1 アルゴリズムとプログラミング

(1)

28 → 8
35 → 6

(2)

(2-1)

0	1	2	3	4	5	6	7
-1	-1	39	43	-1	45	-1	27

8	9	10	11	12	13	14	15
25	-1	-1	31	-1	-1	65	95

16	17	18	19
76	-1	-1	59

(2-2)

(ア) h = 0
(イ) h
(ウ) h + 1
(エ) -1

(2-3) SKIPを4とした場合 順に実行した時、43を処理する前は以下のようになる

0	1	2	3	4	5	6	7
-1	-1	-1	39	-1	45	-1	27
8	9	10	11	12	13	14	15
-1	25	-1	31	-1	65	-1	95
16	17	18	19				
76	-1	-1	59				

この時 43は 3 → 7 → 11 → 15 → 19 → 3 と遷移し、無限ループで処理が終了しない

(2-4) 互いに素

1 アルゴリズムとプログラミング

(1)

	21				15			28	
0	1	2	3	4	5	6	7	8	9

	21				15	35		28	
0	1	2	3	4	5	6	7	8	9

(2-1)

-1	-1	39	43	-1	45	-1	27	25	-1	-1	31	-1	-1	65	95	76	-1	-1	59
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

(2-2)

(ア) hash(d)

(イ) h

(ウ) next(h)

(エ) -1

(2-3)

			39		45		27		25		31		65		95	76			59
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

43をtableに格納する際に何回関数nextを実行しても
空いているセルが見つからずに19行目のwhile文が無限に実行されるから。

(2-4)

MAXとSKIPが互いに素でなければならない。

$\boxed{2}$

(1)

(1-1) $A(a_1, a_0)$, $B(b_1, b_0)$

b_1b_0 \ a_1a_0	00	01	11	10
00	1	1	1	1
01	0	1	1	1
11	0	0	1	0
10	0	0	1	1

Sの出力

$$S = \bar{b}_1\bar{b}_0 \vee a_1\bar{b}_1 \vee a_1\bar{b}_0 \vee a_1a_0 \vee a_0\bar{b}_1$$

(1-2) i) $S_1=0$のとき（$E_1=1$にならないので、$E_1=1$の場合はdon't care）

$a_3a_2 < b_3b_2$より、$(S, E) = (0, 0)$

ii) $S_1=1$、$E_1=0$のとき

$a_3a_2 > b_3b_2$より、$(S, E) = (1, 0)$

iii) $S_1=1$、$E_1=1$、$S_0=0$のとき（$E_0=1$にならないので、$E_0=1$の場合はdon't care）

$a_3a_2 = b_3b_2$かつ$a_1a_0 < b_1b_0$より　$(S, E) = (0, 0)$

iv) $S_1=1$、$E_1=1$、$S_0=1$、$E_0=0$のとき

$a_3a_2 = b_3b_2$かつ$a_1a_0 > b_1b_0$より　$(S, E) = (1, 0)$

v) $S_1=1$、$E_1=1$、$S_0=1$、$E_0=1$のとき

$A=B$より　$(S, E) = (1, 1)$

E_1E_0 \ S_1S_0	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	d	d	1	d
10	d	d	1	0

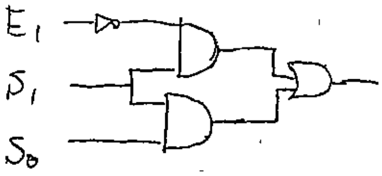
Sの出力

E_1E_0 \ S_1S_0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	d	d	1	d
10	d	d	0	0

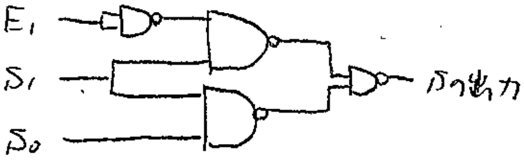
Eの出力

$S = S_1\bar{E}_1 \vee S_1S_0$　　　$E = E_1E_0$

(1-3)



