計算機工学 (西野)

4/16

第1章 序論

1.1 最初の言語

◎形式言語理論の2つの起源

(1) Norm Chomsky (アメリカの言語学者) による自然言語の構造に対する厳密な特徴づけ (1950年代)
→ 言語学 (文学部?) (拡大写像理論, GB理論…)
→ 変形文法理論.

学問に文系・理系の区別があるのは日本だけ.
↓
日本では隔りがあるが, 本来区別はない.

(2) プログラミング言語 ALGOL60 に対する形式的仕様記述の開発.
→ コンパイラ構成論.
手続き型
→ コンピュータサイエンス (理工学部?)

プログラミング言語論 (関数型, 論理型, オブジェクト指向型…)

◎ プログラミングパラダイム (思想)

手続き型 … プログラムは手続きだ!
関数型 … 〃 合成関数だ!
論理型 … 〃 定理証明だ!
オブジェクト指向型 … 〃 オブジェクト間のメッセージ伝達の定義だ!

例 1.1.1 (括弧言語)

正しく対応した左括弧と右括弧の任意の列から成る集合.

ex.) {(( )) ok,
( )( ) ok} → LISP (インタプリタ言語)
(( )(( )); NG

◎ 言語とは ある条件を満たす記号列の集合である

ex.) 英語とは. [英語を母語とする人が文法的と認める] という条件を満たす a, b, c, …, z からなる記号列の集合である.

括弧言語の帰納的定義

(1) 記号列 ( ) は対応した括弧である.
(2) 記号列 A が対応した括弧ならば (A) も対応した括弧
(3) 記号列 AB 〃 AB 〃

([( )]) は (A) と書けるので.
(1)' A (2)' 対応した括弧

([( )( )]) (3)
(2) A (1) (1) B (2)

↓

④ 指示に対する形式文法 = 「プロダクション」の集合

↪ (1) $S \hookrightarrow ( )$

↪ (2) $S \hookrightarrow (S)$ ＞ 「プロダクション」と呼ぶ

↪ (3) $S \hookrightarrow SS$

（書き換え規則の集合）

$S \underset{(1)}{\Rightarrow} SS \underset{(2)}{\Rightarrow} (S)S \underset{(1)}{\Rightarrow} \boxed{(( ))( )}$

4/23 続

(1) ⟶ 『Sを( )で置き換えて新しい記号列を作っても良い』という意味

$S \underset{(3)}{\Rightarrow} SS \underset{(2)}{\Rightarrow} (S)S \underset{(1)}{\Rightarrow} (( ))S$

$\underset{(2)}{\Rightarrow} (( ))(S) \underset{(3)}{\Rightarrow} (( ))(SS)$

$\underset{(1)}{\Rightarrow} (( ))(( )S) \underset{(1)}{\Rightarrow} (( ))(( )( ))$

導出過程 //

↳ $S \overset{*}{\Rightarrow} (( )(( )( ))$ と表記する

『Sはいくつかのステップののちに (( )(( )( )) を導出する』

- vとwを任意の記号列とするとき、記法 $v \Rightarrow w$ を『vはwを直接導出する』と読む。

④ 文法には2種類の記号が現れる

1) 導出には表れるが、最終的な記号列には現れない記号 ex.) S

2) 生成しようとしている記号列のアルファベットを構成する記号 //

ex.)「(」と「)」 最終的な記号列.

1) ⇒ 非終端記号 … 書き換えられる記号

2) ⇒ 終端記号 … 書き換えられない記号.

英語だったらアルファベットや.,()など
プログラミング言語だったらキーボードから打てる文字

④ プログラミング言語の文法記法において用いられる記法

BNF (Backus(バーカス) - Naur(ナウアー) - Form)

- 非終端記号 < > で囲む.
- → は ::= で表記する.

グーグルが欲しい日本人の人材
= 形態素解析, 自然言語処理が出来る技術者.

(P4) 1.2 文法と言語

X：記号の空でない有限集合
アルファベット と呼ぶ.

(ex) (1) 英語の場合.

$X = \{a, b, c, \cdots, x, y, z\}$

(2) 二進法表現の場合

$X = \{0, 1\}$

今後 何度も出てくる.

X^* X上の有限長の記号列全体 から成る集合（無限集合）

ex.) $X^* = \{0, 1\}$ の場合

$X^* = \{\lambda, 0, 1, 00, 01, 10, 11, 000, 001, \cdots\cdots\}$

λ : 空列 … 長さ0の記号列.

$\{\underbrace{\lambda}_{\text{長さ0}}, \underbrace{0, 1}_{\text{長さ1}}, \underbrace{00, 01, 10, 11}_{\text{長さ2}}, \underbrace{000, 001}_{\text{長さ3}}, \cdots\}$

長さ2 $2^2 = 4$ 通り　　長さ3 $2^3 = 8$ 通り

X上の言語L とは X^*上の任意の部分集合 のことをいう

ex.) $X = \{(,)\}$ のとき

括弧言語M は $M \subseteq \{(,)\}^*$
であるから 確かに言語である

$M = \{(), (X), ((X)), \cdots\}$

$X \cdots (,), ((, (()), \cdots$

→ not 括弧言語.

X と V 両方に入ることはない.

言語Lの各記号列 w は.
有限の長さ $|w|$ をもつ
空列の長さ $|\lambda| = 0$ である

$w = x_1, x_2, x_3, \cdots, x_n$ のとき.（ただし 各 j に対し $x_j \in X$ のとき）$|w| = n$ である

→ つまり. 長さ $|w|$ とは. w に含まれる記号の個数（文字数）である.

$X^+ = X^* - \{\lambda\}$ ← X上の長さ1以上の記号列の集合.

ex.) $X = \{0, 1\}$ のとき

$X^* = \{\lambda, 0, 1, 00, 01, 10, 11, 000, \cdots\}$

$X^+ = \{0, 1, 00, 01, 10, 11, 000, \cdots\}$

$w = x_1 x_2 \cdots x_n$, $w' = x_1' x_2' \cdots x_m'$ のとき
w と w' を連接すると 記号列

$ww' = x_1 x_2 \cdots x_n x_1' x_2' \cdots x_m'$

が得られる（$|w| = n$, $|w'| = m$ である）

連接 … 記号列を並べて結合する演算

ex) w = news , w' = paper のとき.
ww' = newspaper となる

定義 1.2.1

句構造文法 G とは 4項組（こうぐみ） (X, V, S, P) のことをいう. ただし X と V は 交わりを 持たない アルファベット であり

(1) X は文法の 終端アルファベット であり.
(2) V は文法の 非終端 〃 であり.

続

(3) Sは 文法の開始記号 であり さらに.

(4) Pは 文法の プロダクション の集合である.

→ (u, w) ∈ P のとき u → w と記す.
(u: 長さ1以上, w: 長さ0以上)

Pは 組(u, w)の集合である ただしuは.
少なくとも1つの非終端記号を含む (X ∪ V)上の
記号列であり. wは (X ∪ V)* の任意の元である

X = {(,)}, V = {S}, Sは開始記号
P = {(S, ( )), (S, (S)), (S, SS)}
↳ S → (S) と記していた (旧形式)

ex.) (括弧言語の場合)

略式
P = { S → ( ), S → (S), S → SS }

P = { (S, ( )), (S, (S)), (S, SS) }
u w
↖ 正式

X = {(,)}, V = {S}
Sは 開始記号であった.

5/7
(PS)

・定義 1.2.2

Gを文法とし yとzを X ∪ V に属する
有限長の記号列とする.

yの中に現われているあるプロダクションの1つ
の左辺を その右辺で 置き換えることにより
yからzが得られるとき y ⇒ z と書き
yはzを 直接導出する という.

例えば u → v が Gの プロダクション であり.
y = puq, z = pvq と書けるとき y ⇒ z
である

⑧ yとzであるか. xは. 記号列の並び.
$w_1, w_2, \cdots w_n$ が存在して.
$w_1 = y$ かつ $w_2 = z$ であり かつすべての
iに対して w_i が w_{i+1} を 直接導出
するとき. yはzを 導出する という.)

$w_i \Rightarrow w_{i+1}$

$w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_i \Rightarrow w_{i+1} \Rightarrow \cdots \Rightarrow w_n$
y = になるのは ⑧の場合.

