4. 正則言語の性質(1):
(テキスト4.1,4.2)

4.1. 言語が正則でないことの証明

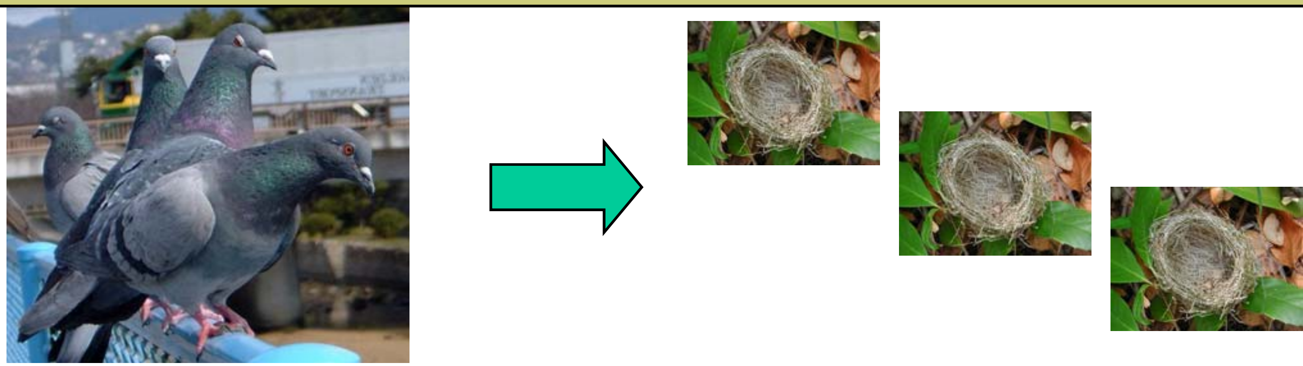
- 有限オートマトンは状態が有限個しかない。

  →「有限個の状態しかないと区別できないもの」は区別できない。

(典型的な)鳩ノ巣原理(Pigeon Hole Principle):

$n+1$羽(以上)の鳩が n 個の巣に入っている。

このとき、どこかの巣には鳩が2羽以上入っている。



4. 正則言語の性質(1): (テキスト4.1,4.2)

4.1. 言語が正則でないことの証明

例: 言語 $L=\{0^n1^n \mid n \geqq 1\}$

- n はどんなに大きくてもよい
- DFA A が m 状態なら、$n>m$ のときに 0^n1^n に関して A のふるまいは…?

例: 言語 $L=\{0^n1^n \mid n \geqq 1\}$ は正則ではない。

証明: L が正則であったと仮定して、矛盾を導く。

L は正則なので、L を受理する DFA A が存在する。A の状態集合を $q_1,q_2,\ldots,q_m$ とする(mは有限)。$n=m+1$のとき、鳩ノ巣原理から、

$$0,00,0^3,0^4,\ldots,0^n$$

の中には、「Aが遷移したときに同じ状態になる、長さの異なるペア」が存在する。これらを $0^i,0^j$ とおく。つまり A は$0^i,0^j$ のどちらを読み込んだときも同じ状態 q になる。

ここで入力0^i1^j を考える。$i \neq j$なので、これは L の要素ではない。しかし A は入力0^i1^jと入力0^j1^jを区別できない。したがって、両方とも受理するか、両方とも受理しないか、どちらかしかできない。これは A が L を受理する、という仮定に反する。

したがって L は正則ではない。

4. 正則言語の性質(1)
(テキスト4.1,4.2)

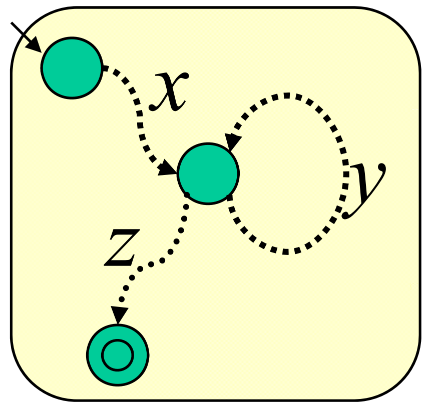
ある言語が正則でないことを示すのに使う標準的な補題

4.1. 言語が正則でないことの証明

正則言語に対する反復補題(Pumping Lemma):

