

3. 正則表現: (テキスト3.1,3.2)

3.1. 正則表現

- **正則表現**(または**正規表現**)とは、文字列の集合(=言語)を有限個の記号列で表現する方法の1つ

例: $(01)^*$...「01を繰り返す文字列」

$0(0+1)^*$...「0の後に0か1が繰り返す文字列」

つまり

$(01)^* = \{\epsilon, 01, 0101, 010101, 01010101, \dots\}$

$0(0+1)^* = \{0, 00, 01, 000, 001, 010, 011, 0000, \dots\}$

- UNIX系の人にはおなじみ...grep, emacs, awk, perl, ...
- Windows系の人にも...ファイル名のワイルドカードなど

3. 正則表現: (テキスト3.1,3.2)

3.1. 正則表現の直感的な定義と意味

– 文字や文字列はそのまま解釈:

- $a \rightarrow \{a\}$

- $ab \rightarrow \{ab\}$

– 「+」は「または」の意味:

- $ab+a \rightarrow \{ab,a\}$

– 「()」はグループ化

– 「*」は「0回以上の繰り返し」の意味

- $(ab)^* \rightarrow \{\varepsilon, ab, abab, ababab, \dots\}$

ちょっと複雑な例:

$$((ab)^*c)+(a^*)$$

$$\rightarrow \{\varepsilon, a, c, aa, aaa, abc, aaaa, aaaaa, ababc, \dots\}$$

3. 正則表現: (テキスト3.1,3.2)

3.1.1. 正則表現の演算

1. **和集合(union)**: 二つの言語 L, M の和集合 $L \cup M$ は、 L か M のどちらかに含まれる要素の集合。
 - 例: $\{abc\} \cup \{a,b,c\} = \{a,b,c,abc\}$
2. **接続(concatenation)**: 二つの言語 L, M の接続 LM (または $L \cdot M$) は、それぞれの要素を一つづつとってつなげたものの集合
 - 例: $\{abc\} \{a,b,c\} = \{abca, abcb, abcc\}$
3. **閉包(closure)**: ある言語 L の閉包 L^* は、 L の要素を0個以上接続したものの集合
 - 例: $\{a,b,c\}^* = \{\varepsilon, a, b, c, aa, ab, ac, ba, bb, bc, ca, cb, cc, aaa, \dots\}$

3. 正則表現: (テキスト3.1,3.2)

3.1.1. 正則表現の演算

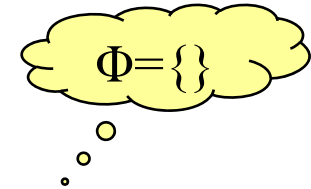
2.5. 言語の接続の補足: LL は L^2 , LLL は L^3 と書くことがある。

- 例: $\{a,ab\}^2 = \{a,ab\}\{a,ab\} = \{aa,aab,aba,abab\}$
- 定義: $L^0 := \{\varepsilon\}$, $L^1 := L$, $L^k := L^{k-1}L$ ($k>1$)

3.5. 言語の閉包の補足: 2.5 より、 L^* は以下の定義と同値。

$$L^* := \bigcup_{i=0}^{\infty} L^i$$

3. 正則表現: (テキスト3.1,3.2)



3.1.2. 正則表現の構成

正則表現 E とそれが表現する言語 $L(E)$ の定義

1. 定数 ε と Φ は正則表現で、 $L(\varepsilon) = \{\varepsilon\}$, $L(\Phi) = \Phi$.
2. 記号 a に対して、 a は正則表現で、 $L(a) = \{a\}$.
3. E と F が正則表現のとき、
 1. $E+F$ は正則表現。定義される言語; $L(E+F) = L(E) \cup L(F)$
 2. EF (または $E \cdot F$) は正則表現。定義される言語;
 $L(EF) = L(E)L(F)$
 3. E^* は正則表現。定義される言語; $L(E^*) = (L(E))^*$
 4. (E) は正則表現。定義される言語; $L((E)) = L(E)$

3. 正則表現: (テキスト3.1,3.2)