すべての i に対し $w_i \Rightarrow w_{i+1}$ が成立
- $i = 1$ のとき $w_1 \Rightarrow w_2$
- $i = 2$ のとき $w_2 \Rightarrow w_3$
- ⋮
- $i = n-1$ のとき $w_{n-1} \Rightarrow w_n$

$y \overset{*}{\Rightarrow} z$ … プロダクション を 0回以上 有限回
適用して. yからzを導出することができる.

定義 1.2.3.

5/14

Gを開始記号Sをもつ文法とする.
Gによって生成される言語L(G)は

$L(G)=\{w \mid w\in X^* \text{ かつ } S \overset{*}{\Rightarrow} w\}$

と定義される

$L(G)=\{w \mid \underbrace{w\in X^*}_{\text{wは長さ0以上の終端記号列}} \text{ かつ } \underbrace{S \overset{*}{\Rightarrow} w}_{\substack{\text{wは開始記号Sから始めてプロダクションを}\\ \text{0回以上有限回適用することで導出できる.}}}\}$

例 1.2.4.

$V=\{S\}$, $X=\{a,b\}$
$P=\{S\to aSb,\ S\to ab\}$
S: 開始記号

$L(G)=\{a^n b^n \mid n>0\}$

↳ $a^n = \underbrace{aa\cdots a}_{n\text{個}}$

$S \Rightarrow ab$
$S \Rightarrow aSb \Rightarrow aabb$
$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaabbb$

(p7) 例 1.2.5

(*)

$S\to aSa \mid bSb \mid a \mid b \mid \lambda$ ← λ列 (長さ0の記号列)

- 通常の文法の記法.

(例 1.2.4)

$V=\{S\}$, $X=\{a,b\}$
$P=\{S\to aSb,\ S\to ab\}$

非終端は大文字, 終端は小文字 で表記.

文法の略記法.

単にプロダクションを並べて表記

$S\to aSb \mid ab$

↳ 正式に書くと.

$V=\{S\}$, $X=\{a,b\}$
$P=\{S\to aSa,\ S\to bSb,\ S\to a,\ S\to b,\ S\to \lambda\}$

④ この文法(*)が生成する言語は?

$S\Rightarrow \lambda$
$S\Rightarrow a$
$S\Rightarrow b$

$S\Rightarrow aSa \Rightarrow aa$ ($S\Rightarrow\lambda$に)
$S\Rightarrow aSa \Rightarrow aaa$ ($S\Rightarrow a$に)
$S\Rightarrow aSa \Rightarrow aba$ ($S\Rightarrow b$に)

$S\Rightarrow bSb \Rightarrow bb$
$S\Rightarrow bSb \Rightarrow bab$
$S\Rightarrow bSb \Rightarrow bbb$

$S\Rightarrow aSa \Rightarrow abSba \Rightarrow abaSaba \Rightarrow abaaSaaba \Rightarrow abaabaaba$

S(があったところ)を中心として, 左右対称.

⑤ アルファベット{a,b}上の回文を生成する文法.

(参考) 通常の回文.

日本語 : シンブンシ, ワケヤブヤケタ, 山本山

英語 : Madam, I'm Adam

例. 126.

$V=\{S,A,B\}$, $X=\{0,1\}$

$S \to 0B \mid 1A$

$A \to 0 \mid 0S \mid 1AA$

$B \to 1 \mid 1S \mid 0BB$

Ⓢ 開始記号 : S

Ⓝ 終端記号 : X

Ⓢ この文法が生成する言語は?

$S \Rightarrow 0B \Rightarrow 01$

$S \Rightarrow 1A \Rightarrow 10$

$S \Rightarrow 0B \Rightarrow 01S \Rightarrow 011A \Rightarrow 0110$

$S \Rightarrow 0B \Rightarrow 00BB \Rightarrow 001B \Rightarrow 0011$

$S \Rightarrow 1A \Rightarrow 11AA \Rightarrow 110SA$

$\Rightarrow 11００BA \Rightarrow 11001A \Rightarrow 11００10$

Ⓢ 0と1を同数含む2進列を導出する文法

→ このことを数学的帰納法で証明すると ....

↓

導出される終端記号列の長さに関する帰納法

証明すべきこと.

(1) 0, 1を同数含む記号列のみを導出.

(2) 0, 1を同数含む記号列は全てこの文法で導出できること

(1)を証明する代わりに, 次の3つの命題を同時に証明.

① Sからは 0, 1を同数含む記号列のみが導出可能であり.

② Aから導出可能な記号列は, 0を1つ余計に含み.

③ Bから導出可能な記号列は, 1を1つ余計に含む.

$S \Rightarrow 1A \overset{*}{\Rightarrow} 1$ [終端記号列]

Aが1個余計

5/21.

(p11)

1.3. 文脈自由言語 と 文脈依存言語

定義 1.3.1 (文脈自由文法)

文法Gが文脈自由である とは.

Pの各プロダクション $v \to w$

において, vが1つの非終端記号であるときをいう.

定義 1.3.2.

文法Gは各プロダクションが次のいずれかの形のとき 文脈依存 である

(1) $yAz \to ywz$

(ただし $A \in V$, $z \in (V \cup X)^*$, $w \in (V \cup X)^+$)

(2) $S \to \lambda$

(Sはプロダクションの右側に表れない)

$\gamma A \delta \rightarrow \gamma \omega \delta$

Aの右側にγがあり、右側にδがある
という文脈において、Aをωと書き換えてよい。

※ ただし

- $A \in V$ （Aは非終端記号）
- $\delta \in (V \cup X)^*$
  （δは非終端記号または終端記号からなる
  長さ1以上の記号列。）

その他に …
$S \rightarrow \lambda$ も許されている。
（Sは開始記号、λは空列。）

チョムスキー階層　言語の複雑さの階層

文脈依存言語 ∪	X → 自然言語
文脈自由言語 ∪	X → プログラミング言語
有限状態言語（正則言語）	↔（有限オートマトン）
	X → 鳥の歌（さえずり）

（複雑 ↑）

② 導出過程（0011の導出過程）

$S \Rightarrow 0B \Rightarrow 00BB \Rightarrow 001B \Rightarrow \boxed{0011}$

- 入力記号列 0011 から左の導出木を
  生成する処理を構文解析という

p13.

ある記号列が文脈自由文法によって
正しく導出できるならば その導出を
次の性質をもつ木Tによって記述
することができる。

(1) 根は開始記号Sでラベル付け
されている。

(2) 葉ではない各ノードは非終端記号
でラベル付けされている

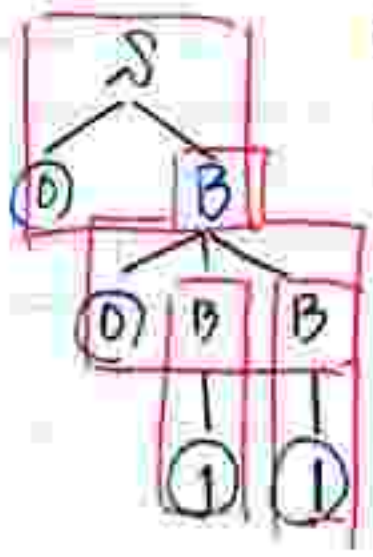
(3) 葉である各ノードは終端記号
（又は λ）でラベル付けされている。

(4) ノードNがAでラベル付けされており
Nが k 個の子ども $N_1, N_2, \cdots, N_k$
をもち、それらがそれぞれの記号
$A_1, A_2, \cdots, A_k$ でラベル付けされている
とき 文法には