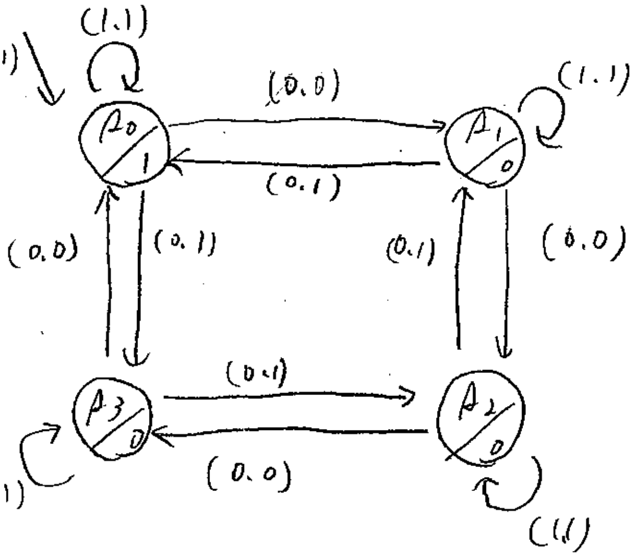
Sの出力　なので…



ANDとNOTをNANDにおきかえる。

(2)

(2-1)



(2-2)

	Q_1 Q_0
p_0	0 0
p_1	0 1
p_2	1 0
p_3	1 1

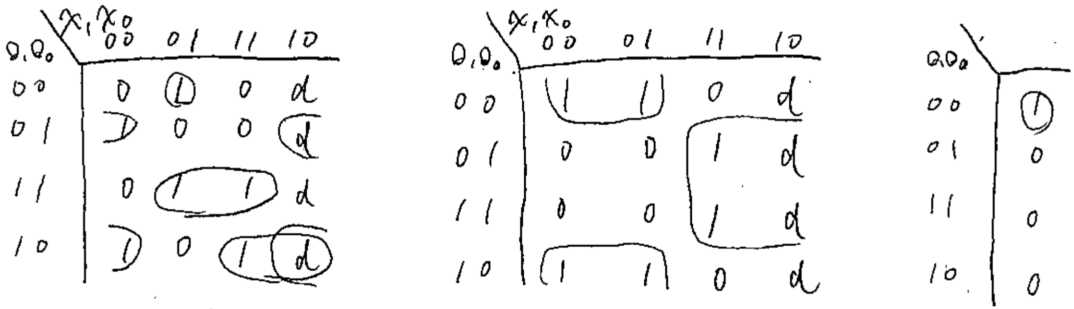
状態割当て

現在の状態	次の状態 x_1x_0= 00	01	10	11	現在の出力
p_0	p_1	p_3	×	p_0	1
p_1	p_2	p_0	×	p_1	0
p_2	p_3	p_1	×	p_2	0
p_3	p_0	p_2	×	p_3	0

状態遷移出力表

(2-3)

Q_1Q_0	$Q_1^+Q_0^+$ x_1x_0= 00	01	10	11	Z
0 0	0 1	1 1	dd	0 0	1
0 1	1 0	0 0	dd	0 1	0
1 0	1 1	0 1	dd	1 0	0
1 1	0 0	1 0	dd	1 1	0



Q_1Q_0 \ x_1x_0	00	01	11	10
00	0	1	0	d
01	1	0	0	d
11	0	1	1	d
10	1	0	1	d

Q_1^+

Q_1Q_0 \ x_1x_0	00	01	11	10
00	1	1	0	d
01	0	0	1	d
11	0	0	1	d
10	1	1	0	d

Q_0^+

Q_1Q_0	
00	1
01	0
11	0
10	0

Z

$$D_1 = \overline{Q_1} Q_0 \overline{x_0} \vee Q_1 Q_0 x_0 \vee Q_1 \overline{Q_0}\, \overline{x_0} \vee Q_1 \overline{Q_0} x_1 \vee \overline{Q_1}\, \overline{Q_0}\, \overline{x_1} x_0$$

$$D_0 = \overline{Q_0}\, \overline{x_1} \vee Q_0 x_1$$

$$Z = \overline{Q_1}\, \overline{Q_0}$$

2 論理回路

(1-1)

