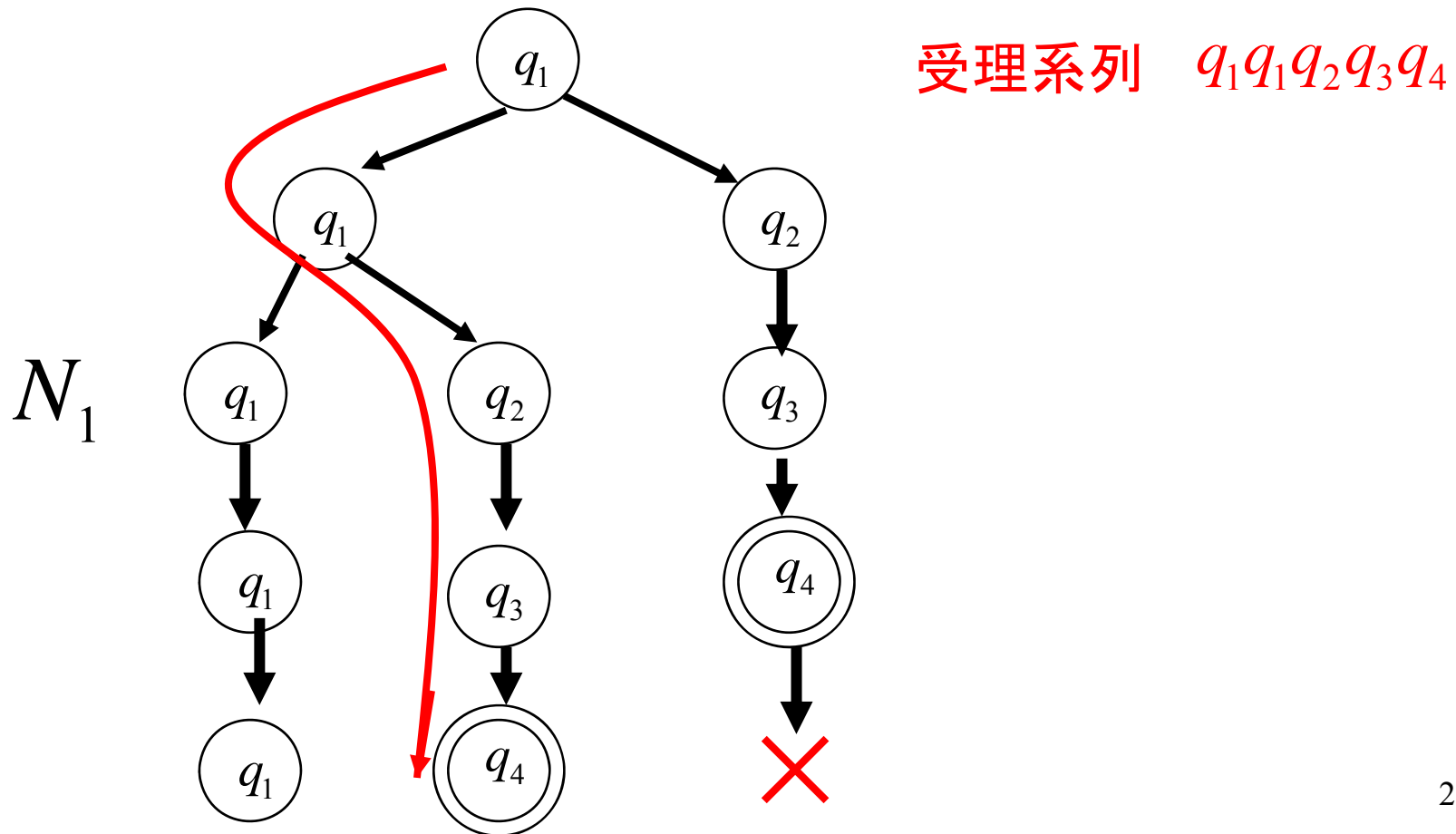
$M_2$  と  $N_1$  を例にして、DFAとNFAの状態遷移を調べる。

入力: 1100 とする。



# NFAの受理

NFAの受理とは、  
入力系列を受理する遷移の系列が**存在**する  
ことである。



# 練習

$M_2$  と  $N_1$  に対して、入力1011の状態遷移を木によって示し、  
受理か不受理かを確認せよ。

## 1-4. 正規表現(正則表現)

DFAで受理できる言語に対して、**正規表現**と呼ばれる別の表現法が知られている。

定義:(正規表現)