– 正則言語 L に対し、以下の条件を満たす定数 n が存在する: $|w| \geqq n$ を満たす任意の文字列 $w \in L$ は、次の条件を満たす3個の部分列 $w = xyz$ に分解できる。

1. $y \neq \varepsilon$
2. $|xy| \leqq n$
3. すべての $k \geqq 0$ に対し、$xy^kz \in L$



4.1. 言語が正則でないことの証明

反復補題(Pumping Lemma):

- 正則言語 L に対し、以下の条件を満たす定数 n が存在する: $|w| \geqq n$ を満たす任意の文字列 $w \in L$ は、次の条件を満たす3個の部分列 $w = xyz$ に分解できる。

  (1) $y \neq \varepsilon$ (2) $|xy| \leqq n$ (3) $xy^kz \in L$ $(k \geqq 0)$

[証明] Lは正則言語なので、$L(A)=L$であるDFA Aが存在する。A の状態数を n とする。

長さ n 以上の L に属する任意の文字列 $w=a_1a_2\ldots a_m$ を考える。$(m \geqq n)$

A は文字列 $a_1a_2\ldots a_i$ を処理したあと、状態 p_i になるとする。(初期状態をq_0とすると$p_0=q_0$)

4.1. 言語が正則でないことの証明

反復補題(Pumping Lemma):

- 正則言語 L に対し、以下の条件を満たす定数 n が存在する: $|w| \geqq n$ を満たす任意の文字列 $w \in L$ は、次の条件を満たす3個の部分列 $w = xyz$ に分解できる。

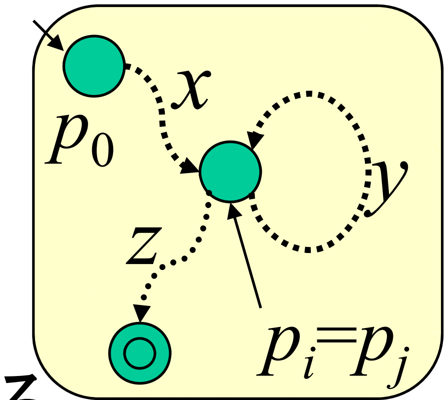
  (1) $y \neq \varepsilon$ (2) $|xy| \leqq n$ (3) $xy^kz \in L$ (k$\geqq$0)

[証明] A は文字列 $a_1a_2\ldots a_i$ を処理したあと、状態 p_i になるとする。(初期状態をq_0とすると$p_0=q_0$)
鳩ノ巣原理により、$p_0,p_1,\ldots,p_m$ の中には同じ状態 p_i, p_j が存在する。($i<j$ としてよい)

- $x = a_1,a_2,\ldots,a_i$
- $y = a_{i+1},\ldots,a_j$
- $z = a_{j+1},\ldots,a_m$

$x=\varepsilon$や$z=\varepsilon$はありえるが$y\neq\varepsilon$

と定義すると A は xy^kz ($k\geqq 0$)を受理する。



例: 言語 $L=\{0^n1^n \mid n \geqq 1\}$ は正則ではない。

反復補題による証明: L が正則であると仮定して、矛盾を導く。

L は正則なので、反復補題より、以下の条件を満たす定数 m が存在する: $|w| \geqq m$ を満たす任意の文字列 $w \in L$ は、次の条件を満たす3個の部分列 $w = xyz$ に分解できる。

$(1) y \neq \varepsilon$ (2) $|xy| \leqq m$ (3) $xy^kz \in L$ $(k \geqq 0)$

ここで文字列$w=0^m1^m$を考える。wを上記の条件を満たすような部分列xyzに分解する。$y \neq \varepsilon$かつ$|xy| \leqq m$なので、$y=0^i$ $(i \geqq 1)$ となる。

すると、$xyz = 0^m1^m$ なので $xyyz = 0^{m+i}1^m$ である。反復補題から、$xyyz \in L$ となるが、実際には $xyyz \notin L$ であるので矛盾。

したがって L は正則ではない。

4. 正則言語の性質(1):
(テキスト4.1,4.2)

4.2. 正則言語に関する閉包性

- 閉包性…集合/言語が演算に関して閉じていること。
  - 正則言語にある操作/演算を加えて、新しい言語を作ったとき、それがまた正則になっているなら、
    - 正則言語はその操作/演算に関して閉じている

