

情報に関する理論

スライド作成者 大東誠弥

目次

- 情報量, 平均情報量
- オートマトン
- 木の走査と逆ポーランド表記法
- BNF(Backus-Naur Form)記法
- 計算量(オーダー)
- AI

情報量

- 情報量 ... ある事象が起きたとき,それが
どのくらい起こりにくいかを表す尺度



起こりにくい事象ほど情報量は高くなる.

情報量

- 選択情報量(自己エントロピー)...ある事象が起こるときの情報量

$$\text{選択情報量} = -\log_2 P$$

(Pは事象が起こる確率)

(例)雨が降る確率が50%の時

$$\begin{aligned}\text{雨の選択情報量} &= -\log_2 \frac{1}{2} \\ &= -\log_2 2^{-1} = 1\end{aligned}$$

平均情報量

- 平均情報量(エントロピー)...全体の平均を取ったときの情報量

(例) 晴れ25%,曇り25%,雨が50%の場合

晴れ,曇りの選択情報量 $= -\log_2 0.25 = 2$

$$\begin{aligned}\text{平均情報量} &= 2 \times 0.25 + 2 \times 0.25 + 1 \times 0.5 \\ &= 1\end{aligned}$$

問題

a, b, c, dの4文字から成るメッセージを符号化してビット列にする方法として表のア～エの4通りを考えた。この表はa, b, c, dの各1文字を符号化するときのビット列を表している。メッセージ中でのa, b, c, dの出現頻度は、それぞれ50%, 30%, 10%, 10%であることが分かっている。符号化されたビット列から元のメッセージが一意に復号可能であって、ビット列の長さが最も短くなるものはどれか。

	a	b	c	d
ア	0	1	00	11
イ	0	01	10	11
ウ	0	10	110	111
エ	00	01	10	11

(平成28年春 応用情報技術者試験 午前 問4)

解説

まず、「符号化されたビット列から元のメッセージが一意に復元可能か」を考える必要があります。

アの場合には、例えばビット列が000のときに、元のメッセージがaaaなのかacなのかcaなのか区別できません。

イも、ビット列が010のとき、baなのかacなのか区別できません。これらの場合には、いくらビット列の長さが短くても採用できません。

エの場合は、すべてが2ビットなので、2ビットごとに区切れば、0110はbcというように一意に復号可能です。しかし、ビット列の長さは2ビット必要になります。

ウの場合、元のメッセージを一意に識別することが可能です。さらに、出現頻度がa, b, c, dそれぞれ50%, 30%, 10%, 10%なので、平均のビット数は、

$$1 \times 0.5 + 2 \times 0.3 + 3 \times 0.1 + 3 \times 0.1 = 1.7 \text{ [ビット]}$$

となり、エの場合(2ビット)よりも短くなります。

◀解答> ウ

オートマトン

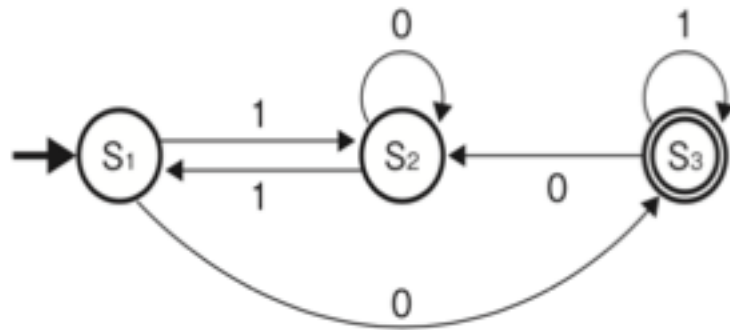
1. 外から連続で情報が入力される
2. 内部に状態を保持する
3. 外へ情報を出力する

これら 3 つの条件を満たすシステムモデルを
オートマトンと呼ぶ。

これらの入出力や状態の数が、有限である場合は
有限オートマトンと呼ぶ。

問題

次に示す有限オートマトンが受理する入力列はどれか。ここで、 S_1 は初期状態を、 S_3 は受理状態を表している。



ア 1011 イ 1100 ウ 1101 エ 1110

(平成21年春 応用情報技術者試験 午前 問3)