$\boxed{A \rightarrow A_1, A_2, \cdots, A_k}$ となる形
のプロダクションが存在する

続→

③ 文脈自由型の導出木

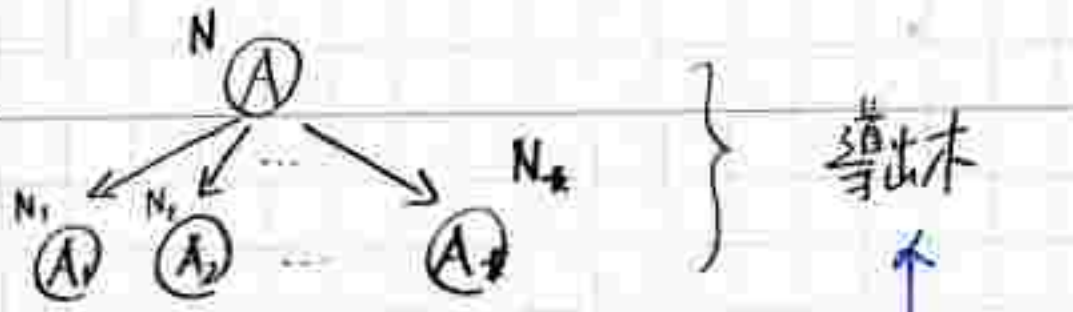


ex. 同数の0と1を含む
2進列を生成する文法。

$S \rightarrow 0B$
$S \rightarrow 0BB$
$B \rightarrow 1$

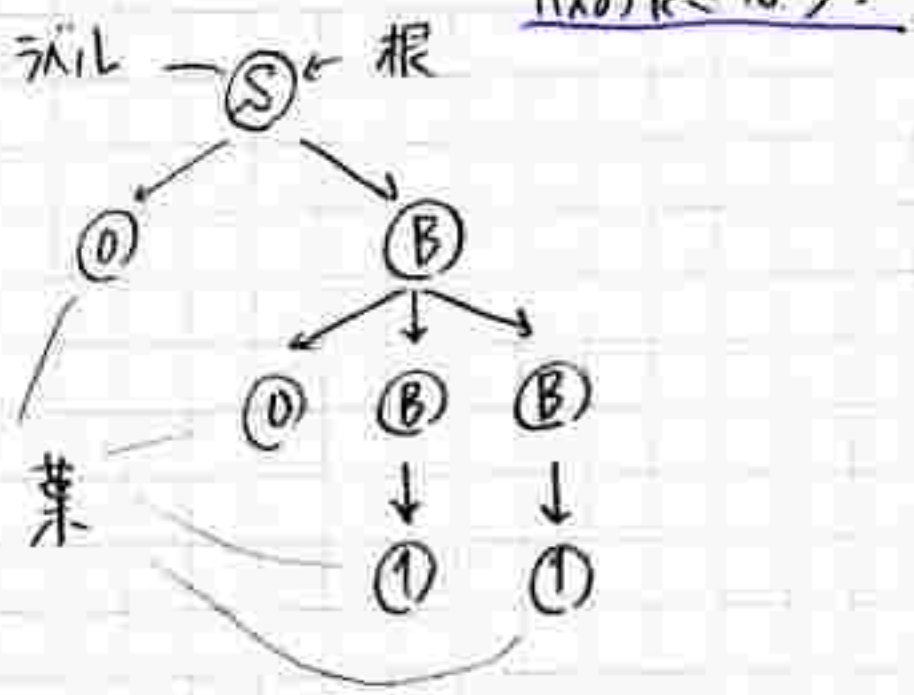
のようなプロダクションを
含んでいる。

参考図。



$A \to A_1, A_2, \cdots, A_k$

記号列 0011 に対する導出木



枝の長さは3.

(5) ある導出木により、導びかれた形式は、その導出に付随した木の葉を左から右へ読むことにより得られる。

- 上の導出木が表現する2つの導出

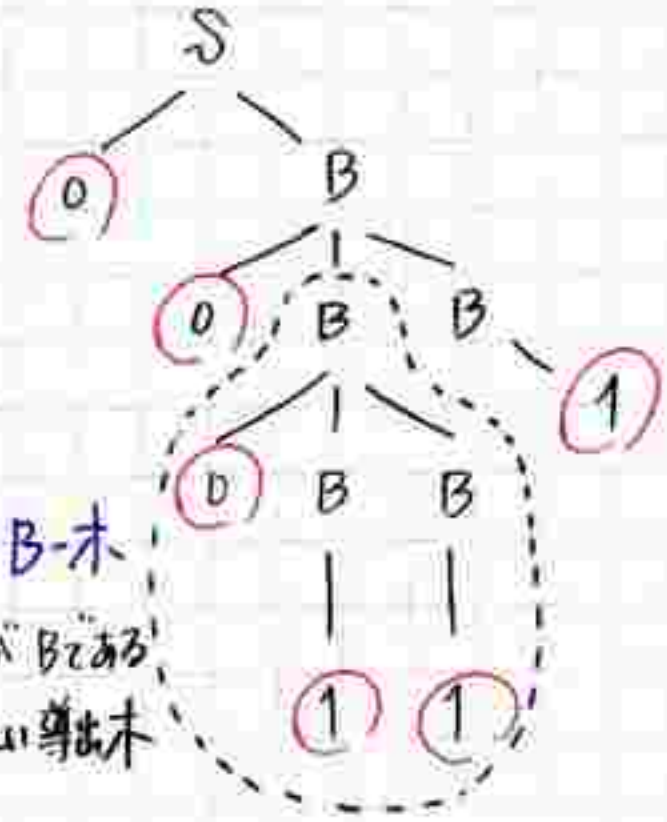
$S \to 0B \to 00BB \to 00\boxed{1}B \to 001\boxed{1}$ （最左導出）

$S \to 0B \to 00BB \to 00B\boxed{1} \to 00\boxed{1}1$ （最右導出）

(p14)

対称自由型の導出木が与えられたとき、根から葉へのパスの長さとそのパス上の非終端の個数を定義する木の高さとは、その木の最長パスの長さである

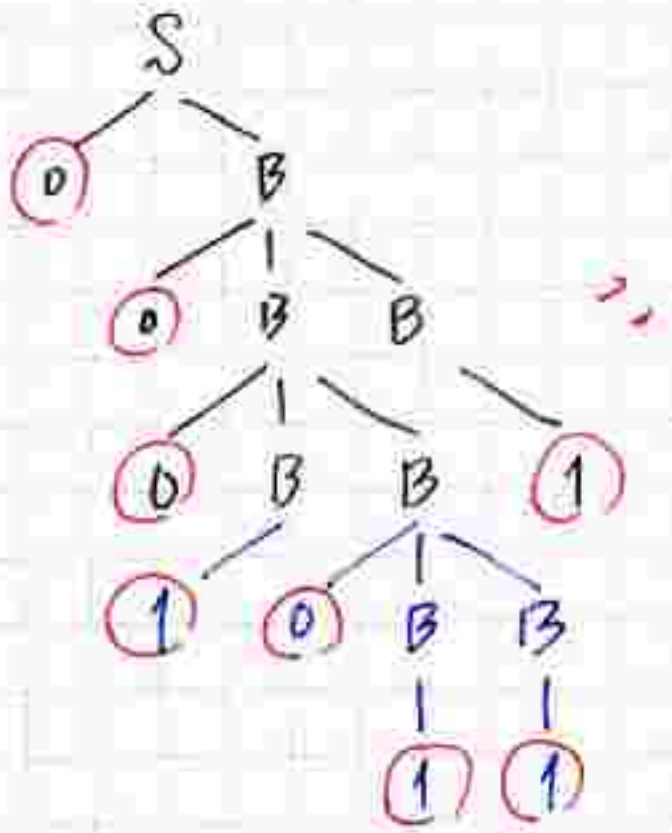
(p13) 図1.1 (b)



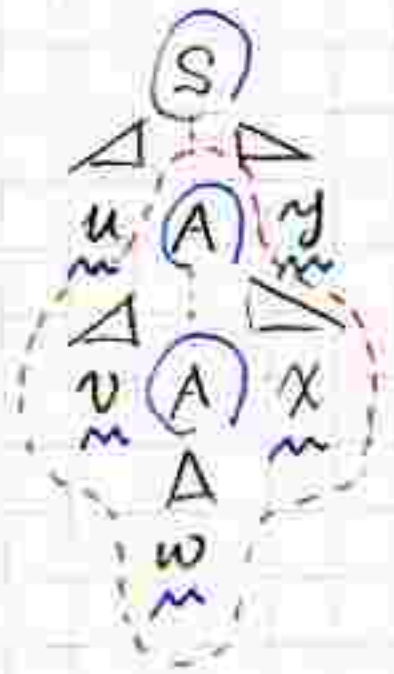
正当なB-木

根のラベルがBである、文法的に正しい導出木

00010111 に対する導出木



(p15) 図1.2



記号列 uvwxy に対する導出木

同一パス上に、非終端記号Aが2回現れる導出木

6/4

(p17) 例1.36

言語 $\{a^n b^n c^n \mid n>0\}$

は 文脈自由ではない

"

文脈自由文法では生成できない.

(p16) 補題1.35 (ポンプの補題)

L を文脈自由言語とする. このとき L のみに依存する定数 p と q が存在して, z が L に属し かつ $|z|>p$ ならば, z は以下の条件を満たすように $uvwxy$ と書くことができる.

z

(1) $|vwx| \leq q$

(2) v と x のうち高々1つが空であり.