  という。この性質を閉包性という。

4.2. 正則言語に関する閉包性

– 正則言語は以下の閉包性を持つ。

① 正則言語 L_1, L_2 について $L_1 \cup L_2$ は正則
② L_1, L_2 について $L_1 \cap L_2$ は正則
③ 正則言語の補集合は正則
④ L_1, L_2 について $L_1 - L_2$ は正則
⑤ 正則言語の反転は正則
⑥ L_1 について $L_1{}^*$ は正則
⑦ L_1, L_2 の連接は正則
⑧ 正則言語の準同型の像は正則
⑨ 正則言語の逆準同型の像は正則

正則言語における4つの証明手法

この授業では範囲外

4.2. 正則言語に関する閉包性

① 正則言語 L_1, L_2 について $L_1 \cup L_2$ は正則

[証明手法1] 正則表現を使ったもの

L_1, L_2は正則言語なので、$L(E_1)=L_1, L(E_2)=L_2$ を満たす正則表現が存在する。$((E_1)+(E_2))$は正則表現で、かつ明らかに $L(((E_1)+(E_2)))=L_1 \cup L_2$ が成立する。

4.2. 正則言語に関する閉包性
③ 正則言語の補集合は正則

[補集合とは] 言語 L の補集合 $\overline{L}=\{ w \mid w \notin L\}$

[証明手法2] オートマトンを使ったもの

言語 L が正則なら、L を受理するDFA $A=(Q,\Sigma,\delta,q,F)$ が存在する。このとき、A の受理状態とそれ以外を入れ替えた DFA $\overline{A}=(Q,\Sigma,\delta,q,Q-F)$ は $\overline{L}$ を受理する。

4.2. 正則言語に関する閉包性

② L_1, L_2 について $L_1 \cap L_2$ は正則

[証明手法3]

ド・モルガンの定理より、

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

したがって L_1, L_2 が正則なら①,③より、$L_1 \cap L_2$ も正則

4.2. 正則言語に関する閉包性

④ L_1 ,L_2 について $L_1 - L_2$ は正則

($L_1 - L_2 = L_1 \cap \overline{L_2}$ なので手法3でもOK)

[証明手法4(直積構成法)]

① L_1, L_2 を受理する DFA を M_1, M_2 とする。

② $L_1 - L_2$を受理するDFA Mは、入力を読みながら、

- その入力に対する M_1 の状態遷移
- その入力に対する M_2 の状態遷移

を同時に模倣する。

③ 入力を読み終えた時点で M_1 が受理かつ M_2 が受理でないなら M は受理。

4.2. 正則言語に関する閉包性

⑤ 正則言語の反転は正則

[反転とは]

文字列 $w=x_1x_2\ldots x_{\mathrm{k}}$ の反転(Reverse) $w^R=x_k\ldots x_2x_1$

言語 L の反転 $L^R=\{\, w \mid w^R \in L\}$

A がDFA
でも A^R は
NFA

[証明]

L を受理するDFA A に対し、

①Aの受理状態を一つにし、

②Aの遷移をすべて逆転し、

③受理状態と初期状態を入れ替えた

ε-NFA A^RはL^Rを受理する。

4.2. 正則言語に関する閉包性

⑥ L_1 について $L_1{}^*$ は正則

⑦ L_1, L_2 の連接は正則

L_1, L_2 を表現する正則表現 E_1, E_2 に対し、

⑥ $(E_1)^*$

⑦ $(E_1)(E_2)$

でOK.

4. 正則言語の性質(2):
(テキスト4.3,4.4)

4.3. 正則言語に関する決定問題

言語に関する基本的な問題

1. 与えられた言語 L が $L=\Phi$か?または $L=\Sigma^*$ か?

   例) $L_1=\{ w \mid w$ に含まれる0の数は偶数$\}$　$L_1 \cap L_2=\Phi$?

   $L_2=\{ w \mid w$ に含まれる0の数は奇数$\}$　$L_1 \cup L_2=\Phi$?

2. 与えられた語 w が言語 L に属するか。

   例) $0000111101011000 \in L_1$?

3. 二つの言語 L_1, L_2 は同じか。

   0と1が交互に現れる文字列

   例) $(\mathbf{01})^* + (\mathbf{10})^* + \mathbf{1}(\mathbf{01})^* + \mathbf{0}(\mathbf{10})^* = (\mathbf{1}+\varepsilon)(\mathbf{01})^*(\mathbf{0}+\varepsilon)$?

