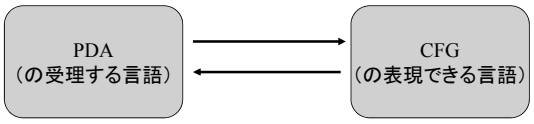
## 4. プッシュダウンオートマトンと文脈自由文法の等価性

1

## 4－1. 目標

ここでは、PDAの受理する言語と、CFGが表現できる言語が等しいことを示す。この言語を文脈自由言語（CFL）と呼ぶ。




2

## CLG→PDAのアイディア

生成途中の文字列をスタックに入れておく。

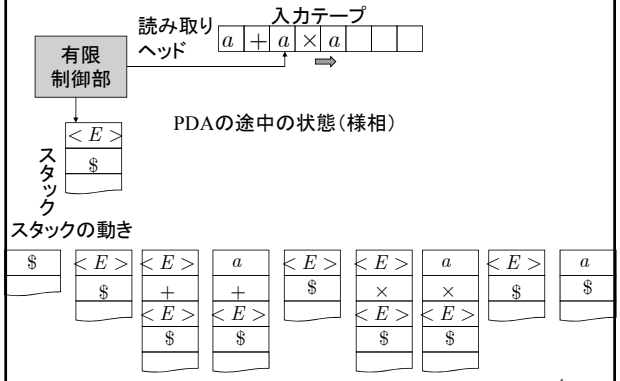
例、下記生成規則における $a+a\times a$ の生成過程。

$R=\{<E>\rightarrow <E>+<E> | <E>\times <E> | a\}$

$$
\begin{aligned}
<E> &\rightarrow <E>+<E> \\
&\rightarrow a+<E> \\
&\rightarrow a+<E>\times <E> \\
&\rightarrow a+a\times <E> \\
&\rightarrow a+a\times a
\end{aligned}
$$

3

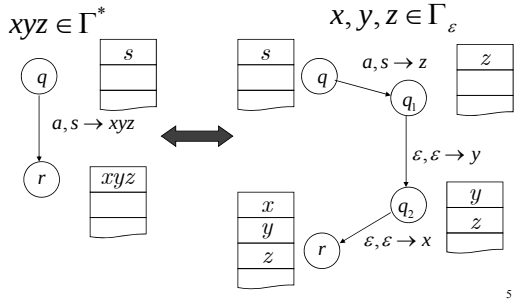
このとき、次のように動作するPDAを構成すればよい。




4

## スタックの拡張

スタックの各セルには、$\Gamma_\varepsilon$ の一文字だけでなく、$\Gamma^*$ の文字列を蓄えることが可能であるとする。




5

CFG → PDA

ある言語がCFGで記述できるとき、
その言語を受理するPDAが存在する。

証明

CFGを $C=(V,\Sigma,R,S)$ とする。

このとき、PDA $P=(Q,\Sigma',\Gamma',\delta,q_{start},F)$ を構成する。

まず、$\Sigma'=\Sigma$, $\Gamma'=V\cup\Sigma$ とする。

また、$Q=\{q_{start},q_{loop},q_{end}\}\cup E$ とし、$F=\{q_{end}\}$ とする。

ここで、$E$ はスタックの拡張を実現する付加的な状態の集合である。

6

状態遷移関数 $\delta$ は次のように定める。

$$\delta(q_{start},\varepsilon,\varepsilon)=\{(q_{loop},S\$)\}$$

規則による導出過程を表す遷移

各 $A\in V$ に対して、

$$\delta(q_{loop},\varepsilon,A)=\{(q_{loop},w)\mid(A\to w)\in R\}$$

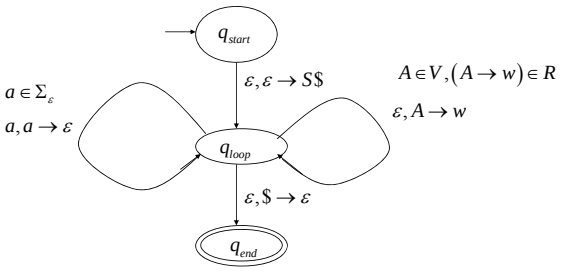
各 $a\in\Sigma$ に対して、

$$\delta(q_{loop},a,a)=\{(q_{loop},\varepsilon)\}$$

左の終端記号の除去。テープ読み取りヘッドの移動を伴う。

$$\delta(q_{loop},\varepsilon,\$)=\{(q_{end},\varepsilon)\}$$

7




$\mathcal{QED}$

8

例

$$C_1=(\{<E>\},\{+,\times,a\},R,<E>)$$
$$R=\{<E>\to<E>+<E>\mid<E>\times<E>\mid a\}$$