3.1.2. 正則表現の構成

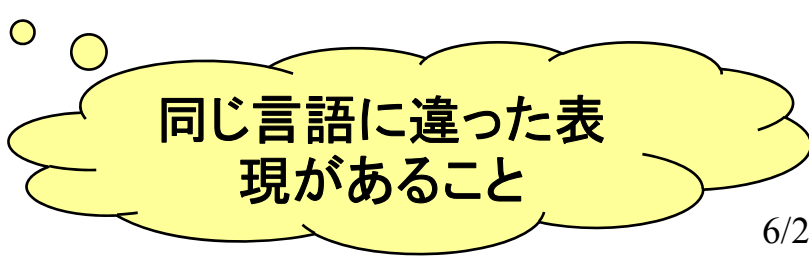
例: 「0と1が交互に現れる文字列」という言語

1. 発想(1): (a)01の繰り返し か (b)10の繰り返し か (c)1のあとに(a) か (d) 0のあとに(b)

- $(01)^* + (10)^* + 1(01)^* + 0(10)^*$

2. 発想(2): 01の繰り返しの前に1かεを追加、後に0かεを追加

- $(1+\epsilon)(01)^*(0+\epsilon)$



同じ言語に違った表現があること

3. 正則表現: (テキスト3.1,3.2)

3.1.2. 正則表現の演算順序

すべて()で明記してもよいが、曖昧でなく定義すれば、()は適宜省略できる。

1. 同じ演算は左から右: $abc = (ab)c$,
 $a+b+c=(a+b)+c$
2. *は最優先: $ab^*=a(b)^*\neq(ab)^*$
3. ・は2番目: $a+bc = a+(bc)\neq(a+b)c$
4. +は最後: $a+bc^*+d = (a+(b(c^*))) + d$

3. 正則表現: (テキスト3.1,3.2)

3.2. 有限オートマトンと正則表現

- **ゴール**: 正則表現で表現できる言語 = オートマトンで受理できる言語
 1. 与えられた正則表現から、 ϵ -NFAが構成できること
 2. 与えられたDFAから正則表現が構成できること
 - 2'. 与えられた ϵ -NFAから正則表現が構成できること

- ϵ -NFAは(見かけ上)表現力が高い
- DFAは構成要素が(見かけ上)少ない

3. 正則表現: (テキスト3.1,3.2)

3. 2. 3. 正則表現 \rightarrow ϵ -NFA

正則表現とそれが表現する言語の定義

1. ϵ, Φ , 記号 a は正則表現; $L(\epsilon)=\epsilon, L(\Phi)=\Phi, L(a) = \{a\}$.
2. 正則表現 E と F に対し、
 1. $E+F$ は正則表現; $L(E+F) = L(E) \cup L(F)$
 2. EF (または $E \cdot F$)は正則表現; $L(EF) = L(E)L(F)$
 3. E^* は正則表現; $L(E^*) = (L(E))^*$
 4. (E) は正則表現; $L((E)) = L(E)$

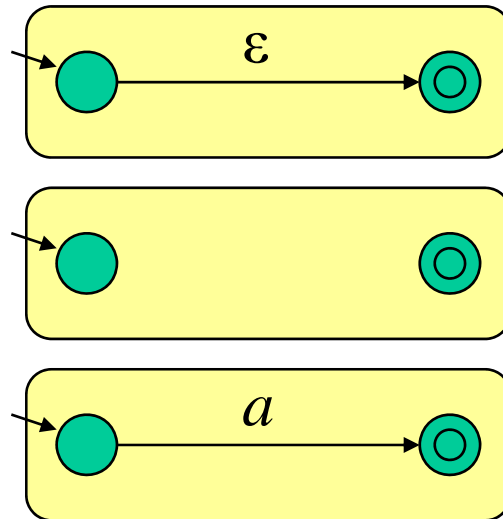
から直接 ϵ -NFAを構成する。

受理状態が1つしかない

3. 正則表現: (テキスト3.1,3.2)

3. 2. 3. 正則表現 \rightarrow ϵ -NFA

1. ϵ, Φ , 記号 a は正則表現; $L(\epsilon)=\epsilon, L(\Phi)=\Phi, L(\mathbf{a}) = \{a\}$.