(3) すべての $i \geq 0$ に対し.
$uv^i wx^i y$ は L に属する.

文脈自由言語 L
$=$ z
$\{ \cdots, uvwxy, \cdots$
$\cdots, uv^2wx^2y, \cdots$
$\cdots, uv^3wx^3y, \cdots$
$\cdots uv^i wx^i y, \cdots \}$

v, x が膨んでいく様子から.
"ポンプ"の補題と呼ばれる.

(p17) 例1.36の証明

- 背理法で証明する為に, Ⓛ 言語 $\{a^n b^n c^n \mid n>0\}$ が, 文脈自由であるとみると, 左の条件(ポンプの補題)を満たす p, q が存在する.

$r>p$ かつ $r>q$ とし 列 $a^r b^r c^r = z$ とおく. この列には, ポンプの補題が適用でき, $z = uvwxy$ と書ける

$z =$ | a^r | b^r | c^r |

$z =$ | u | v | w | x | y |

$|vwx| \leq q$ より 部分列 vwx は
(1) aのブロック, bのブロック, またはcのブロックのどれか1つの内側に完全に含まれるか.
(2) 1つの境界をまたぐことができるだけである

この(1),(2)の2つのどちらの場合についても引き延ばした列 $uvvwxxy$ において a, b または c のうち少なくとも1つの記号が増加できない $uvvwxxy \notin L$

(1) 部分列 vwx が a, b, または c のブロックのどれか1つの内側に完全に含まれる場合.

$z =$ | a^r | b^r | c^r |

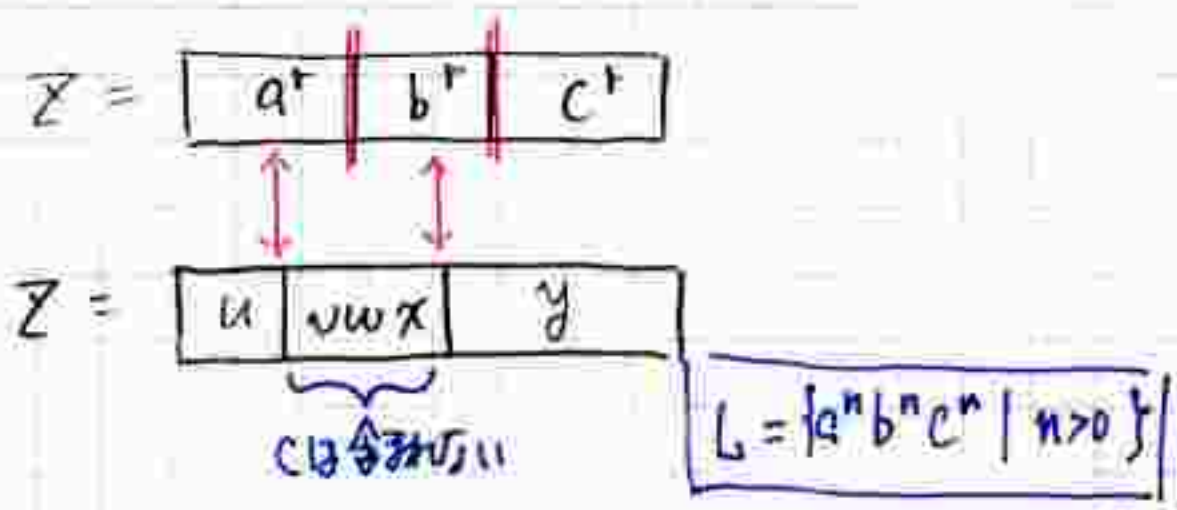
z | u | v | w | x | y |

$b \cdots b$

$L = \{a^n b^n c^n \mid n>0\}$

例えば uvx が b のブロックに含まれる場合には
uv^2wx^2y においては b の個数しか増加しない

(2) 部分列 vwx が1つの境界（a と b の境界
が または b と c の境界をまたぐ場合）
をまたぐ場合。



例えば、上の状況ならば uv^2wx^2y においては
c の個数だけ増加できない。

(p18) 例 1.3.7

言語 $\{a^k \mid k$ は完全平方数 $\}$

は文脈自由ではない。

6/11

(問1). 次の文脈自由文法を考える。

導出は開始記号から。

$S \to aAB$
$A \to Bba$
$B \to bB$, $B \to c$

このとき 終端記号列

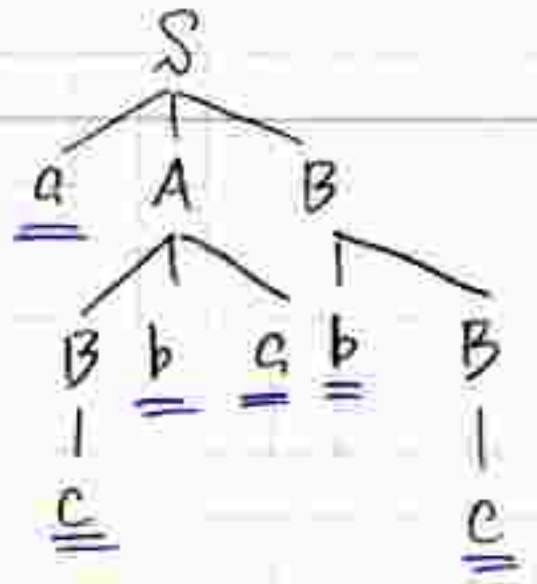
$acbabc$

に対する導出木を書け

$S \Rightarrow aAB \Rightarrow aBbaB$
$\Rightarrow acbabB \Rightarrow acbabc$ //

} 最左導出

↓



導出木

→ つまり これは 構文解析 をしている。

(問2).

開始記号 →

① $S \to aB$	② $B \to b$
③ $B \to bA$	④ $A \to aB$

このとき、文法 G によって生成される
言語を求めよ。

$S \Rightarrow aB \to abA \Rightarrow abaB$
$\Rightarrow ababA \Rightarrow \cdots \Rightarrow ab\cdots baba$

A. ~~a と b が交互に来る文法~~ //
ab が連続する文法

① 小さな具体例で試す。

$S \Rightarrow aB$ ② $\Rightarrow ab$
$S \Rightarrow aB$ ③ $\Rightarrow abA$ ④ $\to abaB$ ② $\Rightarrow abab$

① ⇓ 導出の開始のときのみ

③ ④ ↓ 記号の列の長さを伸ばす。

② ⇓ 導出の最後でのみ使う

$\therefore L(G) = \{(ab)^n \mid n \geq 1\}$ //

(問3) アルファベット{a,b}上の次の2つの言語を考える.

$L = \{a^r b^s \mid r,s > 0\}$

$L' = \{a^n b^n \mid n > 0\}$

このとき, L と L' を生成する文法を書け.

$L = \{ab$
aab, abb
$aabb, aaabbb, \cdots\}$

$L' = \{ab, aabb, aaabbb, \cdots\}$

L' に対する文法.

$S \to ab, \quad S \Rightarrow aSb$
$\Rightarrow aaSbb$

L に対する文法.

$A \Rightarrow aA, \quad A \to \lambda$

$A \Rightarrow aA \Rightarrow aaA \Rightarrow aa$ ← $a^n (n>0)$ に等しい文法

$S \Rightarrow aA, \quad A \to aA, \quad A \to b$
$A \to bB, \quad B \to bB,$
$B \to b$

6/18.

(ps) 例 1.2.7

言語 $\{a^n b^n c^n \mid n>0\}$

を生成する文脈保存文法.

$$S \xrightarrow{①} aSBC \mid \underset{②}{aBC}.$$

③ $CB \to BC$
④ $aB \to ab$
⑤ $bB \to bb$
⑥ $bC \to bc$
⑦ $cC \to cc$

ex. $a^3b^3c^3$ の導出. ($n=3$の場合)

$S \underset{①}{\Rightarrow} aSBC$
$\underset{①}{\Rightarrow} aaSBCBC$
$\underset{②}{\Rightarrow} aaaBCBCBC$
$\underset{③}{\Rightarrow} aaaBBCCBC$
$\underset{③}{\Rightarrow} aaaBBBCCC$
$\underset{④}{\Rightarrow} aaabBBCCC$
$\underset{⑤}{\Rightarrow} aaabbBCCC$
$\underset{⑥}{\Rightarrow} aaabbbCCC$
$\underset{⑦}{\Rightarrow} aaabbbcCC$
$\underset{⑦}{\Rightarrow} aaabbbccC$
$\underset{⑦}{\Rightarrow} aaabbbccc$

3番目のプロダクションを $\frac{n(n-1)}{2}$ 回使うと.