$$P_1=(\{q_{start},q_{loop},q_{end},q_1,\cdots,q_k\},\{a,+,\times\},\{a,+,\times,<E>,\$\},\delta,q_{start},\{q_{end}\})$$

$$\delta(q_{start},\varepsilon,\varepsilon)=\{(q_{loop},<E>\$)\}$$

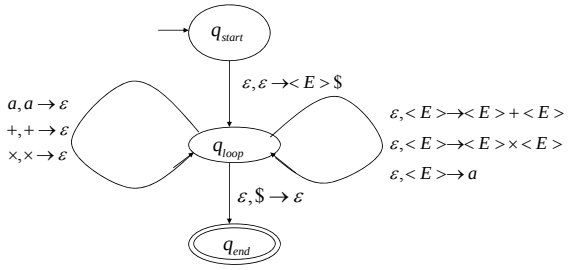
$$\delta(q_{loop},\varepsilon,<E>)=\{(q_{loop},<E>+<E>),(q_{loop},<E>\times<E>),(q_{loop},a)\}$$

$$\delta(q_{loop},a,a)=\{(q_{loop},\varepsilon)\}$$
$$\delta(q_{loop},+,+)=\{(q_{loop},\varepsilon)\}$$
$$\delta(q_{loop},\times,\times)=\{(q_{loop},\varepsilon)\}$$

$$\delta(q_{loop},\varepsilon,\$)=\{(q_{end},\varepsilon)\}$$

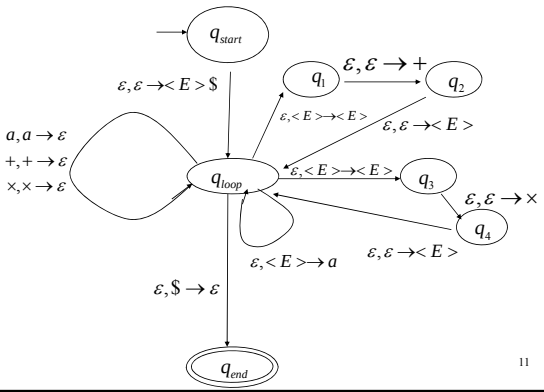
9

（拡張スタックを用いた）等価なPDA




10

1文字スタックへの変換




11

練習

次のCFGが記述している言語を受理するPDAを状態遷移図で示せ。

$$C_2=(\{S,T\},\{a,b\},R',S)$$
$$R'=\{S\to aTb\mid b,$$
$$T\to Ta\mid\varepsilon\}$$

12

## PDA→CFGのアイディア

PDAのスタックの高さを基にして、CFGの規則を生成する。
そのために、PDAを次のように制限する。

1. 唯一つの受理状態 $q_{end}$ を持つ。
2. 受理する前にスタックを空にする。
3. 各遷移は、pushかpopのいずれかであり、同時には行わない。

このように制限しても、PDAの受理能力に変化はない。

13

## PDAからCFGの構成

PDAを $P=(Q,\Sigma,\Gamma,\delta,q_{start},\{q_{end}\})$ として、
CFG $C=(V,\Sigma',R,S)$ を構成する。