S \ a_1a_0	00	01	11	10
b_1b_0 00	1	1	1	1
01	0	1	1	1
11	0	0	1	0
10	0	0	1	1

$S = \bar{b}_1\bar{b}_0 \vee a_1a_0 \vee a_0\bar{b}_1 \vee a_1\bar{b}_1 \vee a_1\bar{b}_0$

(1-2)

S \ S_0E_0	00	01	11	10
S_1E_1 00	0	d	0	0
01	d	d	d	d
11	0	d	1	1
10	1	d	1	1

$S = S_1\bar{E}_1 \vee S_1S_0$

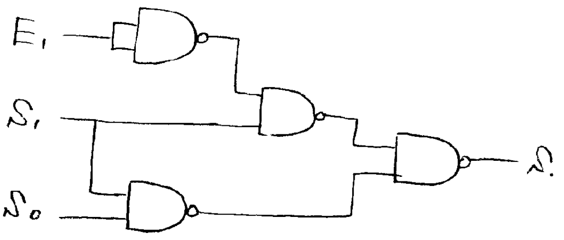
E \ S_0E_0	00	01	11	10
S_1E_1 00	0	d	0	0
01	d	d	d	d
11	0	d	1	0
10	0	d	0	0

$E = E_1E_0$

※ d は don't care

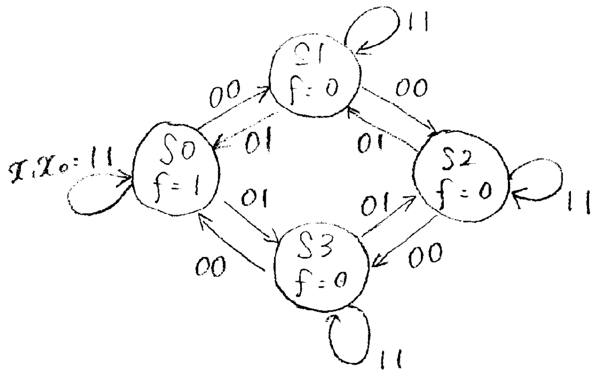
(1-3)

$$\begin{aligned} S &= S_1\bar{E}_1 \vee S_1S_0 \\ &= S_1 \cdot nand(E_1, E_1) \vee S_1S_0 \\ &= \overline{\overline{S_1 \cdot nand(E_1, E_1)} \cdot \overline{S_1S_0}} \\ &= \overline{nand(S_1, nand(E_1, E_1)) \cdot nand(S_1, S_0)} \\ &= nand(nand(S_1, nand(E_1, E_1)), nand(S_1, S_0)) \end{aligned}$$



$\boxed{2}$

(2-1)



(2-2) 状態遷移表.

状態	Q_1Q_0	x_1x_0	次状態	次Q_1Q_0
S0	00	00	S1	01
		01	S3	11
		10		
		11	S0	00
S1	01	00	S2	10
		01	S0	00
		10		
		11	S1	01
S2	10	00	S3	11
		01	S1	01
		10		
		11	S2	10
S3	11	00	S0	00
		01	S2	10
		10		
		11	S3	11

↑ D_1D_0のこと.

出力表.

状態	Q_1Q_1	f
S0	00	1
S1	01	0
S2	10	0
S3	11	0

(2-3)

Q_1Q_0

D_1	00	01	11	10
00	0	1	0	1
01	1	0	1	0
11	0	0	1	1
10	d	d	d	d

x_1x_0

$$D_1 = \overline{Q_1}\,\overline{Q_0}\,\overline{x_1}x_0 \vee \overline{Q_1}Q_0\overline{x_0} \vee Q_1Q_0x_0 \vee Q_1x_1 \vee Q_1\overline{Q_0}\,\overline{x_0}$$

Q_1Q_0

D_0	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	0	1	1	0
10	d	d	d	d

x_1x_0

$$D_0 = \overline{Q_0}\,\overline{x_1} \vee Q_0x_1$$

$$f = \overline{Q_1}\,\overline{Q_0}$$

$\boxed{3}$ (1) (1-1)