4. 正則言語の性質(2): (テキスト4.3,4.4)

[余談]
現実的には
NFA→DFAで
指数関数的に
状態数が増える
ことはあまりない。
ただし人工的に
そうした例を構成
することはできる。

4.3. 正則言語に関する決定問題

4.3.1. 異なる表現の間の変換

1. NFA→DFAのコスト(時間): $O(n^3 2^n)$
2. DFA→NFAのコスト: $O(n)$
3. オートマトン→正則表現: $O(n^3 4^n)$
4. 正則表現→ε-NFA: $O(n)$

多項式/指数関数かどう
かはシビアな問題

最悪の場合は
指数関数的(=爆発的)
に増加

4. 正則言語の性質(2):
(テキスト4.3,4.4)

4.4. オートマトンの等価性と最小性

3. 二つの言語 L_1, L_2 は同じか。

例) **(01)*** + **(10)*** + **1(01)*** + **0(10)*** と (**1**+ε)(**01**)*(**0**+ε) は同じ言語か?

[目標]

- DFA には「最小」のものがある
- 最小のDFAは本質的に1つしかない
- 最小のDFAは計算によって求めることができる
- <u>二つの正則言語の同値性を効率よく判定できる。</u>

4. 正則言語の性質(2):
(テキスト4.3,4.4)

4.4. オートマトンの等価性と最小性

4.4.1. 状態の同値性の判定

DFA における状態 p, q が同値(equivalent)

$\Updownarrow$

すべての文字列 w に対して、

$\hat{\delta}(p,w)$が受理状態$\Leftrightarrow\hat{\delta}(q,w)$が受理状態

が成立する

必ずしも同じ状態で
なくてもOK

4. 正則言語の性質(2):
(テキスト4.3,4.4)

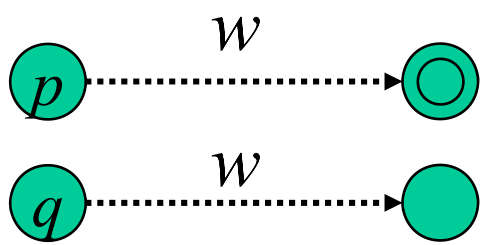
4.4. オートマトンの等価性と最小性

4.4.1. 状態の同値性の判定

DFA における状態 p, q が区別可能(distinguishable)

$\Updownarrow$

状態 p,q が<u>同値ではない</u>



$\Updownarrow$

ある文字列 w が存在して、以下が成立:

$\hat{\delta}(p,w)$, $\hat{\delta}(q,w)$ の一方は受理状態で、
他方はそうでない

4. 正則言語の性質(2):
(テキスト4.3,4.4)

