3. Regular Expression:
(Text 3.1.3.2)
3.1. Regular Expression
- Regular Expression describes a set of words (=language) by finite characters
  \[ (\text{Ex:} \{a, b, c\}^* \text{ means } \{a, b, c, \ldots\}^*) \]
- It's familiar if you are UNIX-user...grep, emacs, awk, perl, ...
- So-called 'Wild-card' in filenames on Windows

3.1.1. Operations over regular expressions
1. The union \( L \cup M \) contains elements in \( L \) or \( M \).
   \[ (\text{Ex:} [A, B, C] \cup [a, b, c] = [a, b, c, A, B, C]) \]
2. The concatenation of two languages \( L \) and \( M \) is denoted by \(LM\) (or \( L \cdot M \)) is a set of all combinations of two elements from \(L\) and \(M\), respectively.
   \[ (\text{Ex:} [A, B, C][a, b] = [Aa, Ab, Ac, Ba, Bb, Bc, Ca,Cb,Cc]) \]
3. The closure \( L^* \) of a language \( L \) is a set of all concatenations of any number of elements in \( L \).
   \[ (\text{Ex:} [a,b,c]^* = \{a^0, a^1, a^2, ab, ac, bc, aab, abc, abbc, \ldots\}) \]
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. 言語の連接の補足: $LL$ は $L^2$, $LLL$ は $L^3$ と書くことがある。

- 定義: $L^2 = \{ wL | w \in L \}$
- 定義: $L^3 = \{ wL^2 | w \in L \}$

#### 3.5. 言語の閉包の補足: $L^*$ は以下の定義と同値。

3.1.2. 正則表現の構成

#### 3.1.2. Construction of a regular expression

Definition of a regular expression

Ex: A language that is the set of words such that 0 and 1 appear alternately.

1. Two regular expressions $L_1, L_2$ are regular expressions that represent
   - $L_1 \cup L_2$ or $L_1 L_2$ or $L_1^*$
   - $L_1 a, ab, aab, aba, abab$
   - $L_1^*$

#### 3. Regular Expression:
(テキスト3.1.3.2)

### 3.1.2. Construction of a regular expression

Definition of a regular expression $E$ and corresponding language $L(E)$

1. Two constants $\emptyset$ and $\Sigma$ are regular expressions that represent
   - $L(\emptyset) = \emptyset, L(\Sigma) = \Sigma$
2. For a symbol $a$, $a$ is regular expression that represents $L(a) = \{a\}$
3. For two regular expressions $E$ and $F$,
   - $E F$ is R.E. which represents $L(E) L(F)$
   - $E F = \{ a \ b \mid a \in L(E), b \in L(F) \}$

#### 3.5. Comment on Closure: 由 2.5, $L^*$ can be defined as follows.

$L^* := \bigcup_{i=0}^{\infty} L^i$
3. Regular Expressions: (Text 3.1,3.2)

3.1.2. Priority of the operations

We can omit some ()s if we define the priority of the operations below:
1. From left to right for the same operations:
   - \( abc = (ab)c \)
   - \( a+b=c = (a+b)c \)
2. \( \ast \) has top priority:
   - \( ab*=(ab)* \)
3. \( \ast \) has the second priority:
   - \( a+bc = a+(bc) \)
4. \( \ast \) is the last:
   - \( a+bc+d = (a+(b(c)))+d \)

3. Regular Expression: (Text 3.1,3.2)

3.2. Finite automata and regular expressions

- Goal: Class of languages represented by regular expressions = Class of languages accepted by automata
1. For any given regular expression, we can construct an NFA that accepts the same language.
2. For any given DFA, we can construct a regular expression that represents the same language.
   - For any given an NFA, we can construct a regular expression that represents the same language.

- NFA seems more descriptive
- DFA has simpler structure than NFA
3. 正則表現:
(テキスト3.1,3.2)
3. 2. 3. 正則表現 $\Sigma \cdot \Delta$ -NFA

1. $\Sigma$, $\Delta$, 記号が正則表現: $L(\Sigma) = \{ \), $L(\Delta) = \{ [\}。

2. $E$ と $F$ の NFA に対し
   1. $E + F$ は正則表現: $L(E + F) = L(E) \cup L(F)$
   2. $EF$ は正則表現: $L(EF) = L(E) \cdot L(F)$
   3. $E^*$ は正則表現: $L(E^*) = \{E\}^*$
   4. $E^1$ は正則表現: $L(E^1) = L(E)$

3. 正則表現:
(テキスト3.1,3.2)
3. 2. 3. 正則表現 $\Sigma \cdot \Delta$ -NFA

2. $E$ と $F$ の NFA に対し
   1. $E + F$ は正則表現: $L(E + F) = L(E) \cup L(F)$
   2. $EF$ は正則表現: $L(EF) = L(E) \cdot L(F)$
   3. $E^*$ は正則表現: $L(E^*) = \{E\}^*$
   4. $E^1$ は正則表現: $L(E^1) = L(E)$