1. 変数の設定

$V=\{A_{pq} \mid p,q\in Q\}$

任意の状態の組に対応して変数を用意

2. 終端記号の設定

$\Sigma'=\Sigma$

アルファベットは共通

3. 開始記号の設定

$S=A_{q_{start}q_{end}}$

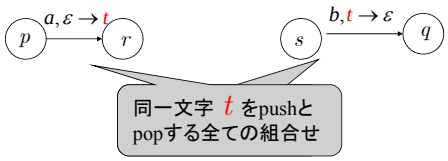
14

## 4. 規則の設定

（1）各々の $p,q,r,s\in Q, t\in\Gamma, a,b\in\Sigma_\varepsilon$ に対して、

$(r,t)\in\delta(p,a,\varepsilon)$ かつ $(q,\varepsilon)\in\delta(s,b,t)$ ならば

$A_{pq}\rightarrow aA_{rs}b$ をRに加える。




同一文字 $t$ をpushとpopする全ての組合せ

15

（2）各々の $p,q,r\in Q$ に対して、

$A_{pq}\rightarrow A_{pr}A_{rq}$ をRに加える。

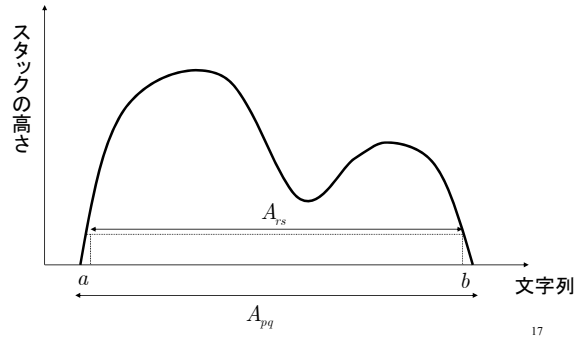
（3）各々の $p\in Q$ に対して、

$A_{pp}\rightarrow\varepsilon$ をRに加える。

16

## イメージ1

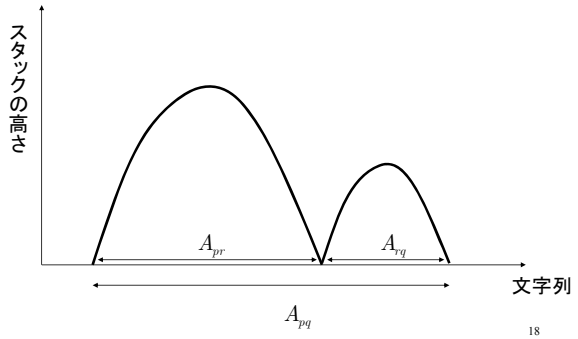
（1）




17

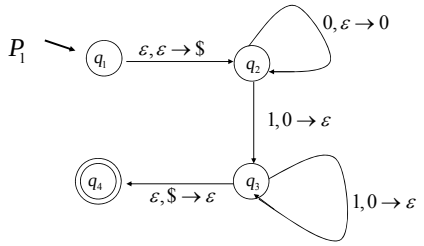
## イメージ2

（2）




18

例




まず、 $V=\{A_{11},A_{12},A_{13},A_{14},\cdots,A_{44}\}$

$\Sigma=\{0,1\}$

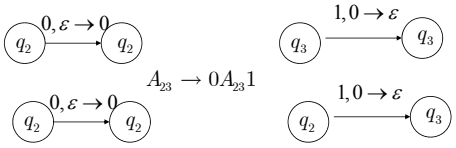
$S=A_{14}$

19

（1） $t=\$$ のとき、


$A_{14}\to A_{23}$

$t=0$ のとき、




$A_{23}\to 0A_{23}1$




$A_{23}\to 0A_{22}1$

20

（2） $A_{13}\to A_{12}A_{23}$

$A_{14}\to A_{12}A_{24}\mid A_{13}A_{34}$

$\vdots$

（3） $A_{11}\to\varepsilon$

$A_{22}\to\varepsilon$

$\vdots$

なお、規則としては、以下だけで生成できることがわかる。

$A_{14}\to A_{23}$

$A_{23}\to 0A_{23}1$

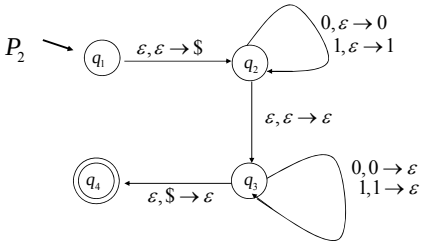
$A_{23}\to 0A_{22}1$

$A_{22}\to\varepsilon$

21

練習

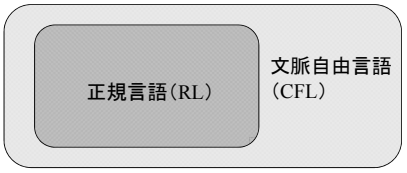
次のPDAが受理する言語を生成する
CFGを示せ。
（変数、規則は、必要部分だけでよい。）




22

正規言語(RL)と文脈自由言語(CFL)

正規言語は有限オートマトンで受理される。
文脈自由言語はプッシュダウンオートマトンで受理される。
プッシュ機能を用いなければPDAはDFAとしても機能する。
よって、正規言語すべてをPDAは受理する。
逆に、正規言語でない言語もPDAは受理できる。
したがって、言語の包含関係は下図のようになる。