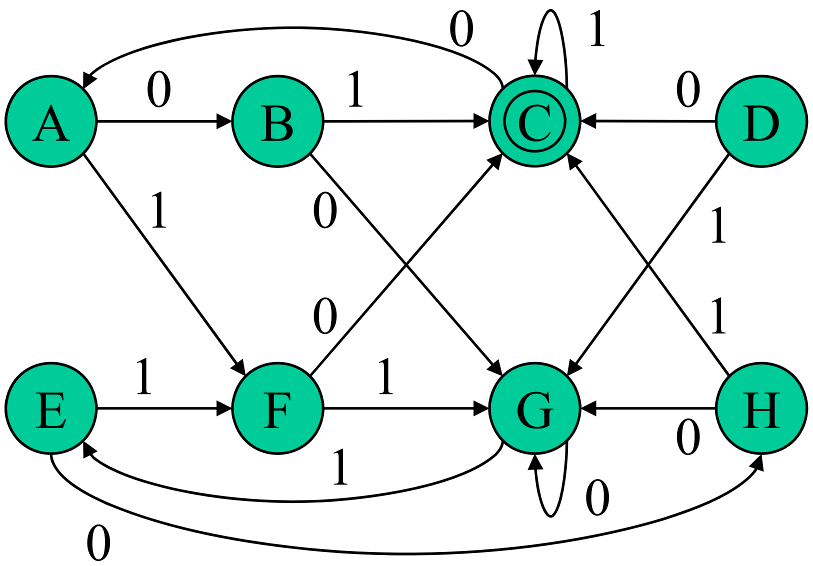
4.4. オートマトンの等価性と最小性

4.4.1. 状態の同値性の判定

例) 受理状態の集合をX={C}と書く。

$$\hat{\delta}(C,\varepsilon) \in X$$

$$\hat{\delta}(G,\varepsilon) \notin X$$



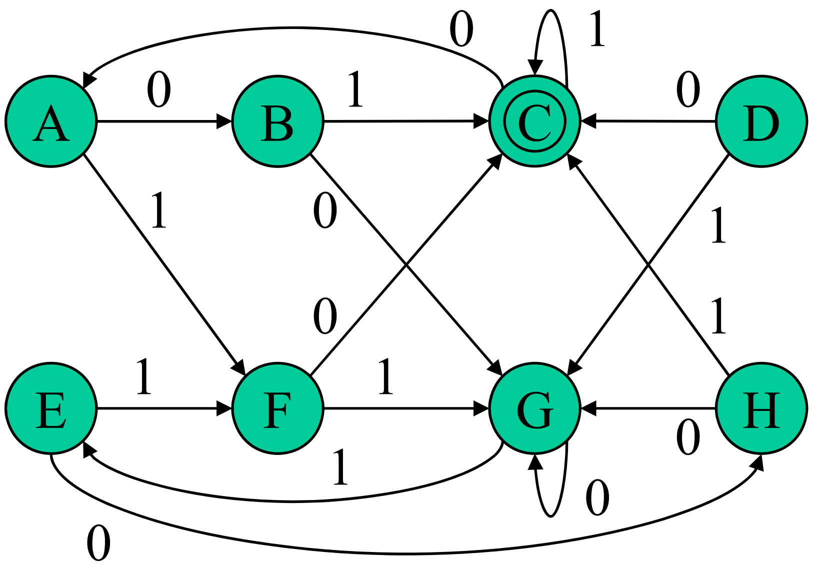
CとGは区別可能

4. 正則言語の性質(2): (テキスト4.3,4.4)

4.4. オートマトンの等価性と最小性

4.4.1. 状態の同値性の判定

例) 受理状態の集合をX={C}と書く。



$$\hat{\delta}(A,\varepsilon)\notin X,\ \hat{\delta}(G,\varepsilon)\notin X$$

$$\hat{\delta}(A,0)\notin X,\ \hat{\delta}(G,0)\notin X$$

$$\hat{\delta}(A,1)\notin X,\ \hat{\delta}(G,1)\notin X$$

$$\hat{\delta}(A,01)\in X,\ \hat{\delta}(G,01)\notin X$$

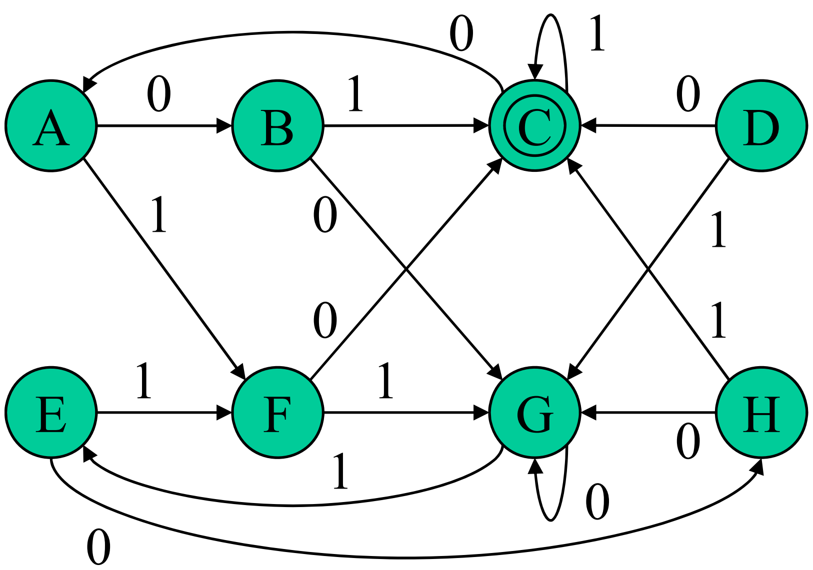
AとGは区別可能

4. 正則言語の性質(2): (テキスト4.3,4.4)