解説

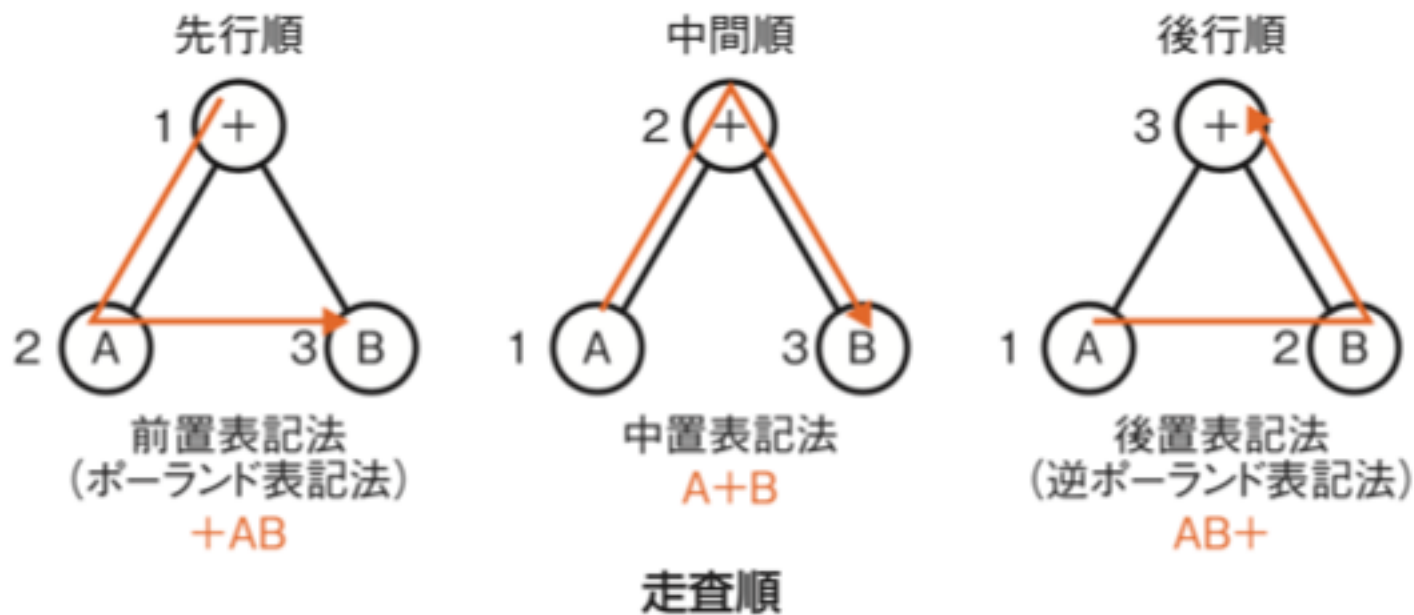
初期状態が S_1 なので、 S_1 を起点としてそれぞれの文字列を入力して状態遷移を見ていきます。

ア	1011	$S_1 \xrightarrow{1} S_2 \xrightarrow{0} S_2 \xrightarrow{1} S_1 \xrightarrow{1} S_2$
イ	1100	$S_1 \xrightarrow{1} S_2 \xrightarrow{1} S_1 \xrightarrow{0} S_3 \xrightarrow{0} S_2$
ウ	1101	$S_1 \xrightarrow{1} S_2 \xrightarrow{1} S_1 \xrightarrow{0} S_3 \xrightarrow{1} S_3$ 受理状態
エ	1110	$S_1 \xrightarrow{1} S_2 \xrightarrow{1} S_1 \xrightarrow{1} S_2 \xrightarrow{0} S_2$