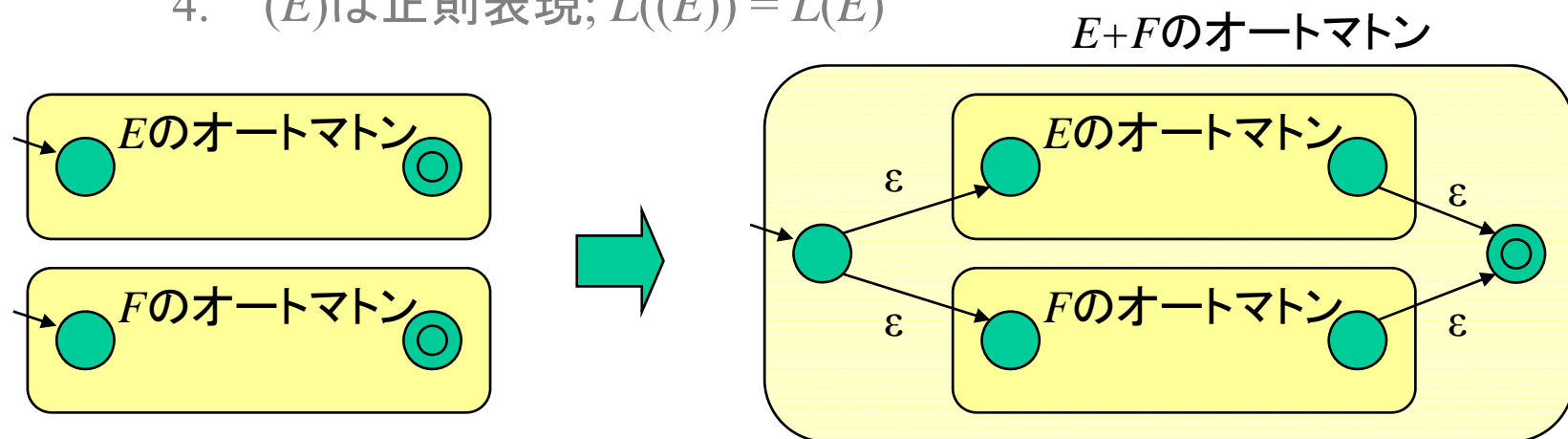


3. 正則表現: (テキスト3.1,3.2)

3. 2. 3. 正則表現 $\rightarrow \epsilon$ -NFA

2. 正則表現 E と F に対し、

1. $E+F$ は正則表現; $L(E+F) = L(E) \cup L(F)$
2. EF (または $E \cdot F$) は正則表現; $L(EF) = L(E)L(F)$
3. E^* は正則表現; $L(E^*) = (L(E))^*$
4. (E) は正則表現; $L((E)) = L(E)$



3. 正則表現: (テキスト3.1,3.2)