$\Sigma$  をアルファベットとする。

$\Sigma$  上の正規表現とは、下記の4つにより帰納的に定義される。

1.  $\phi$  で、その表す集合は、空集合である。
2.  $\varepsilon$  で、その表す集合は、 $\{\varepsilon\}$  である。
3.  $\Sigma$  の各元  $a$  に対して、 $a$  は正規表現で、その表す集合は、 $\{a\}$  である。
4.  $r$  と  $s$  がそれぞれ言語  $R$  と言語  $S$  を表す正規表現のとき、 $(r + s)$ ,  $(rs)$ ,  $(r^*)$  は正規表現で、それぞれ  $R \cup S$ ,  $RS$ ,  $R^*$  を表す。

# 正規演算の優先順位

正規表現の演算記号に優先順位をつけることによって、括弧を省略できる。

$$+ < \circ <^* < ()$$

通常は、上のように優先順位があると考えて、不必要な括弧は省略する。



## 例

アルファベット  $\Sigma = \{0,1\}$  上の正規表現を考える。

$$\varepsilon = \{\varepsilon\}, 0 = \{0\}, 1 = \{1\}, 01 = \{01\}, 10 = \{10\}$$

$$1\varepsilon = \{1\}, 0+1 = \{0,1\}, 01+10 = \{01,10\},$$

$$(1+0)(01+10) = \{101,110,001,010\}$$

$$1^* = \{\varepsilon, 1, 11, 111, 1111, 11111, \dots\}$$

$$01^* = \{0, 01, 011, 0111, 01111, 011111, \dots\}$$

$$\Sigma^* = (0+1)^*$$

$$= \{0,1\}^*$$

$$= \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, \dots\}$$

$$= \{\text{全ての 2 進数}\}$$

## 練習

アルファベットを  $\Sigma = \{a, b, c, d, \dots, z\}$  とする。

このとき、

次の正規表現で表される言語に含まれる文字列をいくつか示し、その直感的な意味を述べよ。

(1)  $m(a+e)n$

(2)  $bo^*$

(3)  $a\Sigma^*$

(4)  $\Sigma^*b\Sigma^*$

(5)  $(a + b + c)^*$

## 正規表現の応用

UNIXシェルでは、正規表現で引数を指定できる。  
ただし、UNIXの正規表現は、UNIX独特のものなので注意する。

\* : 任意の文字列を表す。 $\Sigma^*$

+ : 一文字以上の文字列。 $\Sigma^* - \{\epsilon\}$

$[c_1c_2 \cdots c_n]$  :  $c_1$  から  $c_n$  までのいずれかの1文字  
( $c_1 + c_2 + \cdots + c_n$ )

$[c_1 - c_n]$  :  $c_1$  から  $c_n$  までのいずれかの1文字  
( $c_1 + c_2 + \cdots + c_n$ )

# 例

```
~$ls *.c
average.c  hello.c      sort.c       sum.c
~$ls [ab]*
average    average.c
~$ls [h-s]*.c
hello.c    sort.c       sum.c
~$
```

\*.cは.cで終わる文字列。  
(拡張子で区別すると、特定種類のファイルだけを指定できる。)

[ab]\*はaかbで始まる文字列。  
(長いファイル名を一括して扱える。)

[h-s]\*.cはhからsのどれかの文字で始まり、.cで終わる文字列。  
(組み合わせでファイルを絞り込める。)

## 1-5. 拡張NFA

DFA、NFA共に、入力記号1文字に対して、  
1つの遷移を行っていた。

この制限を緩和した計算機モデルが考えられる。



**拡張NFA**とは、遷移のラベルとして正規表現を許す  
NFAである。

拡張NFA : Generalized Non-deterministic finite Automaton  
なのでGNFAと略する。

# GNFAの形式的定義

定義: (拡張非決定性オートマトン)

GNFAは、 $G = (Q, \Sigma, \delta, q_s, q_a)$  の5項組  
で与えられる。ここで、

1.  $Q$  は有限集合で、**状態**を表す。
2.  $\Sigma$  は有限集合で、**入力記号**の集合を表す。
3.  $\delta$  は  $(Q - \{q_a\}) \times (Q - \{q_s\})$  から  $\mathcal{R}$  への写像