4.4. オートマトンの等価性と最小性

4.4.1. 状態の同値性の判定

例)　受理状態の集合をX={C}と書く。



$$\hat{\delta}(A,\varepsilon) \notin X, \hat{\delta}(E,\varepsilon) \notin X$$
$$\hat{\delta}(A,1) = \hat{\delta}(E,1) = F$$
$$\hat{\delta}(A,0) \notin X, \hat{\delta}(E,0) \notin X$$
$$\hat{\delta}(A,00) = \hat{\delta}(E,00) = G$$
$$\hat{\delta}(A,01) = \hat{\delta}(E,01) = C$$

AとEは同値

4. 正則言語の性質(2):
(テキスト4.3,4.4)

実装上の工夫:
区別可能なペアから逆に構成

4.4. オートマトンの等価性と最小性

4.4.1. 状態の同値性の判定

同値な状態のペアを求める穴埋めアルゴリズム
(Table-filling algorithm)

1. 状態 p が受理状態で、q が受理状態ではないとき、$\{p,q\}$ は区別可能
2. 状態 p, q と、ある入力文字 a に対して、$r=\delta(p,a)$, $s=\delta(q,a)$ としたとき、$\{r,s\}$ が区別可能なら $\{p,q\}$ も区別可能
3. ステップ2を繰り返し適用し、それ以上変化しなくなったら終了

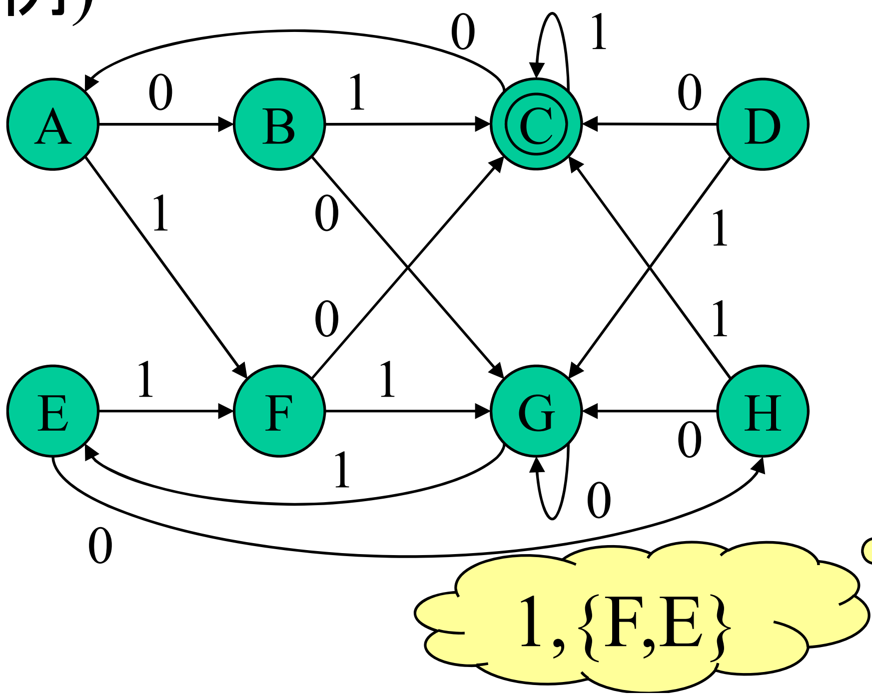
4. 正則言語の性質(2):
(テキスト4.3,4.4)