3. 2. 3. 正則表現 \rightarrow ϵ -NFA

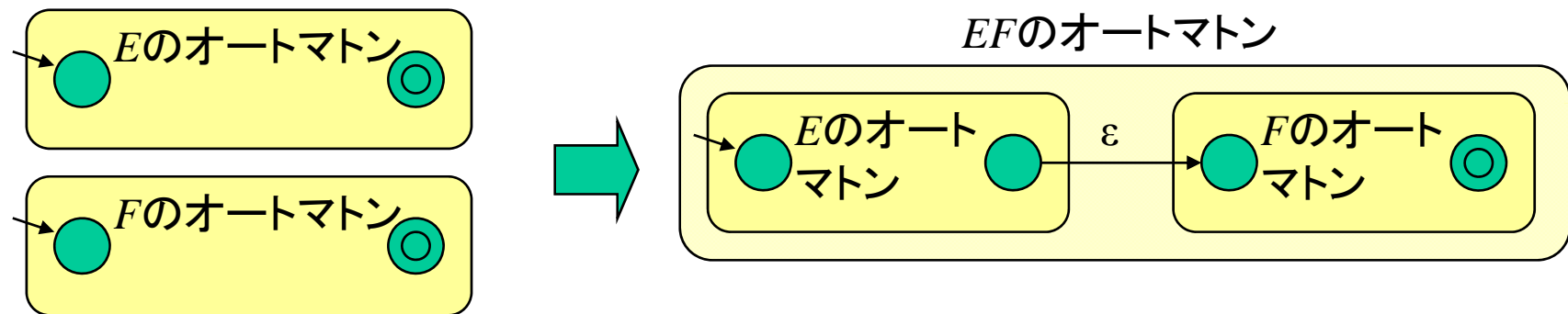
2. 正則表現 E と F に対し、

1. $E+F$ は正則表現; $L(E+F) = L(E) \cup L(F)$

2. EF (または $E \cdot F$)は正則表現; $L(EF) = L(E)L(F)$

3. E^* は正則表現; $L(E^*) = (L(E))^*$

4. (E) は正則表現; $L((E)) = L(E)$



3. 正則表現: (テキスト3.1,3.2)

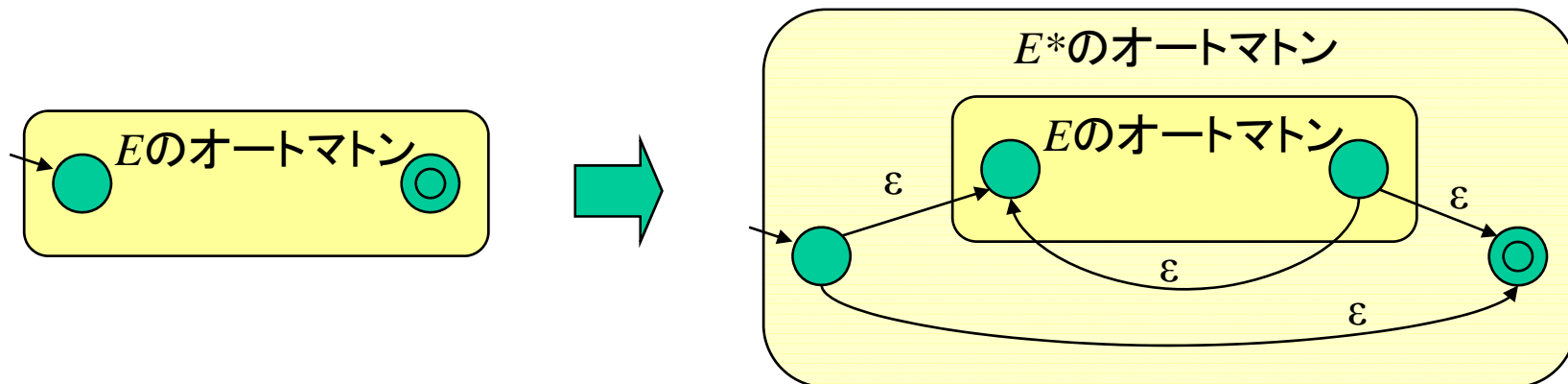
どの規則も受理
状態が1つの
オートマトンしか
作らない

3. 2. 3. 正則表現 $\rightarrow \epsilon$ -NFA

2. 正則表現 E と F に対し、

1. $E+F$ は正則表現; $L(E+F) = L(E) \cup L(F)$
2. EF (または $E \cdot F$) は正則表現; $L(EF) = L(E)L(F)$
3. E^* は正則表現; $L(E^*) = (L(E))^*$
4. (E) は正則表現; $L((E)) = L(E)$