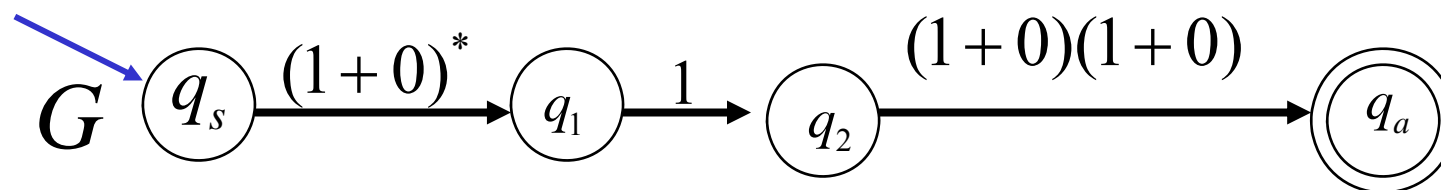
$$\delta: (Q - \{q_a\}) \times (Q - \{q_s\}) \rightarrow \mathcal{R}$$

で、**状態遷移**を表す。 $\delta$  を**状態遷移関数**という。

ただし、 $\mathcal{R}$  は  $\Sigma$  上の正規表現すべてからなる集合  
( $\Sigma$  上の正規言語)を表す。

4.  $q_s \in Q$  は、**初期状態**を表す。
5.  $q_a \in Q$  は**受理状態**を表す。  
とする。

# GNFAの状態遷移図



このオートマトンの形式的定義(数学的定義)は、

$$G = (\{q_1, q_2, q_s, q_a\}, \{0, 1\}, \delta, q_s, q_a)$$

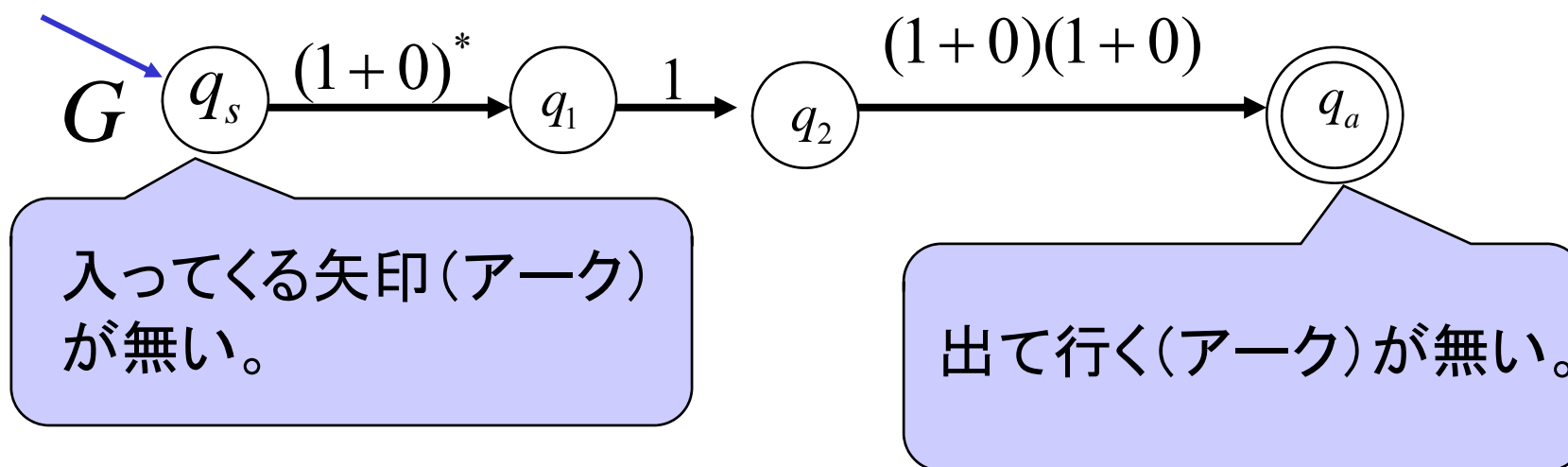
であり、 $\delta$  は次の状態遷移表により定義される。

$\delta$	$q_1$	$q_2$	$q_a$
$q_s$	$(1+0)^*$	$\phi$	$\phi$
$q_1$	$\phi$	1	$\phi$
$q_2$	$\phi$	$\phi$	$(1+0)(1+0)$

# GNFAに関する注意

初期状態  $q_s$  には、他の状態からの遷移がない。  
受理状態  $q_a$  からは、他の状態への遷移がない。

初期状態と、受理状態はそれぞれ1つずつしかない。  
特に、受理状態が1つであることに注意する。





# 練習

$\Sigma = \{0, 1\}$  上の

言語  $\left\{ w \left| \begin{array}{l} w \text{は最後から4文字目が,} \\ 0 \text{である文字列} \end{array} \right. \right\}$

を受理する4状態の拡張NFAを状態遷移図と、  
形式的定義の両方で示せ。