計算論 A / 2015年度

第5章 文脈自由文法と言語 (1/2)

文脈自由文法の基本

角川　裕次

もくじ

第5章 文脈自由文法と言語 (1/2)
【教科書 p192〜】

＊＊＊ 本日の重要概念 ＊＊＊

文脈自由文法, 文脈自由言語,
導出, 構文木

φ(・ω・)メモメモ

第5章の概要

文脈自由言語

文脈自由言語は正則言語よりも広い言語クラス

- 文脈自由言語 : Context Free Language (CFL)

文脈自由文法を文脈自由言語の記述に用いる

- 文脈自由文法 : Context Free Grammar (CFG)

文脈自由言語では再帰的な構造を取り扱える

- 正則言語では再帰的構造は取り扱えない

文脈自由文法の重要な応用

コンパイラにおける構文解析 (Parsing)

文法記述から構文解析器の自動生成

- コンパイラコンパイラ (Compiler-Compiler)

XML (Extensible Markup Language) での文書型定義

第5章(1/2)の概要

5.1 文脈自由文法の定義

- 言語の定義方法を説明

5.2 構文木 (Parse tree)

- 文字列に対する文法上の構造を図示するもの
- コンパイラの構文解析器の出力

5.1 文脈自由文法

5.1.1 直観的な例：回文 (palindrome)

回文

前から読んでも後ろから読んでも同じ文字列

- 「たけやぶやけた」
  — (竹藪焼けた)
- 「いまうんどうかいすいかうどんうまい」
  — (今運動会スイカ饂飩旨い)
- 「のものものはのものもの」
  — (野茂の物は野茂の物)
- 「wasitacatisaw」
  — (Was it a cat I saw?)

文字列 w が回文 $\Leftrightarrow w = w^R$

回文言語 L_{pal}

アルファベット $\{0,1\}$ に限定した回文を考える

言語 L_{pal} ： 回文の集合で定義

- 属する語の例: 0110, 11011, ε
- 属さない語の例: 011, 0101

形式的定義：$L_{\text{pal}} = \{w \in \{0,1\}^* : w = w^R\}$

回文言語 L_{pal} は正規言語ではない (1/4)

回文 $\overbrace{00\ldots00}^{i}1\overbrace{00\ldots00}^{i}$

有限オートマトンでは左右の0の数の一致を検査不能

- 有限状態では無限通りの0の個数を区別できない

回文言語 L_{pal} は正規言語ではない (2/4)

正規言語の反復補題で証明

背理法: L_{pal} が正規言語であると仮定してみる

- 反復補題が成立するはず (p.140 定理4.1参照)

反復補題での定数を n とおく

回文言語 L_{pal} は正則言語ではない (3/4)

回文 $w = 0^n 1 0^n \in L_{\mathrm{pal}}$ を考える

- n： 反復補題での定数

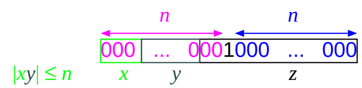
反復補題より以下の条件を満たす分解 $w = xyz$ が存在

1. $y \neq \varepsilon$
2. $|xy| \leq n$
3. 全ての $k \geq 0$ に対し $xy^k z \in L_{\mathrm{pal}}$

回文言語 L_{pal} は正則言語ではない (4/4)

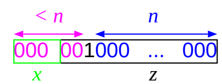
条件1($y \neq \varepsilon$)と条件2($|xy| \leq n$)から以下が成立

- y は左側の0を少なくとも一つ含む
- y に1と右の0は全く含まない



条件3に矛盾: $xy^0 z = xz \notin L_{\mathrm{pal}}$

- 理由: 1の前後の0の数が違う (回文ではない)



回文の自然で再帰的な定義

1. 基礎 : ε, 0, 1 は回文
2. 再帰 : もし w が回文なら $0w0$ と $1w1$ も回文
3. 回文は上記2つの規則で構成できるものに限る

※この言語の定義は再帰的構造を有する

文脈自由文法

再帰的定義を形式的に記述する記法のひとつ

- 文法はいくつかの規則(生成規則)で構成

回文を定義する文脈自由文法の例 (図5.1)

1. $P \to \varepsilon$
2. $P \to 0$
3. $P \to 1$
4. $P \to 0P0$
5. $P \to 1P1$

- Pは文法における変数(非終端記号)