$a^n(BC)^n \overset{*}{\Rightarrow} a^nB^nC^n$

となる → 何故? (試験出る!)

文脈依存言語
$\times \{a^n b^n c^n \mid n>0\}$
文脈自由言語

(p20)

1.4 プログラム言語 構文解析

ex. (if-then-else 構文を生成する簡単な文法)

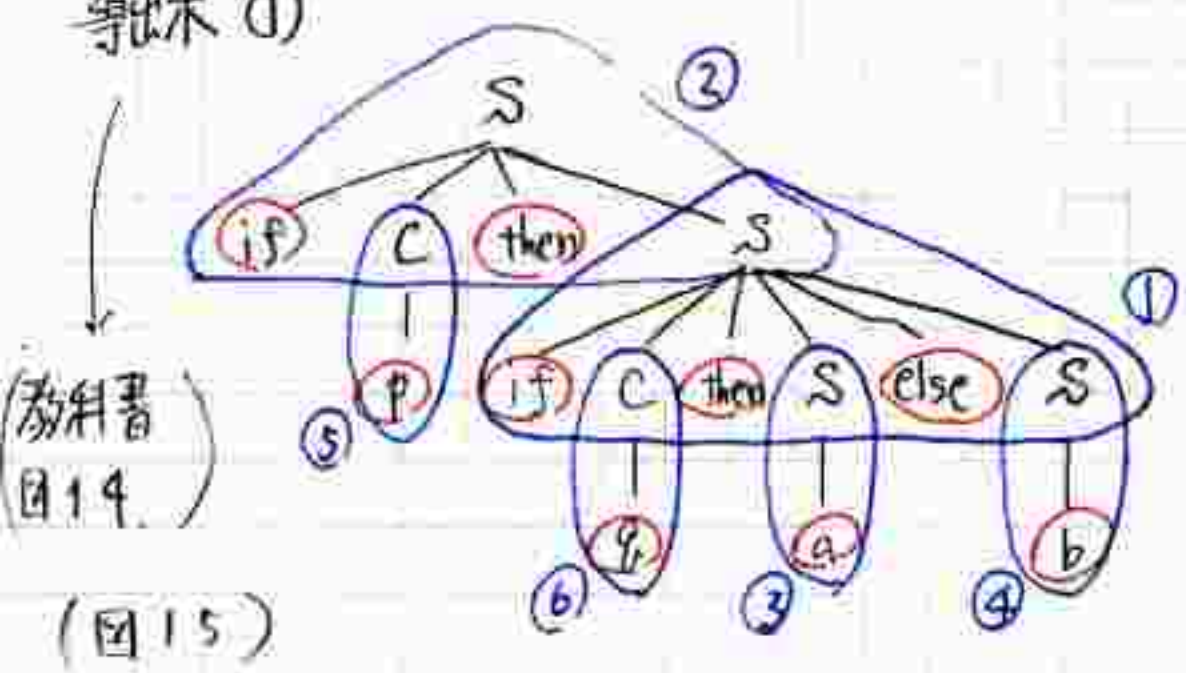
$S \xrightarrow{①}$ if C then S else S |
② if C then S |③a |④b

$C \xrightarrow{⑤} p \mid^{⑥} q$

[非終端記号. … S(文), C(条件文)
終端記号 … a,b(文), p,q(条件)
if, then, else (予約語)]

if p then if q then a else b

導出木①

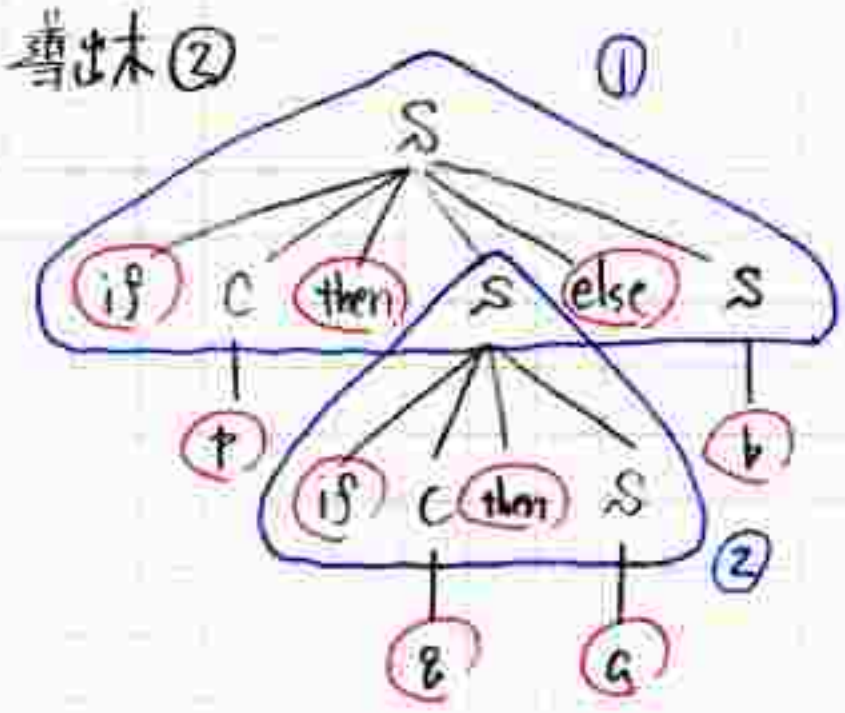


(参考書 図1.4.)

(図1.5)

↑

導出木②



【解釈】

① if p then (if q then a else b)

② if p then (if q then a) else b

Pascal 言語の場合 else は最も近い then に係る → 導出木①が正しい文法.

定義 1.4.2

文脈自由文法 G が 多義的 であるとは.
L(G) に属する ある列 x が 2つの異なる導出木を持つときをいう.

④ プログラミング言語の構文を定義する文法は. 多義的であってはならない.

6/25.

(p23). 文脈自由言語に対する解析問題

⑤ 2つの基本戦略

(1). 上昇型 … 終端記号列 x に対する導出木を下から上に向かって構成していく (葉)(根)
(ボトムアップ)

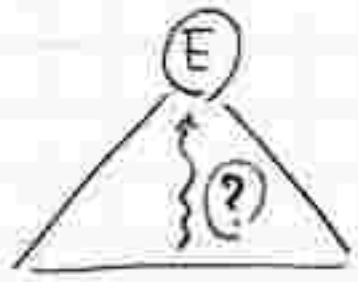
(2) 下降型 … 〃 上から下 〃 (根)(葉)
(トップダウン)

(p23). 例 143

(1). E → E+T
(2). E → T
(3) T → T*F
(4) T → F
(5) F → (E)
(6) F → 2

(開始記号：E)

※ 上昇型解析の例



2+2*2

⇐ F + 2 * 2 (6)を反転
⇐ T + 2 * 2 (4)を反転
⇐ E + 2 * 2 (2)を反転
⇐ E + F * 2 (6)を反転
⇐ E + T * 2 (4)を反転

どの
この次のプロダクションを反転するか？

3つの選択肢.