(a) ウ　(b) ア　(c) カ　(d) オ
(e) コ　(f) オ　(g) イ

(1-2)

LRU

使用順を記憶しておく必要があるので
実装は難しい
時間的局所性があるので、
ブロック枠の数が少なくてもヒット率は高い

FIFO

キューを使用できるので
実装は容易である

ページ枠の数を増加させてもヒット率が
高くならない場合もある

(1-3)

(1-3-1)

キャッシュのブロック枠数が4、
セット数2　より　4÷2=2　群は2個
よって群番号に1ビット使用する
1ブロックの大きさ4語より、
ブロック内アドレスに2ビット使用する
よって群内ブロック番号には、
8-1-2=5　5ビット使用する

(1-3-2)

a[i]	0	4	8	53	54	55	56	4	20	21
ブロック番号	0	1	2	13	13	13	14	1	5	5
群	0	1	0	1	1	1	0	1	1	1

		0	4	8	53	54	55	56	4	20	21
キャッシュヒット						○	○		○		○
キャッシュ枠	00	0	0	0	0	0	0	14	14	14	14
	01			2	2	2	2	2	2	2	2
	10		1	1	1	1	1	1	1	1	1
	11				13	13	13	13	13	5	5

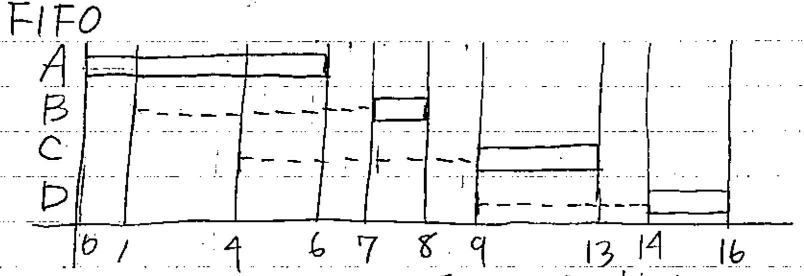
ヒット率は、$\frac{4}{10} = 0.4$

(2) (2-1) (a) カ (b) キ (c) シ

(d) ウ (e) ソ (f) キ

(2-2)

FIFO

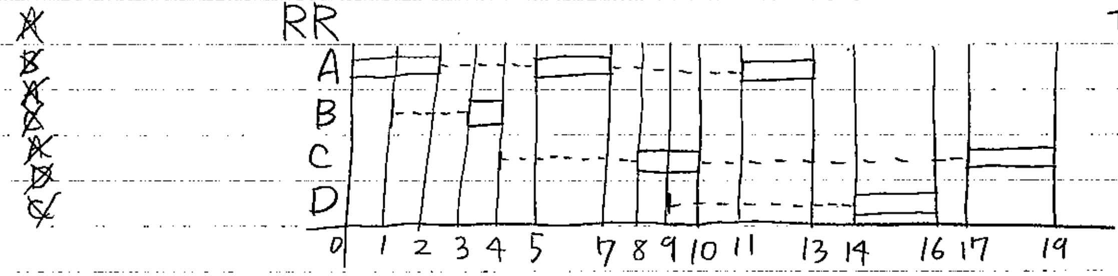


	TAT	応答
A	6	0
B	7	6
C	9	5
D	7	5

平均ターンアラウンドタイム 7.25

平均応答時間 4

RR



	TAT	応答
A	13	0
B	3	2
C	15	4
D	7	5

平均ターンアラウンドタイム 9.5

平均応答時間 2.75

3 計算機システムとシステムプログラム

(1-1)

(a) ウ・時間的局所性
(b) ア・空間的局所性
(c) カ・完全連想マッピング
(d) オ・直接マッピング
(e) コ・低
(f) カ・完全連想マッピング
(g) イ・セット連想マッピング

(1-2)