5.1.2 文脈自由文法の定義

言語の文法的記述の4つの構成要素 V, T, P, S

文脈自由文法 G は 4つ組 $G = (V, T, P, S)$ で表現

T: 終端記号の集合

- 定義される言語の文字列を構成する記号の有限集合

V: 変数 (非終端記号)の集合

- 文字列の集合を表現する記号

S: 出発記号 (始記号)

- 定義する言語を表す変数

P: 規則 (生成規則)の有限集合

- 言語の再帰的定義を表現
- ※詳しくは次のスライド

規則 (生成規則) の形式

「頭部 → 本体」の形式

- 例: $X \rightarrow XY\mathbf{0}$

頭部

- その生成規則で(部分的に)定義される変数

本体

- 終端記号と変数からなる列 (長さ 0以上)
- 頭部が表す言語の中の記号列の構成方法を表現

生成規則の適用

変数を生成規則で置き換えてゆく

終端記号は置き換えない

最終的に終端記号だけの列へ

例 5.2: 回文文法 G_{pal}

$G_{\text{pal}} = (\{P\}, \{\mathbf{0}, \mathbf{1}\}, P, A)$

- 変数の集合: $\{P\}$
- 終端記号の集合: $\{\mathbf{0}, \mathbf{1}\}$
- 出発記号 P

生成規則の集合 A

- $P \rightarrow \varepsilon$
- $P \rightarrow \mathbf{0}$
- $P \rightarrow \mathbf{1}$
- $P \rightarrow \mathbf{0}P\mathbf{0}$
- $P \rightarrow \mathbf{1}P\mathbf{1}$

例 5.3: 式 (1/4)

演算子と識別子だけで構成

演算子: 加算 $+$ と乗算 $*$ のみ

- 括弧の使用を許す

識別子: 文字は $a, b, \mathbf{0}, \mathbf{1}$ のみに限定

- 最初の文字は a または b に限定
- その後に $\{a, b, \mathbf{0}, \mathbf{1}\}^*$ の任意の列を追加してよい
- 正則表現での表現: $(a + b)(a + b + \mathbf{0} + \mathbf{1})^*$

例 5.3: 式 (2/4)

変数 E: 式を表す

- 再帰的な定義を行なう (このあとで)

変数 I: 識別子を表す

例 5.3: 式 (3/4)

式文法 $G_{\text{exp}} = (\{E, I\}, T, P, E)$

- 変数は E と I
- 終端記号 $T = \{a, b, \mathbf{0}, \mathbf{1}, +, *, (,)\}$
- 生成規則の集合は P (次のページで)
- 出発記号は E

例 5.3: 式 (4/4)

生成規則の集合 P

1. $E \to I$
2. $E \to E + E$
3. $E \to E * E$
4. $E \to (E)$
5. $I \to a$
6. $I \to b$
7. $I \to Ia$
8. $I \to Ib$
9. $I \to I0$
10. $I \to I1$

規則1〜4: 式の構成法の再帰的な記述

規則5〜10: 識別子の構成法の記述

生成規則の簡潔な表現方法

頭部の変数が同じ生成規則の本体を1つにまとめる

縦棒で区切って列挙

例1. 式文法 G_{exp}

- ❖ $E \to I \mid E + E \mid E * E \mid (E)$
- ❖ $I \to Ia \mid Ib \mid I0 \mid I1$

例2. 回文文法 G_{pal}

- ❖ $P \to \varepsilon \mid 0 \mid 1 \mid 0P0 \mid 1P1$

5.1.3 文法による導出

言語に所属する文字列を得る方法: 2通り

再帰的推論

- ❖ 本体から頭部へと生成規則を使う

導出

- ❖ 頭部から本体へと生成規則を使う

再帰的推論: 本体から頭部へと生成規則を使う

本体が終端記号だけの列から出発

所属が既知の文字列を本体の変数に当てはめ
更に長い終端記号の列を既知のものにする

最終的には出発記号に到達させる

教科書の例5.4参照

導出: 頭部から本体へと生成規則を使う

出発記号から開始

生成規則で変数を置き換えて終端記号列を得る

教科書の例5.5参照

記法の習慣 (文脈自由文法の導出)