(1) E+T ⇒ E
(2) T ⇒ E
(3) 2 ⇒ F

ダメだったら戻ってくる
〃
バックトラック

⇐ E + T * F (6)を反転
⇐ E + T (3)を反転
⇐ E (1)を反転
//

例 144 (下降型解析)

トップダウン解析 (根→葉)

C → begin S_1 end (複合文)
S_1 → SS_2
S_2 → ; S_1 | λ
S → A | W | C
A → U := T (代入文)
T → pred(U) | succ(U) | 0
W → while U ≠ U do S (while文)
U → x | y

(| は or , λ は空列)

pred(x) = x - 1
↑
前者関数 (ただし. x ≥ 1)

succ(x) = x + 1
↑
後者関数

```
begin   y := 0 ;
        while  x ≠ y  do
               y := succ(y)
end
```

↳ 解析すべき終端記号列. (y := x と同値)

実際の入力は. 改行コードを含まない

begin y=0; while x≠y do y:=succ(y) end

開始記号

C ⇒ begin S_1 end

⇒ begin $S S_2$ end

⇒ begin $A S_2$ end

⇒ begin $V := T S_2$ end

⇒ begin $y := T S_2$ end

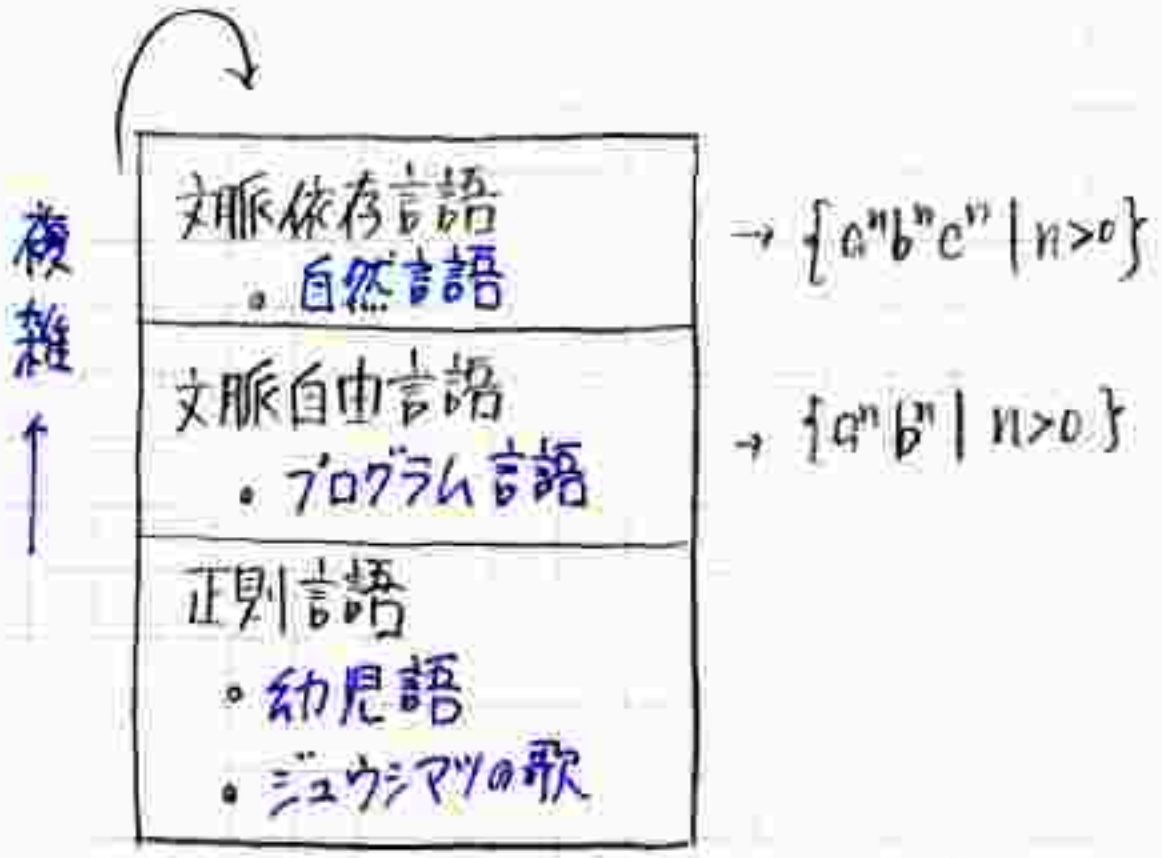
⇒ begin $y := 0 ; S_1$ end

⇒ begin $y := 0 ; S S_2$ end

⇒ begin $y := 0 ; W S_2$ end

LL解析の例

バックトラックせずに
構文解析を行う



7/2

(p29) 1.5. 文脈自由文法と自然言語

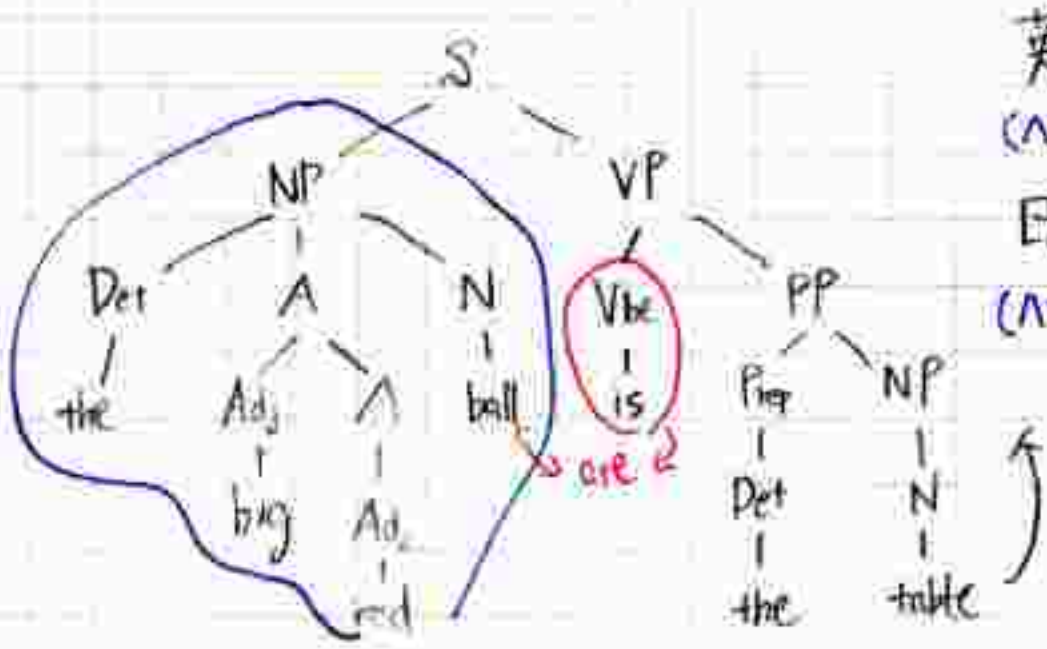
文法規則

S → NP. VP.
NP → Det N | Det AN
A → Adj A | Adj
VP → Vbe PP
PP → Prep NP

S : 文
NP : 名詞句
VP : 動詞句
Adj : 形容詞
Det : 限定詞
V : 動詞
N : 名詞
Prep : 前置詞
A : 形容詞句
PP : 前置詞句

辞書に相当

Vbe → is / are / was / …
Det → the / an / a …
Prep → on / in / for …
N → ball / table …



・情報検索に対する自然言語処理技術の応用

(1) 形態素解析

(2) 構文解析

@ Chomsky の普通文法理論

脳内

普通文法 = 人間言語の素 → 母語の文法

誕生時 → 後天的学習 パラメータ設定

ex) パラメータの例

英語 (ヘッド前置) the man in the park

日本語 (ヘッド後置) 公園にいる男

主要語 (ヘッド)

7/9

計算機工学 9

文脈依存言語 $\{a^n b^n c^n \mid n>0\}$	(言語の複雑さ) ✓自然言語.	(アプリケーション)	(機械モデル) 線形有界オートマトン → チューリング機械
文脈自由言語 $\{a^n b^n \mid n>0\}$	✓プログラミング言語.	✓構文解析. (コンパイラ)	プッシュダウンオートマトン ≈ コンピュータ.
有限状態言語	幼児語. 鳥の歌(ジュウシマツ)	字句解析. 情報検索.	有限状態受理機械 (有限オートマトン)

これからやる

(文法モデル)

✓文脈依存文法.

✓文脈自由文法.

✓右線型文法. (正則文法)

これからやる

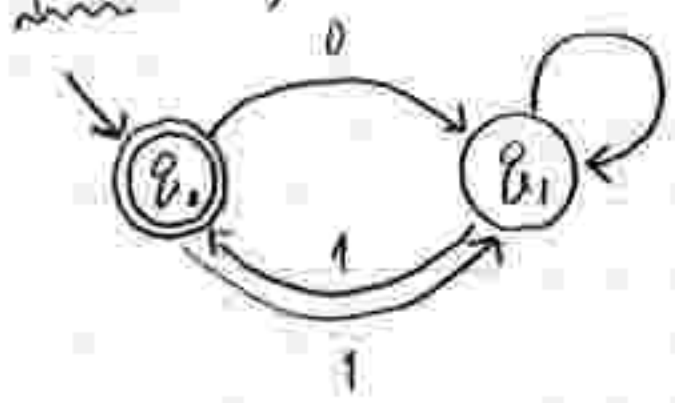
「応用オートマトン工学」

◎有限オートマトンに基づくアプリケーション開発を目指す.