・実装の容易さ

LRU：それぞれのブロック枠がいつ参照されたかを記憶しておく必要があるため、複雑になる。

FIFO：ブロックをキュー構造で保持するだけなので、簡単である。

・ブロック枠の数とヒット率の関係

LRU：ブロック枠の数がnのときの動きが、n+1のときの動きに含まれるスタックアルゴリズムであるため、ブロック枠の数がふえると必ずヒット率が向上する、あるいは変化しない。

FIFO：ブロック枠の数が変化すると保持するブロックの組合せも変化してしまうため、ブロック枠の数をふやしてもヒット率が向上するとは限らない。（ブロック枠の数をふやしたとき、あるブロック参照列についてヒット率が下がってしまうことをBeladyの例外という。）

(1-3-1)

$$\underbrace{0\ 0\ 0\ 0\ 0}_{\text{使用可能}}\ \underbrace{0}_{\text{セット番号を指定}}\ \underbrace{0\ 0}_{\text{ブロック内の4語から1語指定}}$$

したがって、セット内で各ブロックを識別するタグには、5ビット用いることができる。

(1-3-2)

a[0] ~ a[3] : ブロック0
4 ~ 7 : 1
8 ~ 11 : 2
12 ~ 15 : 3
16 ~ 19 : 4
20 ~ 23 : 5
24 ~ 27 : 6
28 ~ 31 : 7
32 ~ 35 : 8
36 ~ 39 : 9
40 ~ 43 : 10
44 ~ 47 : 11
48 ~ 51 : 12
52 ~ 55 : 13
56 ~ 59 : 14
60 ~ 63 : 15

	セット1		セット2		ヒット
a[0]	0				
4	0		1		
8	0	2	1		
53	0	2	1	13	
54	0	2	1	13	○
55	0	2	1	13	○
56	14	2	1	13	
4	14	2	1	13	○
20	14	2	1	5	
21	14	2	1	5	○
	ブロックNo偶数		奇数		

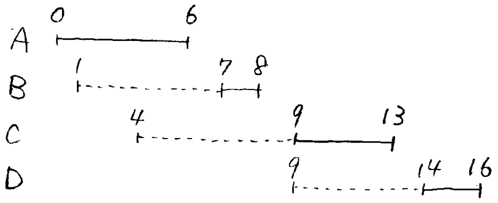
ヒット率は 0.4

(2-1)

(a) カ・実行可能状態
(b) キ・ディスパッチ
(c) シ・実行状態
(d) ウ・待ち状態
(e) ソ・プリエンプション
(f) イ・コンテクストスイッチ

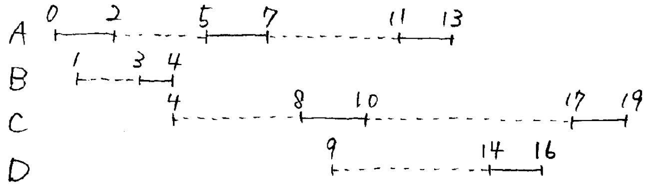
(2-2)

FIFO方式



平均ターンアラウンドタイム：(6+7+9+7)÷4 = 7.25
平均応答時間：(0+6+5+5)÷4 = 4

RR方式



平均ターンアラウンドタイム：(13+3+15+7)÷4 = 9.5
平均応答時間：(0+2+4+5)÷4 = 2.75

2008 $\boxed{8}$ 情報論理学

(1)

(1-1) (a) $p(x)$ を $x \geq 0$ とすると, $\forall x(p(x) \rightarrow p(a))$ は a に関わらず恒真

$p(x)$ を $x \leq 2$ とすると, 〃 なので.

解釈の例として, $p(x)$ を $x \geq 0$ とし, $a=1$ とする,

$p(x)$ を $x \leq 2$ とし, $a=2$ とする.

などがあげられる.