特に何も
しない

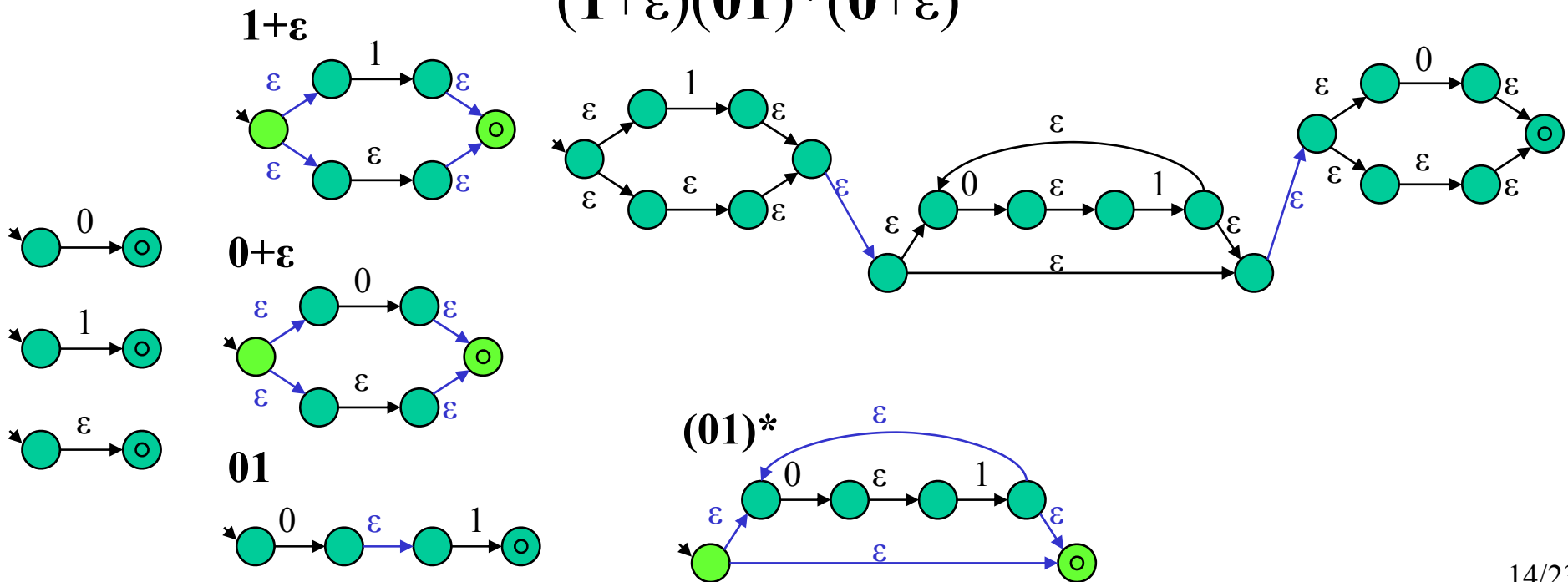


3. 正則表現: (テキスト3.1,3.2)

3. 2. 3. 正則表現 $\rightarrow \epsilon$ -NFA

例: 「0と1が交互に出てくる文字列」の正則表現

$$(1+\epsilon)(01)^*(0+\epsilon)$$



3. 正則表現: (テキスト3.1,3.2)

3. 2. *. ϵ -NFA \rightarrow 正則表現

補題: 任意の ϵ -NFA A に対し、 $L(A)=L(A')$ で、以下の条件を満たす ϵ -NFA A' が存在する。

1. 受理状態は1つで、受理状態からの遷移はない
2. 任意の状態 q に対し、初期状態から q への遷移と、 q から受理状態への遷移が存在する

3. 正則表現: (テキスト3.1,3.2)

3. 2. *. ϵ -NFA \rightarrow 正則表現

補題: 任意の ϵ -NFA A に対し、 $L(A)=L(A')$ で、以下の条件を満たす ϵ -NFA A' が存在する。

1. 受理状態は1つで、受理状態からの遷移はない
2. 任意の状態 q に対し、初期状態から q への遷移と、 q から受理状態への遷移が存在する

証明:

2. 初期状態から到達できない状態と、受理状態に到達できない状態は無関係な言葉とは無関係なので、取り除いてよい。

3. 正則表現: (テキスト3.1,3.2)

3. 2. *. ϵ -NFA \rightarrow 正則表現

補題: 任意の ϵ -NFA A に対し、 $L(A)=L(A')$ で、以下の条件を満たす ϵ -NFA A' が存在する。

1. 受理状態は1つで、受理状態からの遷移はない
2. 任意の状態 q に対し、初期状態から q への遷移と、 q から受理状態への遷移が存在する

証明:

1. レポート1の解答になるので、秘密。

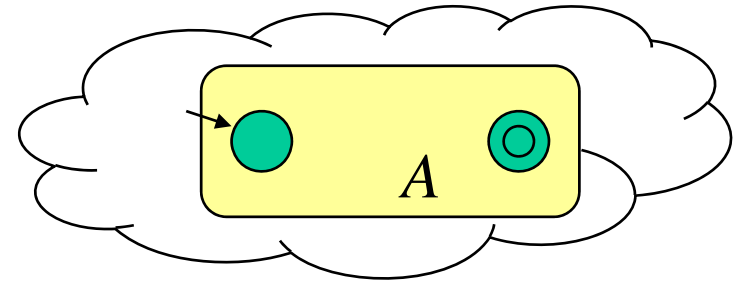
3. 正則表現: (テキスト3.1,3.2)

3. 2. *. ϵ -NFA \rightarrow 正則表現

定理: 任意の ϵ -NFA A に対し、 $L(A)=L(E)$ となる正則表現 E が存在する。

証明:

- $L(A)=\Phi$ なら、 $E=\Phi$
- 以下では $L(A)\neq\Phi$ と仮定する。 A は補題の条件を満たすと
する。
 1. 受理状態は1つで、受理状態からの遷移はない
 2. 任意の状態 q に対し、初期状態から q への遷移と、 q から受理状態への遷移が存在する



3. 正則表現: (テキスト3.1,3.2)

3. 2. *. ϵ -NFA \rightarrow 正則表現

定理: 任意の ϵ -NFA A に対し、 $L(A)=L(E)$ となる正則表現 E が存在する。

証明:

証明のアイデア:

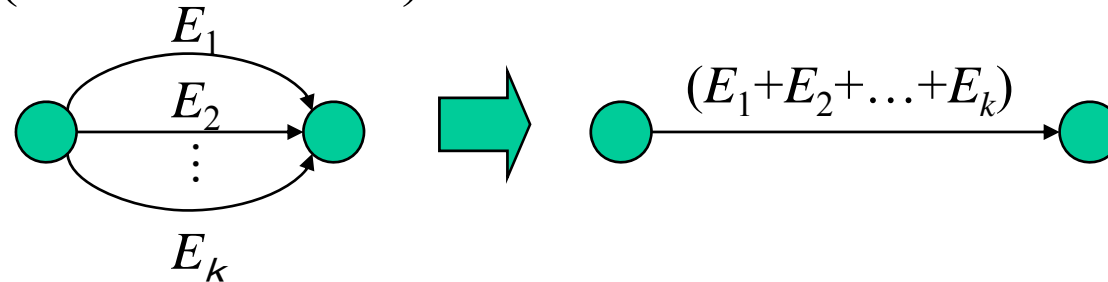
- 辺のラベルから正規表現を構築していく
- 頂点を順番に削除していく

3. 2. *. ϵ -NFA \rightarrow 正則表現

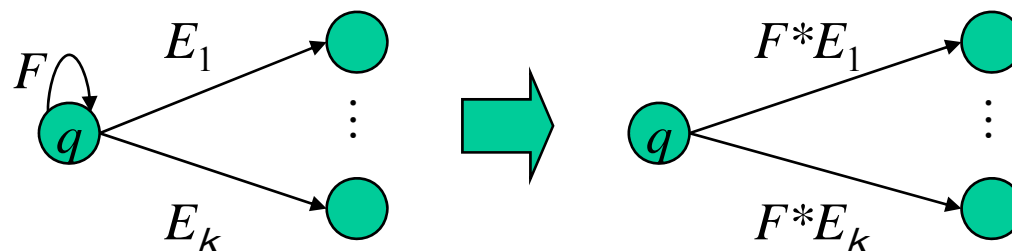
定理: 任意の ϵ -NFA A に対し、 $L(A)=L(E)$ となる正則表現 E が存在する。

証明:

T1 (多重辺の削除): 同じ端点を持つ複数の辺の一本化



T2: (ループの除去): 頂点 q から q への遷移が1本するとき



T3: (頂点 q の削除):