ア、イ、エは S_2 、ウのみ S_3 になります。受理状態とは、その状態で終了すると受理するという状態で、この問題の場合、受理状態は S_3 です。

《解答》 ウ

木の走査と逆ポーランド表記法



問題

後置表記法(逆ポーランド表記法)では, 例えば, 式 $Y = (A - B) \times C$ を $YAB - C \times =$ と表現する。

次の式を後置表記法で表現したものはどれか。

$$Y = (A + B) \times (C - (D \div E))$$

ア $YAB + C - DE \div \times =$

イ $YAB + CDE \div - \times =$

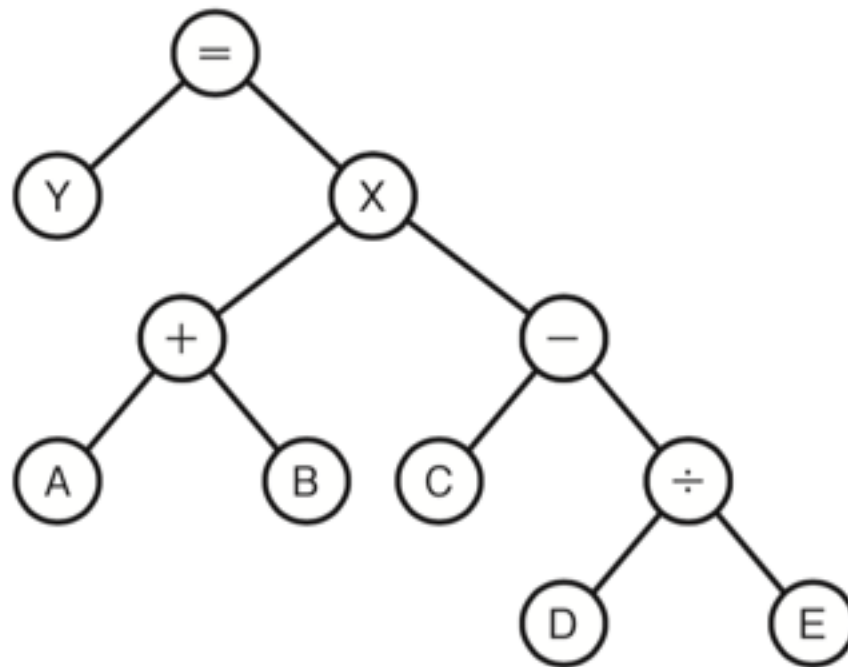
ウ $YAB + EDC \div - \times =$

エ $YBA + CD - E \div \times =$

(平成22年秋 応用情報技術者試験 午前 問1)

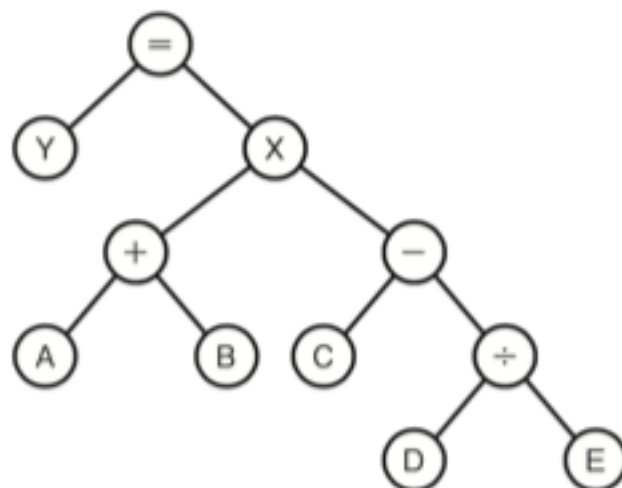
ヒント

$$Y = (A + B) \times \{C - (D \div E)\}$$



解説

問題文の式は中置表記法なので、まず次のように木構造に置き換えます。



この木を逆ポーランド表記法にするには後行順で順に読んでいけばいいので、次のようになります。

$$YAB + CDE \div - \times =$$

《解答》 イ

BNF(Backus-Naur Form)記法

- 文法などの形式を定義するために用いられる言語
- ひたすら再起を用いて表記していく

< > 識別子

| または

::= イコール

(例)

- 1桁以上の数字(先頭の数字は0以外)を表す場合

$\langle \text{number} \rangle ::= \langle \text{non-zero-digit} \rangle \mid \langle \text{non-zero-digit} \rangle \langle \text{digits} \rangle$

$\langle \text{digits} \rangle ::= \langle \text{digit} \rangle \mid \langle \text{digit} \rangle \langle \text{digits} \rangle$

$\langle \text{non-zero-digit} \rangle ::= 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

$\langle \text{digit} \rangle ::= 0 \mid \langle \text{non-zero-digit} \rangle$

問題

あるプログラム言語において、識別子 (identifier) は、先頭が英字で始まり、それ以降に任意個の英数字が続く文字列である。これをBNFで定義したとき、aに入るものはどれか。

$\langle \text{digit} \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

$\langle \text{letter} \rangle ::= A \mid B \mid C \mid \dots \mid X \mid Y \mid Z \mid a \mid b \mid c \mid \dots$
 $\mid x \mid y \mid z$

$\langle \text{identifier} \rangle ::= \boxed{\text{a}}$

ア $\langle \text{letter} \rangle \mid \langle \text{digit} \rangle \mid \langle \text{identifier} \rangle \langle \text{letter} \rangle \mid$
 $\langle \text{identifier} \rangle \langle \text{digit} \rangle$