(教科書200ページ)

英小文字 (a, b など): 終端記号を表す

- 数字, $+$, 括弧なども終端記号

英大文字で最初の方 (A, B など): 変数を表す

英小文字で最後の方 (w, z など): 終端記号の列を表す

英大文字で最後の方 (X, Y など):
終端記号または変数を表す

ギリシャ小文字 (α, β など):
終端記号と変数の一方または両方の列を表す

導出: 記号 $\Rightarrow$

関係記号 $\underset{G}{\Rightarrow}$ を定義

生成規則1回適用の前後の記号列間の関係を表現

$\alpha A \beta \underset{G}{\Rightarrow} \alpha \gamma \beta$

- $G = (V, T, P, S)$: 文脈自由文法
- α と β: $(V \cup T)^*$ の中の列
- A: 変数 ($\in V$)
- $A \rightarrow \gamma$: 生成規則 ($\in P$)

省略記法 $\Rightarrow$: G が明らかなとき $\underset{G}{\Rightarrow}$ の代わりに使用

導出: 記号 $\overset{*}{\underset{G}{\Rightarrow}}$

生成規則複数回適用の前後の記号列間の関係を表現

- 複数回 = 0回以上のこと

再帰的定義

- 基礎: $\alpha \overset{*}{\underset{G}{\Rightarrow}} \alpha$
- 再帰: $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$ かつ $\beta \underset{G}{\Rightarrow} \gamma$ ならば $\alpha \overset{*}{\underset{G}{\Rightarrow}} \gamma$

省略記法 $\overset{*}{\Rightarrow}$: G が明らかなとき $\overset{*}{\underset{G}{\Rightarrow}}$ の代わりに使用

例 5.5: 導出の例 (文法は図5.2 のもの)

$$
\begin{aligned}
E & \\
&\Rightarrow E * E \\
&\Rightarrow I * E \\
&\Rightarrow a * E \\
&\Rightarrow a * (E) \\
&\Rightarrow a * (E + E) \\
&\Rightarrow a * (I + E) \\
&\Rightarrow a * (a + E) \\
&\Rightarrow a * (a + I) \\
&\Rightarrow a * (a + I0) \\
&\Rightarrow a * (a + I00) \\
&\Rightarrow a * (a + b00)
\end{aligned}
$$

例 5.5: 導出の例 (文法は図5.2 のもの), その2

$$
\begin{aligned}
E & \\
&\overset{*}{\Rightarrow} a * (a + b00)
\end{aligned}
$$

5.1.4 最左導出と最右導出

最左導出

常に最も左の変数に生成規則を適用する導出方法

関係記号: $\underset{左}{\Rightarrow}$ あるいは $\underset{左}{\overset{*}{\Rightarrow}}$

用いる文法を明示する時: $\underset{左\,G}{\Rightarrow}$ あるいは $\underset{左\,G}{\overset{*}{\Rightarrow}}$

例 5.6: 最左導出の例 (文法は図5.2 のもの), 1/2

$$
\begin{aligned}
&E\\
&\quad\underset{左}{\Rightarrow} E * E\\
&\quad\underset{左}{\Rightarrow} I * E\\
&\quad\underset{左}{\Rightarrow} a * E\\
&\quad\underset{左}{\Rightarrow} a * (E)\\
&\quad\underset{左}{\Rightarrow} a * (E + E)\\
&\quad\underset{左}{\Rightarrow} a * (I + E)\\
&\quad\underset{左}{\Rightarrow} a * (a + E)
\end{aligned}
$$

例 5.6: 最左導出の例 (文法は図5.2 のもの), 2/2

$$
\begin{aligned}
&\underset{左}{\Rightarrow} a * (a + I)\\
&\underset{左}{\Rightarrow} a * (a + I0)\\
&\underset{左}{\Rightarrow} a * (a + I00)\\
&\underset{左}{\Rightarrow} a * (a + b00)
\end{aligned}
$$

最右導出

常に最も右の変数に生成規則を適用する導出方法

関係記号: $\underset{右}{\Rightarrow}$ あるいは $\underset{右}{\overset{*}{\Rightarrow}}$

用いる文法を明示する時: $\underset{右\,G}{\Rightarrow}$ あるいは $\underset{右\,G}{\overset{*}{\Rightarrow}}$