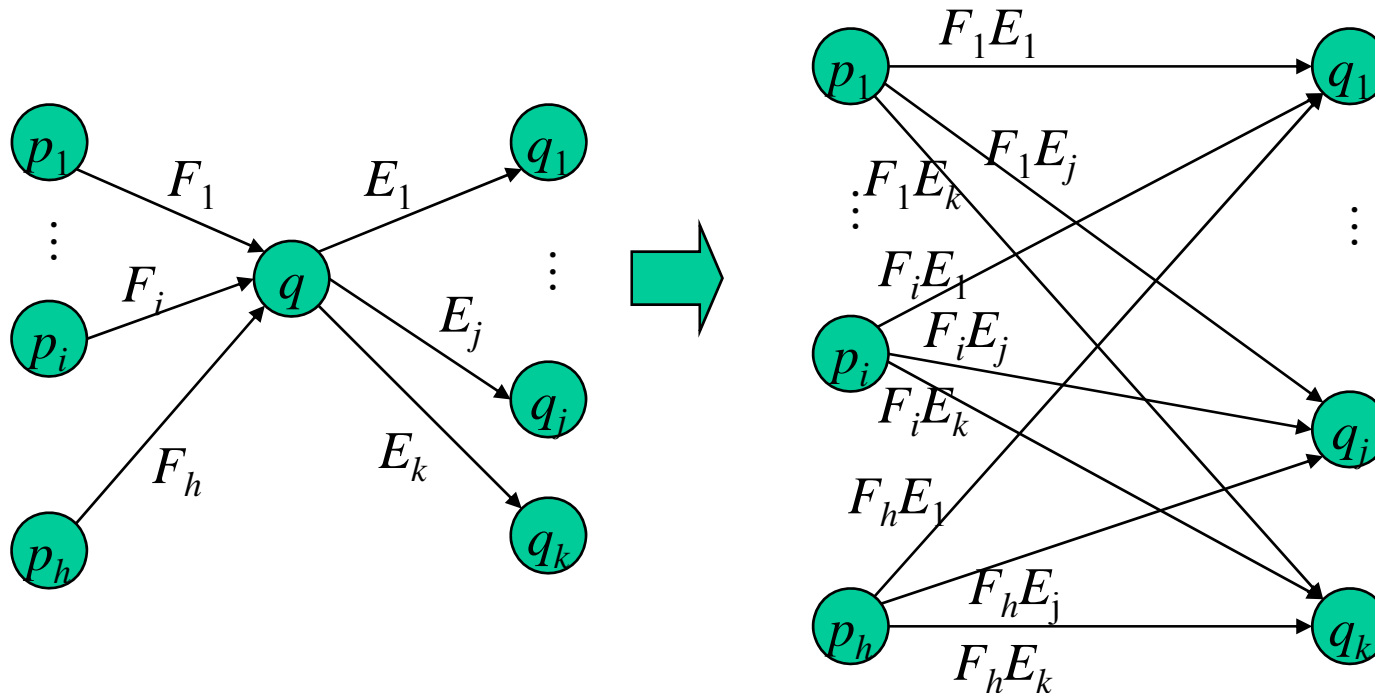
3. 2. *. ϵ -NFA \rightarrow 正則表現

定理: 任意の ϵ -NFA A に対し、 $L(A)=L(E)$ となる正則表現 E が存在する。

証明:

T3: (頂点 q の削除):

- q は初期状態、受理状態でない
- q から q への遷移はない



3. 正則表現: (テキスト3.1,3.2)

3. 2. *. ϵ -NFA \rightarrow 正則表現

定理: 任意の ϵ -NFA A に対し、 $L(A)=L(E)$ となる正則表現 E が存在する。

証明: 与えられた ϵ -NFA A に対し、

1. T1(多重辺の除去)を可能な限り適用
2. T2(ループの除去)を可能な限り適用
3. T3(頂点の削除)を適用

すると、 A の初期状態と(唯一の)受理状態以外の状態が一つ減る。これを繰り返すと、初期状態と受理状態だけのNFA A'



ができる。このときの辺のラベル E が求める正規表現となる。

3. 正則表現: (テキスト3.1,3.2)

3.2. *. ϵ -NFA \rightarrow 正則表現

例:

1. まず a が 0 個以上 続き、
2. 次に $[b$ が 0 個以上] または $[c$ が 0 個以上] 続き、
3. 最後に d が 0 個以上 続く