イ $\langle \text{letter} \rangle \mid \langle \text{digit} \rangle \mid \langle \text{letter} \rangle \langle \text{identifier} \rangle \mid$
 $\langle \text{identifier} \rangle \langle \text{digit} \rangle$

ウ $\langle \text{letter} \rangle \mid \langle \text{identifier} \rangle \langle \text{digit} \rangle$

エ $\langle \text{letter} \rangle \mid \langle \text{identifier} \rangle \langle \text{digit} \rangle \mid \langle \text{identifier} \rangle$
 $\langle \text{letter} \rangle$

解説

記号の「::=」は「定義する」、「|」は「いずれか」を意味します。
したがって、

$$\langle \text{digit} \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

は、 $\langle \text{digit} \rangle$ が0から9までの数字であることを定義し、

$$\langle \text{letter} \rangle ::= A \mid B \mid C \mid \cdots \mid X \mid Y \mid Z \mid a \mid b \mid c \mid \cdots \mid x \mid y \mid z$$

は、 $\langle \text{letter} \rangle$ が大文字か小文字の英字であることを定義しています。

識別子 $\langle \text{identifier} \rangle$ については、問題文に「先頭が英字で始まり、それ以降に任意個の英数字が続く文字列」と記載されています。
つまり、識別子 $\langle \text{identifier} \rangle$ は、

$\langle \text{identifier} \rangle ::= \langle \text{letter} \rangle$ …最初の1文字(英字)のみか、そこに再帰を用いて、

$$\langle \text{identifier} \rangle ::= \langle \text{identifier} \rangle \langle \text{digit} \rangle$$

または、

$$\langle \text{identifier} \rangle ::= \langle \text{identifier} \rangle \langle \text{letter} \rangle$$

というように、2番目以降には英数字(英字か数字のどちらか)が続くという状態を表現できます。したがって、次のようになります。

$$\langle \text{identifier} \rangle ::= \langle \text{letter} \rangle \mid \langle \text{identifier} \rangle \langle \text{digit} \rangle \mid \langle \text{identifier} \rangle \langle \text{letter} \rangle$$

《解答》 工

計算量(オーダー)

- あるプログラムにかかる時間を,入力データの増加量で表したもの.

(例)入力データ数(n)が増加した時,その計算量が n^2 に比例して増加する時.


$$O(n^2)$$

オーダーは n が非常に大きい場合のみを考えるため,数値が小さい場合は無視をして,定数も無視する.

計算量(オーダー)

(例)入力データ数(n)が増加した時,その計算量が $3n^2 + n + 2$ に比例して増加する時.

計算量(オーダー)

(例) 入力データ数(n)が増加した時, その計算量が
 $3n^2 + n + 2$ に比例して増加する時.



$$O(n^2)$$

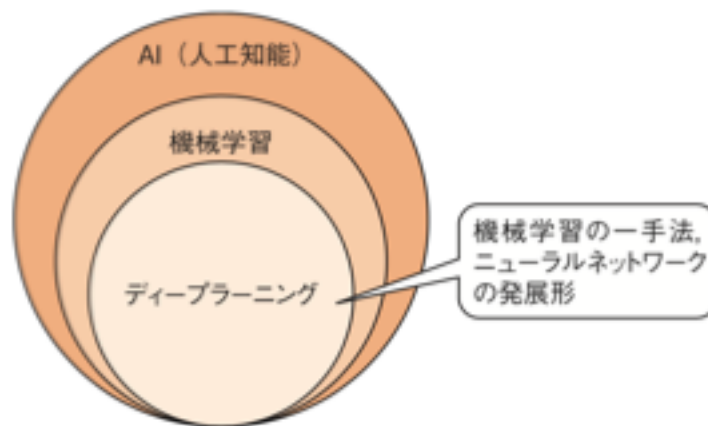
計算量(オーダー)

代表的な O (オーダー)とその例

O (オーダー)	例 (アルゴリズム)
$O(1)$	ハッシュ
$O(\log n)$	2分探索
$O(n)$	線形探索
$O(n \log n)$	クイックソート, シェルソート
$O(n^2)$	バブルソート, 挿入ソート

AI(artificial intelligence)

- AI技術でよく用いられるのは機械学習
- 機械学習の中でもニューラルネットワークから派生したものがディープラーニング(深層学習)



AIと機械学習、ディープラーニングの関係

覚えよう！

- 木の操作順は, 先行順, 中間順, 後行順
(後行順で読むと逆ポーランド表記法)
- 二分探索の計算量は $O(\log n)$
クイックソートの計算量は $O(n \log n)$