(b) 解答例: $f(x) = x+0$

$p(x) = x \geq 0$

(c) 解答例1: $p(x) = x < 0$ (理由. $\forall x(p(x) \rightarrow (\neg p(a) \vee \neg p(b)))$ で, $p(x)$ が $x=0,1,2$ で全て偽なので,

a, b はなんでも OK

$(p(x) \rightarrow (\neg p(a) \vee \neg p(b)))$ は $\forall x$ で真

解答例2: $p(x) = x \geq 1$, $a=0$, $b=1$

$$\underset{偽}{p(0)} \rightarrow (\underset{真}{\neg p(0)} \vee \underset{偽}{\neg p(1)}) = 真$$

$$\underset{真}{p(1)} \rightarrow (\underbrace{\underset{真}{\neg p(0)} \vee \underset{偽}{\neg p(1)}}_{真}) = 真$$

$$p(2) \rightarrow \neg p(0) \vee \neg p(1) = 真$$

上に同じく.

(1-2) (a) $p(x) = x < 0$ $q(x) = x \geq 0$

(b) $\neg \forall x(p(x) \wedge q(x)) = \exists x \neg (p(x) \wedge q(x))$

$= \exists x(\neg p(x) \vee \neg q(x))$ なので,

$p(x) = x=0$ $q(x) = x=1$ など.

(c) $\neg \exists x(\neg p(x) \wedge \neg q(x)) = \forall x(p(x) \vee q(x))$ なので,

$p = x < 0$, $q = x \geq 0$ など。

情報論理学 2008 続き

(1-3) $p(x) = x < 0$ $q(x) = x \geq 0$ とすれば,

$\forall x(p(x) \vee q(x))$ は真 $\forall x\, p(x)$ は偽, $\forall x\, q(x)$ も偽 なので,

$\forall x\, p(x) \vee \forall x\, q(x)$ は偽

(2)
(2-1) $A = (\forall y \forall x\, p(x,y) \wedge \exists y \forall x(p(x,y) \rightarrow q(x))) \rightarrow \forall z\, q(z)$

$= \neg(\forall y \forall x\, p(x,y) \wedge \exists y \forall x(p(x,y) \rightarrow q(x))) \vee \forall z\, q(z)$ $(\because p \rightarrow q \equiv \neg p \vee q)$

すなわち,

$\neg A = (\forall y \forall x\, p(x,y) \wedge \exists y \forall x(p(x,y) \rightarrow q(x))) \wedge \neg \forall z\, q(z)$

$= \forall y \forall x\, p(x,y) \wedge \exists y \forall x(p(x,y) \rightarrow q(x)) \wedge \exists z \neg q(z)$

$= \exists y(\forall y \forall x\, p(x,y) \wedge \forall x(p(x,y) \rightarrow q(x)) \wedge \exists z \neg q(z))$ (∵ $\exists y$ を出す.)

$= \exists y \exists z(\forall y \forall x\, p(x,y) \wedge \forall x(p(x,y) \rightarrow q(x)) \wedge \neg q(z))$ (∵ $\exists z$ を出す)

$= \exists y \exists z \forall w(\forall x\, p(x,w) \wedge (p(w,y) \rightarrow q(w)) \wedge \neg q(z))$ (∵ $\forall y \forall x\, p(x,y)$ の y を w に, $\forall x(p(x,y) \rightarrow q(x))$ の x を w に おきかえて まとめて $\forall w$ で出す)

$= \exists y \exists z \forall w \forall x(p(x,w) \wedge (\neg p(w,y) \vee q(w)) \wedge \neg q(z))$ // (→を消す)

(2-2) $\forall w \forall x(p(x,w) \wedge (\neg p(w,a) \vee q(w)) \wedge \neg q(b))$ //

$y = a$, $z = b$ で おきかえ.

(2-3) $p(x,w)$ ---- (1)

$\neg p(w,a) \vee q(w)$ --(2),

$\neg q(b)$ ----- (3),

(2) の w に b を代入

$\neg p(b,a) \vee q(b)$ --- ④

③と④から $\neg p(b,a)$ --- ⑤

① に $x=b$, $w=a$ を代入

$p(b,a)$ --- ⑥

⑤と⑥から,

0 //

0が導出されたので, A' は充足不能

9

(1)

(1-1)

10, 010, 1110, 0010

(1-2)