23

(CFLの)ポンピング補題

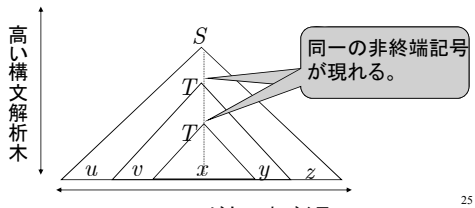
(CFLの)ポンピング補題

AがCFLであるならば、ある数 $p$ (ポンピング長)が
存在して、$p$より長い任意の文字列 $s\in A$ に対して、
次を満たすように $s$ を
$s=uvxyz$
に分割できる。
1. 各 $i\geq 0$ について、 $uv^ixy^iz\in A$
2. $|vy|\geq 1,(y\neq\varepsilon)$
3. $|vxy|\leq p$

24

## ポンピング補題の意味

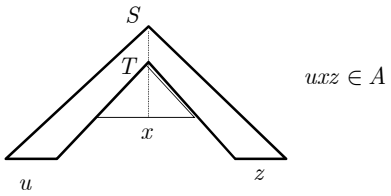
ものすごく長い文字列では、構文解析木の高さも高くなる。
このとき、開始変数から終端記号までの“道”上に
同じ非終端記号が現れてしまう。
このように、いったん同じ非終端記号が現れたときには、
この非終端記号を繰り返し適用することによって、
文字列を長くできる。




$s = uvxyz$ がものすごく長い

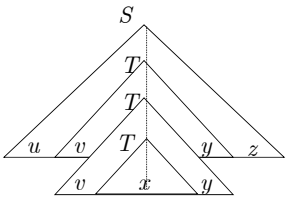
25

構文解析木の葉から開始記号までの道上に
同じ非終端記号が現れたとき、
下のような言語もCFGにより生成されるはずである。




$uxz \in A$

26




$uvvxyyz \in A$

$uv^ixy^iz \in A$

27

## ポンピング補題の証明

CFL $A$を認識するCFGをGとし、
$b$ を基礎の右辺にある文字の最大数とする。
$b \geq 2$ としてよい。

このとき、構文解析木の各節点は、 $b$ より多くの子
を持つことができない。
したがって、開始記号からの距離が $h$ であるところには、
高々 $b^h$ 個の節点しかない。

ここで、$|V|$ をGの非終端記号の総数とする。
ポンプ長 $p$ を $b^{|V|+2}$ とおく。
このとき、構文解析木の高さ、すなわちSから葉までの
道の長さは、少なくとも $|V|+2$ である。

28

$s$ を少なくとも長さ $p$ であるAの文字列とする。
このとき $S$ を生成する構文解析木の高さは、少なくとも、
$|V|+2$ である。
構文解析木において、終端記号は、葉だけであるので、
開始記号Sから葉の一つ手前まではすべて非終端記号である。
すなわち、$|V|+1$個の非終端記号が出現しているはずである。

一方、非終端記号は$|V|$個しかないので、
同じ非終端記号が繰り返して出現しているはずである。
この記号を $T$ とあらわす。

この場合、前述の図のように、$s = uvxyz$ と分割できること
がわかる。

$\mathcal{QED}$

29

## CFLの限界

次の言語は文脈自由言語ではない。

$$C = \{a^n b^n c^n \mid n \geq 0\}$$

$C = \{w \mid w$はa, b, c順に同じ文字数だけ繰り返す。$\}$
$= \{\varepsilon, abc, aabbcc, aaabbbccc, \cdots\}$

30

**証明**

ポンピング補題を用いる。

CがCFLであると仮定する。(背理法の仮定)
$p$ をポンピング長とする。
文字列を $s = a^p b^p c^p$ とする。
このとき、明らかに、$|s| \geq p$ である。

このとき、ポンピング補題より、$s$ は
$s = uxyz$
と分割できるはずである。

31

このとき、次の2つの場合に分けて考える。
(1) $v$ と $y$ はどちらも1種類の文字からなる。
(2) $v$ と $y$ のどちらかが2種類以上の文字からなる。

場合(1)、
このときは、文字列 $uv^2xy^2z$ は
同じ個数の $a, b, c$ を含むことができない。
したがって矛盾が生じる。

場合(2)、このときは、文字列 $uv^2xy^2z$ では
同じ個数の $a, b, c$ を含むことかもしれない。
しかし、$a, b, c$ の順序に狂いか生じる。
よって、矛盾である。

いずれの場合も矛盾が生じるので、
命題が証明された。 *QED*

32