3. 正則表現:
(テキスト3.1,3.2)
3. 2. 3. 正則表現 $\Sigma \cdot \Delta$ -NFA

2. $E$ と $F$ の NFA に対し
   1. $E + F$ は正則表現: $L(E + F) = L(E) \cup L(F)$
   2. $EF$ は正則表現: $L(EF) = L(E) \cdot L(F)$
   3. $E^*$ は正則表現: $L(E^*) = \{E\}^*$
   4. $E^1$ は正則表現: $L(E^1) = L(E)$

3. 正則表現:
(テキスト3.1,3.2)
3. 2. 3. 正則表現 $\Sigma \cdot \Delta$ -NFA

2. $E$ と $F$ の NFA に対し
   1. $E + F$ は正則表現: $L(E + F) = L(E) \cup L(F)$
   2. $EF$ は正則表現: $L(EF) = L(E) \cdot L(F)$
   3. $E^*$ は正則表現: $L(E^*) = \{E\}^*$
   4. $E^1$ は正則表現: $L(E^1) = L(E)$

3. 正則表現:
(テキスト3.1,3.2)
3. 2. 3. 正則表現 $\Sigma \cdot \Delta$ -NFA

2. $E$ と $F$ の NFA に対し
   1. $E + F$ は正則表現: $L(E + F) = L(E) \cup L(F)$
   2. $EF$ は正則表現: $L(EF) = L(E) \cdot L(F)$
   3. $E^*$ は正則表現: $L(E^*) = \{E\}^*$
   4. $E^1$ は正則表現: $L(E^1) = L(E)$
### 3. Regular Expression: (テキスト3.1.3.2)

3. 2. 3. Regular Expression △ △-NFA

For R.E. \( E \) and \( F \),

1. \( E + F \) is R.E.; \( L(E + F) = L(E) \cup L(F) \)
2. \( EF \) or \( E \cdot F \) is R.E.; \( L(EF) = L(E)L(F) \)
3. \( E^* \) is R.E.; \( L(E^*) = (L(E))^* \)
4. \( (E) \) is R.E.; \( L((E)) = L(E) \)

#### Example:

R.E. of strings such that 0 and 1 appear alternately:

\[(1+0)(01)^*(0+1)\]
3. 正則表現:
(テキスト1.3.2)

3. 2. *. □-NFA □正則表現
補題: 任意の□-NFA Aに対し, L(A)=L(E)で, 以下の条件を満たす□-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. 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受理する言語とは無関係なので, 取り除いてよい。

3. 2. *. □-NFA □ Regular Expression
Lemma: For any □-NFA A, there exists an □-NFA A' with L(A)=L(E) such that
1. A' contains exactly one accepting state, and there are no transitions from the accepting state, and
2. for any state q, there is a path from the initial state, and there is a path to the accepting state.
Proof:
2. The other states are redundant, or they have nothing to accept the language. Hence we can remove them.

3. 2. *. □-NFA □ Regular Expression
Lemma: For any □-NFA A, there is a regular expression E such that L(A)=L(E).
Proof:
1. □-NFA A satisfies the following conditions by Lemma.
2. When L(A)=L(E), we have E=□. Thus we assume that L(A)=□. We moreover suppose that A satisfies the following conditions by Lemma.
3. A contains exactly one accepting state, and there are no transitions from the accepting state, and
4. for any state q, there is a path from the initial state, and there is a path to the accepting state.
3. 正則表現: (テキスト1.3.2)

3.2. *. □-NFA □正則表現

定理: 任意の □-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:
- 辺のラベルに正規表現を構築していく
- 頂点を順番に削除していく

【注意】構築途中で現れる "NFA" は正規にはNFAではない
(∴NFA は実のラベルはアルファベット文字しか使われていない)

3.2. *. □-NFA □ Regular Expression

Theorem: For any given □-NFA \( A \), there is a regular expression \( E \) such that \( L(A)=L(E) \).

Proof:

Idea of the proof:
- We construct the regular expression as the label on an edge of \( A \), and
- We remove states step by step.

【Note】The "NFA" during the process is not real NFA.
(Only one alphabet is allowed as a label on an edge.)
3. Regular Expression:
(Text 3.1,3.2)

3. 2. * ⊇ -NFA ⊇ Regular Expression

Theorem: For any given ⊇ -NFA $A$, there is a regular expression $E$ such that $L(A) = L(E)$.

Proof: For any given ⊇ -NFA $A$,
1. apply T1(Remove multi-edges) as possible as you can,
2. apply T2(Remove self-loops) as possible as you can, and
3. apply T3(Remove a node).

Then a state in $A$ (except initial and accepting states) is removed. Repeating this process, we have an NFA $A'$ consisting of two states:

Then the label $E$ of the unique edge gives us the regular expression.

Ex.:
1. 0 or more 'a's,
2. [0 or more 'b's] or [0 or more 'c's], and
3. 0 or more 'd's.