ε-閉包

h : {h,i,j,k}
i : {i}
j : {j,k}
k : {k}

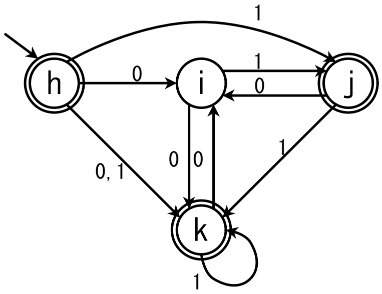
M_2 の状態遷移表（ε 動作を除く）

	0	1
h	-	-
i	k	j
j	-	-
ⓚ	i	k

M_2' の状態遷移表

	0	1
ⓗ	i, k	j, k
i	k	j
ⓙ	i	k
ⓚ	i	k

M_2' の状態遷移図



(1-3)

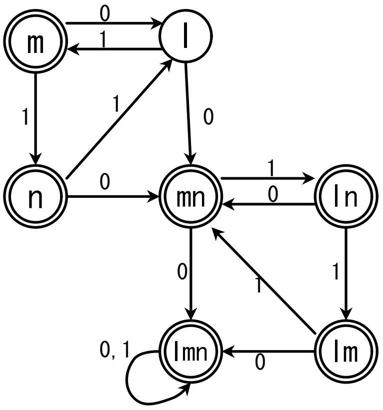
M_3 の状態遷移表

	0	1
l	mn	m
ⓜ	l	n
ⓝ	mn	l

M_3' の状態遷移表

	0	1
l	mn	m
(mn)	lmn	ln
ⓜ	l	n
(lmn)	lmn	lmn
(ln)	mn	lm
ⓝ	mn	l
(lm)	lmn	mn

M_3' の状態遷移図



(2)

(2-1)

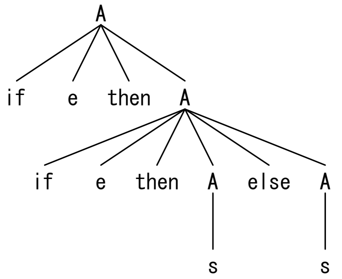
最左導出

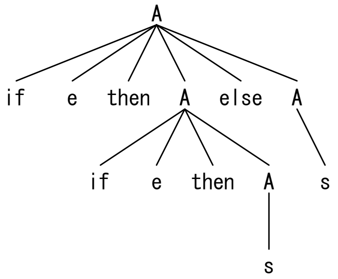
A → if e then A else A
→ if e then s else A
→ if e then s else if e then A
→ if e then s else if e then s

最右導出

A → if e then A else A
→ if e then A else if e then A
→ if e then A else if e then s
→ if e then s else if e then s

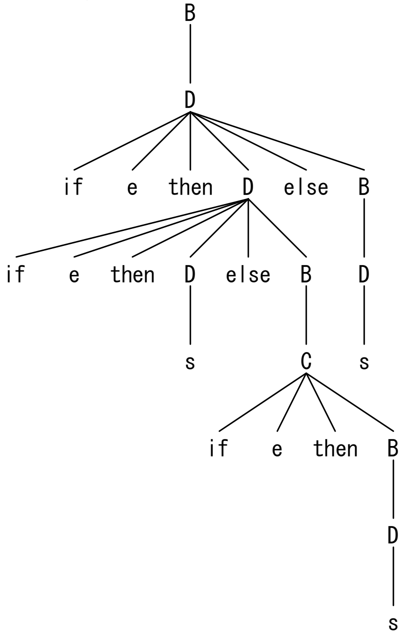
(2-2)

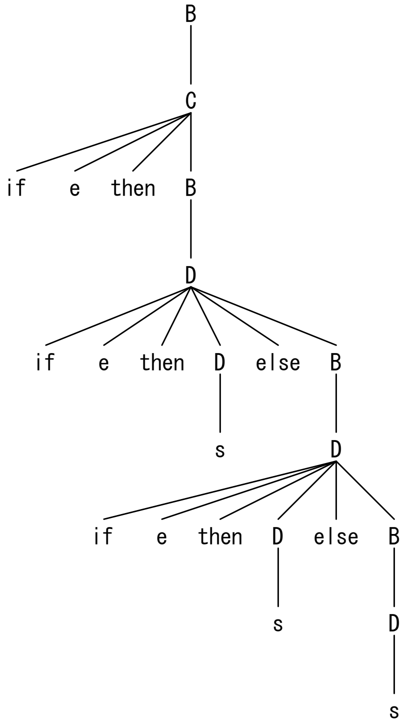




(2-3)