5.1.5 ある文法の言語

記法 $L(G)$

$L(G)$: 文法 $G = (V, T, P, S)$ が生成する言語

- G の出発記号から導出できる終端記号列の集合

形式的定義: $L(G) = \{w \in T^* \mid S \underset{G}{\overset{*}{\Rightarrow}} w\}$

文脈自由言語

文法 G: 文脈自由文法

言語 $L(G)$: 文脈自由言語(CFL)と呼ぶ

- CFL: Context Free Language
- 文脈自由文法で生成される言語

$L = L(G)$ の証明方法 (1/2)

与えられるもの2つ

- ある言語 L
- ある文法 G

判定したいこと

- $L = L(G)$ か否か
- (言語 L は文法 G が生成する言語に一致するか否か)

$L = L(G)$ の証明方法 (2/2)

言語 L, 文法 G で生成される言語 $L(G)$ はともに集合

- いずれも一般には無限集合

$(L = L(G)) \equiv ((L \subseteq L(G)) \wedge (L \supseteq L(G)))$

- 一般に $L_1 \subseteq L_2$ の証明は
  $\forall w \in L_1 : w \in L_2$ を示せば良い

$L \subseteq L(G)$ の証明:
任意の $w \in L$ に対し $w \in L(G)$ を示せば良い

$L \supseteq L(G)$ の証明:
任意の $w \in L(G)$ に対し $w \in L$ を示せば良い

定理5.7: 回文文法 G_{pal} の言語 $L(G_{\text{pal}})$ は文脈自由言語

証明すべきこと:

- 任意の $w \in \{0,1\}^*$ に対し
  $w \in L(G_{\text{pal}}) \iff (w$ は回文である$)$

これを示すには以下の2つを示せばよい

- 十分性: $w \in L(G_{\text{pal}}) \iff$ → $w \in L(G_{\text{pal}}) \Leftarrow (w$ は回文である$)$
- 必要性: $w \in L(G_{\text{pal}}) \Rightarrow (w$ は回文である$)$

十分性の証明 (1/2)

示すこと: $w \in L(G_{\text{pal}}) \Leftarrow (w$ は回文である$)$

証明は回文 w の長さ $|w|$ に関する帰納法

基礎: $|w|$ が 0 または 1 の場合

- w は ε, 0, 1 のいずれかに限られる
- ε の導出: 生成規則 $P \to \varepsilon$
- 0 の導出: 生成規則 $P \to 0$
- 1 の導出: 生成規則 $P \to 1$

十分性の証明 (2/2)

帰納: $|w| \geq 2$ の場合

- 仮定より w は回文: w は $0x0$ または $1x1$ の形
- x もまた回文
- $|x| = |w| - 2 \ (\geq 0)$
- 帰納法の仮定より: $P \overset{*}{\Rightarrow} x$
- $w = 0x0$ の場合: $P \Rightarrow 0P0 \overset{*}{\Rightarrow} 0x0$
- $w = 1x1$ の場合: $P \Rightarrow 1P1 \overset{*}{\Rightarrow} 1x1$

任意の回文 w は G_{pal} で導出できる

つまり: $w \in L(G_{\text{pal}}) \Leftarrow (w$ は回文である$)$

【十分性の証明おわり】

必要性の証明 (1/2)

示すこと: $w \in L(G_{\mathrm{pal}}) \Rightarrow (w$ は回文である$)$

つまり $P \overset{*}{\Rightarrow} w$ ならば w は回文, を示す

証明は生成規則の適用回数nに関する帰納法

基礎: 生成規則の適用回数 $n = 1$ の場合

- ❖ $P \Rightarrow \varepsilon, P \Rightarrow 0, P \Rightarrow 1$ のいずれかに限る
  - ✔ 終端記号だけにならないといけないから
- ❖ ε, 0, 1 はいずれも回文

必要性の証明 (2/2)