自販機、券売機… 世の中の様々な機械を指す

---

p54 H.)



有限状態受理機

q_0, q_1 : 状態.

0, 1 : 入力信号.

初期状態 (振り出し)　受理状態 (あがり)

ex.) 入力記号列

001 : 受理

0011 : 拒否

$X = \{0, 1\}$

$Q = \{q_0, q_1\}$

$F = \{q_0\}$

$\delta(q_0, 0) = q_1$

$\delta(q_0, 1) = q_1$

$\delta(q_1, 0) = q_1$

$\delta(q_1, 1) = q_0$

定義. 2.3.7 (決定性有限オートマトン)

入力アルファベットが X の有限状態受理機 (FSA) M とは以下の条件を満たす4項組 (Q, δ, q_0, F) のことである.

ただし Q は有限の状態集合

q_0 は初期状態, $F \subseteq Q$ は受理状態の集合, δ は以下の形の状態遷移関数

$$\delta(p, a) = q$$

は M が状態 p のときに記号 a を読んだら, 状態 q に遷移することを表す

(p37) 定義 2.1.5

$G=(X, V, S, P)$ を文法とする P の
プロダクションの形に関する制限に従って
G は タイプ i $(i=0,1,2,3)$ と分類される

ex. タイプ2 = 文脈自由 , タイプ1 = 文脈依存

(1) 文法が タイプ3 であるとは、すべてのプロダクションが

$A \to aC$ または $A \to b$

なる形のときをいう。ただし $A, C \in V$,
$a, b \in X$ または $b = \lambda$ とする。この文法は
右線形文法と呼ばれる。 → 空列

a, b: 0, 1 など

有限状態受理機

定義 2.3.9

入力アルファベット X の FSA $M=(Q, \delta, q_0, F)$
によって受理される言語 $T(M)$ は、 (言語 → 記号列の集合)

$$T(M) = \{ w \mid \delta^*(q_0, w) \in F \}$$

と定義される。

δ^*: δ 関数の有限回の適用

命題 2.3.10

すべての FSL は右線型言語である

注) X^* の部分集合 L は それが ある FSA
M に対して、$T(M)$ と等しくなるとき
有限状態言語 (FSL) であるという

(略証)

$$P = \{ q \to xq' \mid q' \in \delta(q, x) \}$$
$$\cup \{ q \to x \mid \delta(q, x) \in F \}$$

$\delta(q, x) \in F$ → $q \to x$

$\delta(q_0, \lambda) \in F \to q_0 \to \lambda$
(↳ q_0 自身に遷移)

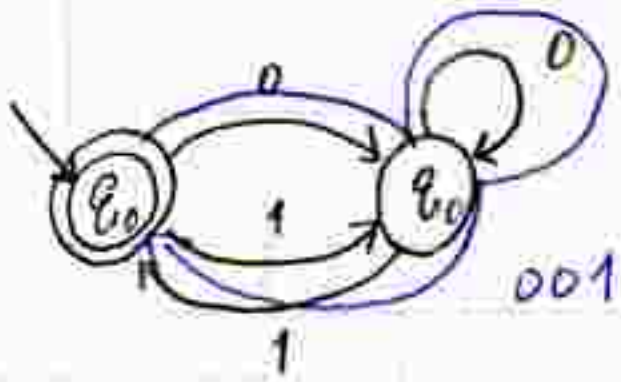
$\delta(q_1, 1) \in F \to q_0 \to 1$

をプロダクションとする右線型文法を
構成すればよい。

対応するプロダクション

$q_0 \to 0q_1$, $q_0 \to 1q_1$,
$q_1 \to 0q_1$, $q_1 \to 1q_0$,
$q_0 \to \lambda$, $q_1 \to 1$

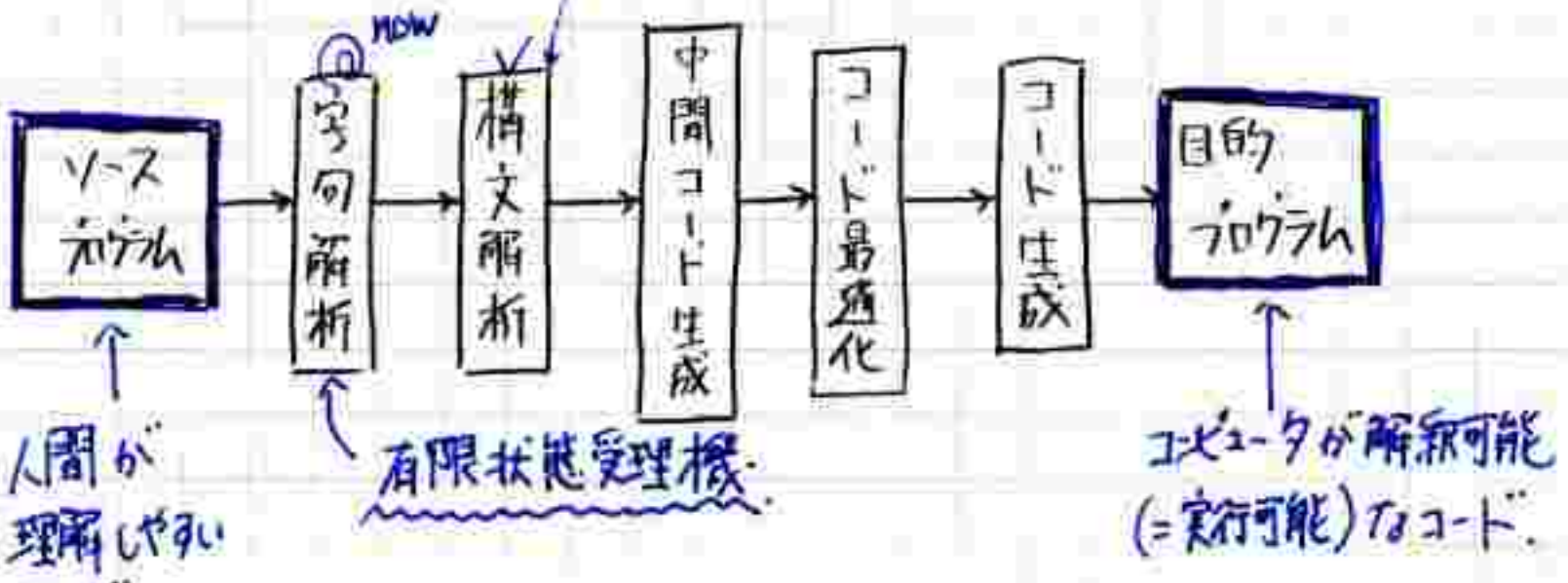
M



ex.) $001 \in T(M)$

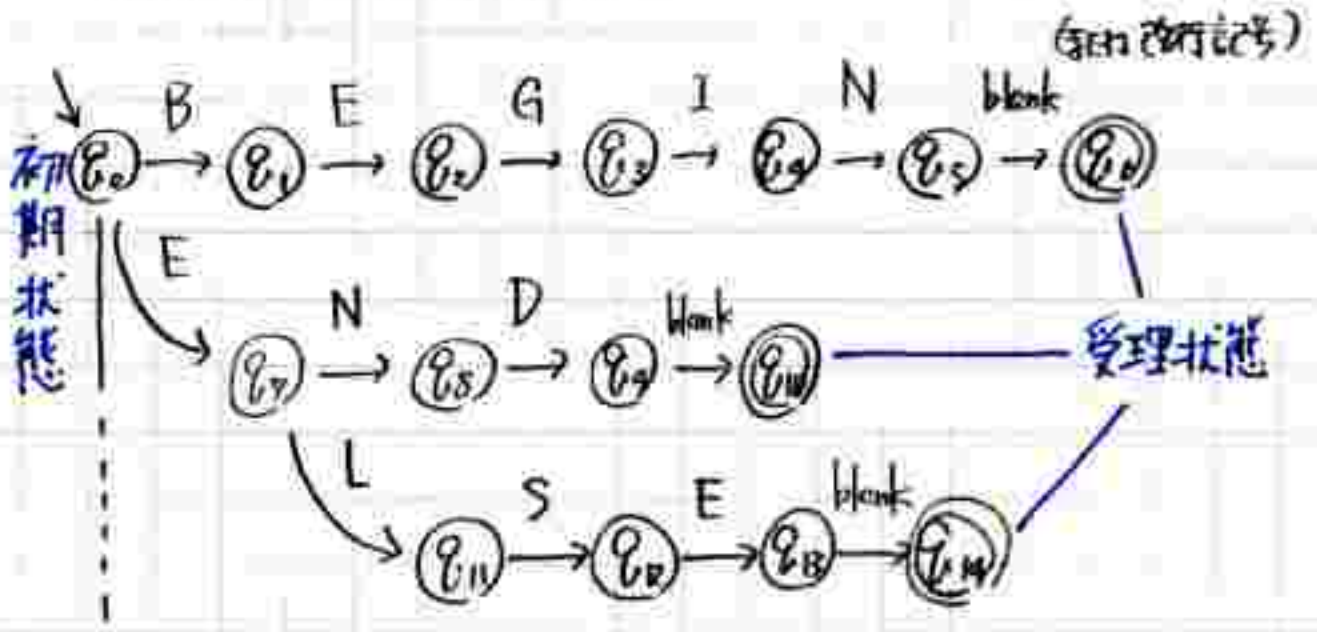
$q_0 \Rightarrow 0q_1 \Rightarrow 00q_1 \Rightarrow 001$ //