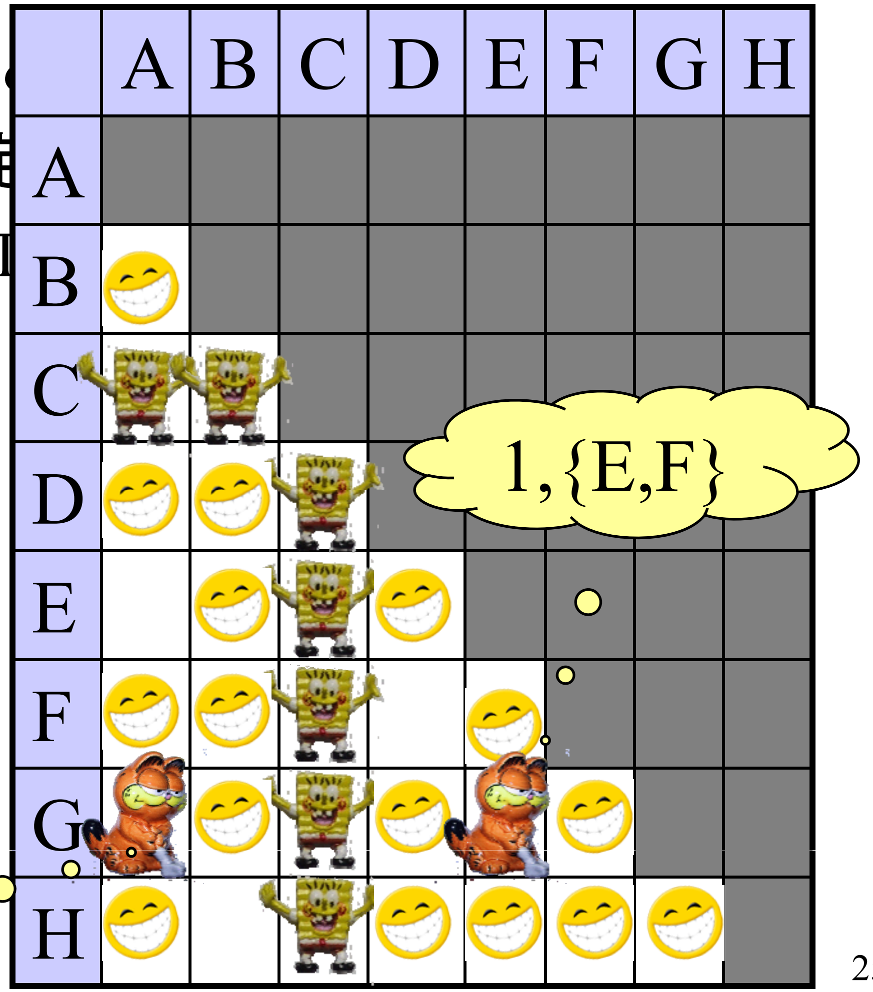
4.4. オートマトンの等価性

4.4.1. 状態の同値性の判定

穴埋めアルゴリズム(T

例)





4. 正則言語の性質(2):
(テキスト4.3,4.4)

4.4. オートマトンの等価性と最小性

4.4.1. 状態の同値性の判定

穴埋めアルゴリズム(Table-filling algorithm)

2. 状態 p, q と、ある入力文字 a に対して、$r=\delta(p,a)$, $s=\delta(q,a)$ としたとき、$\{r,s\}$ が区別可能なら $\{p,q\}$ も区別可能

- $\{r,s\}$が区別可能 $\Rightarrow$ ある文字列 w があって、$\delta(r,w)$ と $\delta(s,w)$ が一方は受理状態で、他方はそうではない
- 文字列 aw が状態 p と q を区別可能にする。

$\Rightarrow$「区別可能」と判断されたものは、区別可能。

4. 正則言語の性質(2):
(テキスト4.3,4.4)

4.4. オートマトンの等価性と最小性

4.4.1. 状態の同値性の判定

穴埋めアルゴリズム(Table-filling algorithm)の正当性

- 区別可能なものは必ず区別可能と判断される
- 同値なペアは最後まで何も判断されず、空白となる

[定理] 穴埋めアルゴリズムによって区別されない二つの状態 p, q は同値である。

[証明] 背理法による。詳細はテキストを参照のこと。

4. 正則言語の性質(2):
(テキスト4.3,4.4)

4.4. オートマトンの等価性と最小性

4.4.2 正則言語の等価性の判定

与えられた正則言語L_1, L_2 の等価性は次の手順で判定できる。

1. L_1, L_2に対する DFA A_1, A_2 を構成する
2. 二つの DFA A_1, A_2 を全体として一つの DFA A とみなす。
3. A について穴埋めアルゴリズムを実行
4. A_1の初期状態とA_2の初期状態が同値なら $L_1=L_2$。そうでないなら $L_1 \neq L_2$。

素直に実装すると$O(n^4)$、工夫すると$O(n^2)$

4.4. オートマトンの等価性と最小性

4.4.3. DFA の最小化

[定理] 与えられた正則言語に対して、その言語を受理するDFA の中で、状態数が最小のDFAを一意的に構成することができる。

[証明] 省略。テキスト参照のこと。