帰納: 生成規則の適用回数 $n+1$ $(n \geq 1)$ の場合

- ❖ 帰納法の仮定: n以下の場合に必要性が成立
- ❖ $n \geq 2$ 場合の導出:
  $P \Rightarrow 0P0$ または$P \Rightarrow 1P1$ のどちらかで始まる
  - ✔ もしそれ以外だと終端記号だけになる
  - ✔ つまり長さ$n \geq 2$の導出になりえない
- ❖ $P \Rightarrow 0P0$ のとき: $P \Rightarrow 0P0 \overset{*}{\Rightarrow} 0x0$
- ❖ $P \Rightarrow 1P1$ のとき: $P \Rightarrow 1P1 \overset{*}{\Rightarrow} 1x1$
- ❖ $0x0$ も $1x1$ も回文: 帰納法の仮定より x は回文

つまり: $w \in L(G_{\mathrm{pal}}) \Rightarrow (w$ は回文である$)$

【必要性の証明おわり】

5.1.6 文形式

文形式とは

文形式 : $S \overset{*}{\underset{G}{\Rightarrow}} \alpha$ である列 α

- ❖ 文法 $G = (V, T, P, S)$
- ❖ $\alpha \in (V \cup T)^*$

開始記号から導出途中の記号列のこと

- ❖ 開始記号と導出が完了したものも含む

教科書の例5.8参照

$L(G) =$ 終端記号のみの文形式の集合

左文形式と右文形式

左文形式

- ❖ 最左導出で現れる文形式のこと

右文形式

- ❖ 最右導出で現れる文形式のこと

5.2 構文木

構文木とは

構文木 ： 木構造による導出の表現

応用例:
コンパイラ内部でのソースプログラム構造の表現

- オブジェクトコードへの翻訳の際の基本となるデータ

5.2.1 構文木の構成

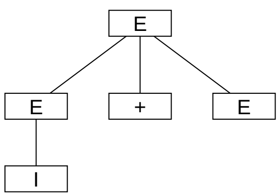
構文木

文法 $G = (V, T, P, S)$ の構文木:
以下の条件を満たす木

- 各内部節点のラベル: 変数
- 各葉のラベル: 変数, 終端記号, あるいは ε
  - ✔ ラベル ε は兄弟節点がない場合に限る
- 内部節点のラベルが A で
  子節点のラベルが左から順に $X_1, X_2, ..., X_k$:
  $A \rightarrow X_1 X_2 \cdots X_k$ が生成規則 P に含まれる

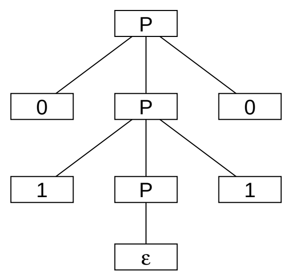
例5.9: 数式のための文法での構文木の例

$E \Rightarrow E + E \Rightarrow I + E$



例5.10: 回文文法での構文木の例

$P \Rightarrow 0P0 \Rightarrow 01P10 \Rightarrow 01\varepsilon 10$



5.2.2 構文木の成果

構文木の成果とは

成果

- 葉のラベルを左から右に並べて得られる文字列
- (構文木の根の変数から導かれる文形式のひとつ)

特に重要な構文木

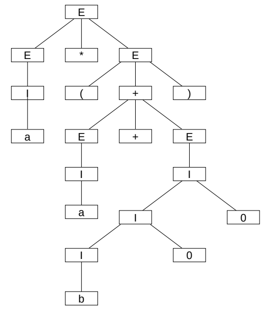
- 根のラベルが出発記号
- 葉のラベルが終端記号あるいはε

文法が生成する言語の別定義

- 出発記号が根で成果が終端記号列の構文木の成果の集合

例5.11: 数式のための文法での構文木の例

$E \overset{*}{\Rightarrow} a * (a + b00)$



5.2.3 推論・導出と構文木
5.2.4 推論・導出から木へ
5.2.5 木から導出へ
5.2.6 導出から再帰的推論へ

同値な5つの導出関係

1. 推論: 推論により
   変数Aの言語に属する終端記号列wを決定可能
2. 導出: $A \overset{*}{\Rightarrow} w$
3. 最左導出: $A \underset{左}{\overset{*}{\Rightarrow}} w$
4. 最右導出: $A \underset{右}{\overset{*}{\Rightarrow}} w$
5. 構文木: Aを根としwを成果とする構文木が存在

証明は省略

おわり

本日のCLEミニレポート

教科書204ページ 問5.1.2(a),(b),(c)