コンパイラの構成.



① 字句解析に用いられる有限状態受理機 (FSA).

① キーワードを受理するFSA



② 識別子 (identifier)

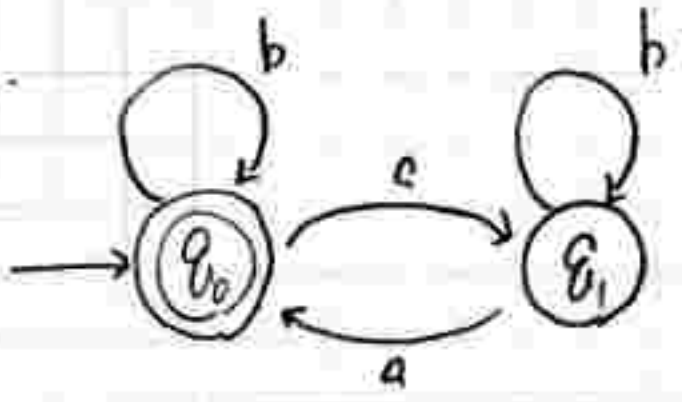
ex.) uec 2011

問1.) a と b を入力記号とする.
a を偶数個含む a と b の記号列のみ受理する FSA を構成せよ.

ex.) aabbbab 4コ
aa 2コ
bbb (aが0個)
ababaa 4コ
} 受理.

ababa 3
bbabb 1
等は受理しない

A.

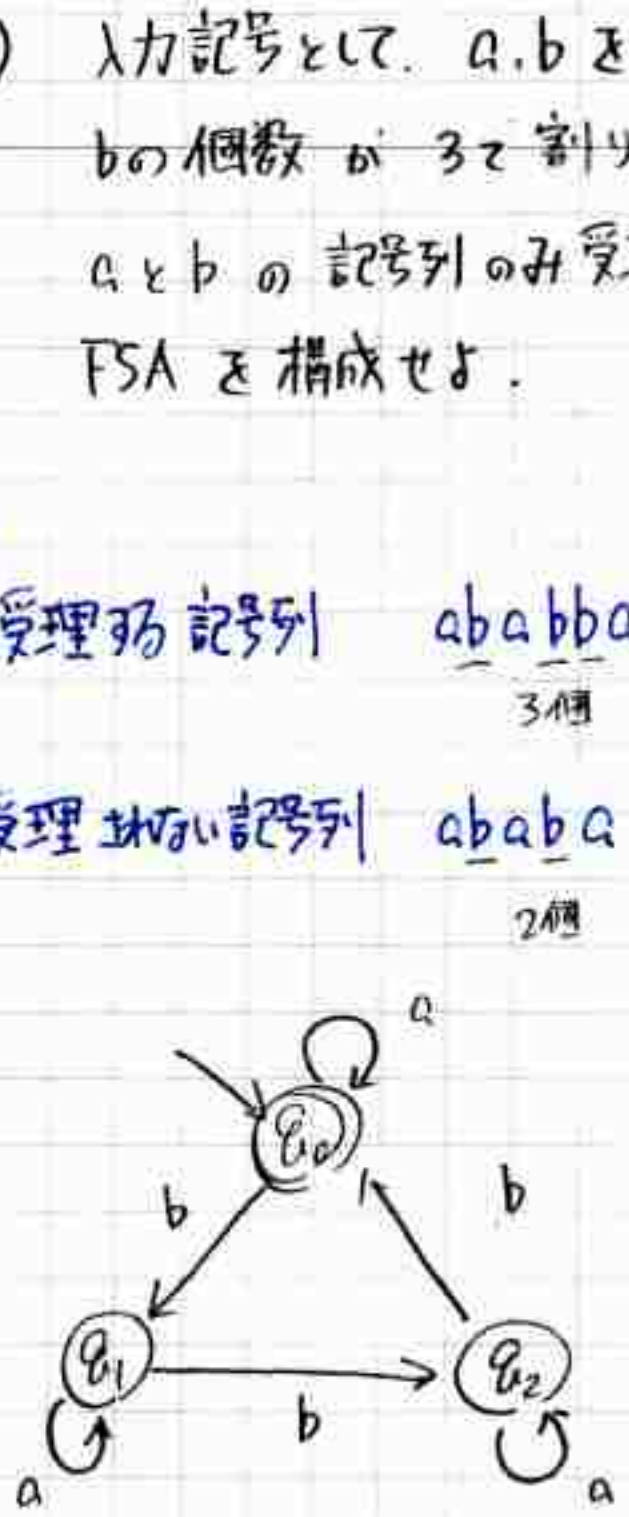


問2.) 入力記号として、a,bを持ち、
bの個数が3で割り切れる
aとbの記号列のみ受理する
FSAを構成せよ.

ex.)
受理する記号列　ababba
（bが3個）

受理しない記号列　ababa
（bが2個）

A.



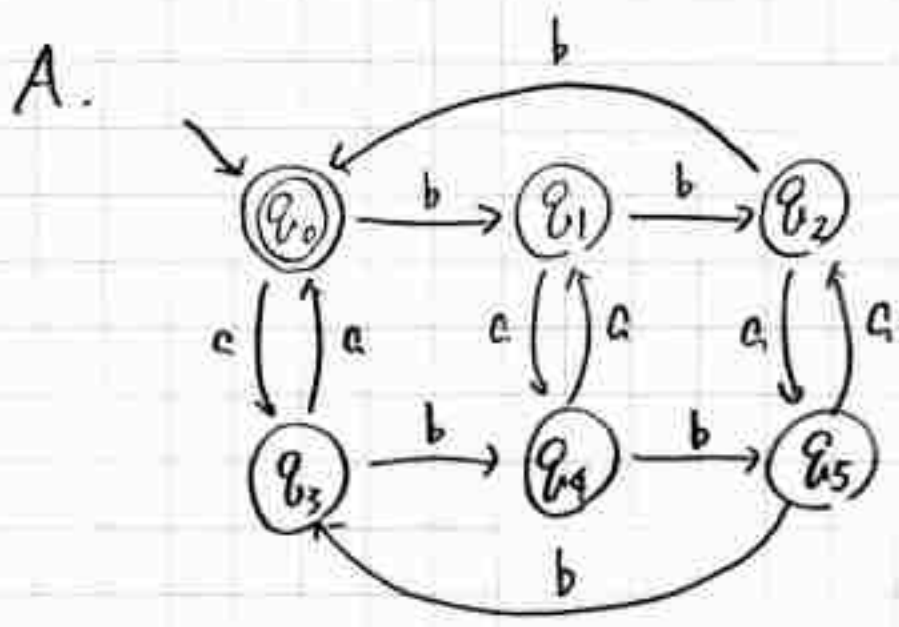
→ bの個数が0個でも3の倍数
として カウント?

問3) 入力記号として、aとbを持ち
aとbの記号列で aの個数
が偶数個で bの個数が3で
割り切れるもののみを受理する
FSAを構成せよ.

ex.)
受理する記号列.　ababbaa

受理しない記号列　ababaa

A.



a … 偶数個 読まれるのは、aが
必ず 上で 終わるとき

b … bは 上下に依らないが 必ず
3回 読まれなければいけない