”if e then if e then s else if e then s else s”に対して，以下の二通りの導出木が存在するため．





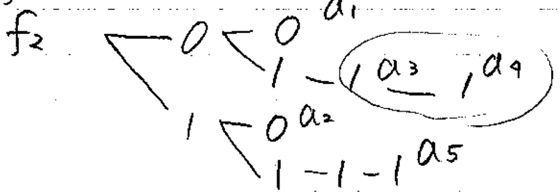
$\boxed{10}$ (1) (i−1番目まで復号完了したとする
i番目からの符号の先頭が、
100 00 100
である場合を考える)

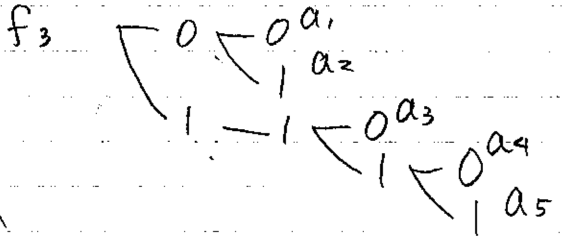
$f_1(a_4)f_1(a_1)f_1(a_4) = \overline{100}\ \overline{00}\ \overline{100}$

$f_1(a_3)f_1(a_1)f_1(a_2)f_1(a_1) = \underbrace{10}\ \underbrace{00}\ \underbrace{01}\ \underbrace{00}$

となり、復号が一意に定まらない

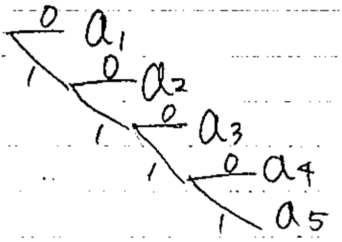
(2) (2−1)





f_3は、先読みせずに瞬時に復号可能なので
f_3による符号化の方が優れている

(2−2)



$f_4(a_1) = 0$
$f_4(a_2) = 10$
$f_4(a_3) = 110$
$f_4(a_4) = 1110$
$f_4(a_5) = 1111$

- 一意に復号可能であることの証明
  - i-1番目までが復号完了したとする
    - i番目が0であるとき
      - i番目はa_0である
    - i番目が1であるとき
      - i+1番目が0であるとき
        - iとi+1番目はa_1である
      - i+1番目が1であるとき
        - i+2番目が0であるとき
          - iからi+2番目はa_2である
        - i+2番目が1であるとき
          - i+3番目が0であるとき
            - iからi+3番目でa_3である
          - i+3番目が1であるとき
            - i+4番目が0であるとき
              - iからi+4番目でa_4である
            - i+4番目が1であるとき
              - iからi+4番目でa_5である
  - これにより、一意に復号化できる

- 平均符号化長が短いことの証明
  - f_2の平均符号化長は、

$$l_2 = \sum_{i=1}^{5} |f_2(a_i)P_i| = 2p_1+2p_2+3p_3+4p_4+4p_5$$

f_3の平均符号化長は

$$\overline{l_3} = \sum_{i=1}^{5} |f_3(a_i)P_i| = 2P_1+2P_2+3P_3+4P_4+4P_5$$

f_4の平均符号化長は

$$\overline{l_4} = \sum_{i=1}^{5} |f_4(a_i)P_i| = P_1+2P_2+3P_3+4P_4+4P_5$$

$$\overline{l_2} - \overline{l_4} = P_1 > 0$$
$$\overline{l_3} - \overline{l_4} = P_1 > 0$$

よってf_2, f_3よりも平均符号化長が短い

(3)(3-1)

(i) $s=2$ のとき

$f_5(a_1)=0$

$f_5(a_2)=1$ よって $Nf_5(1)=2$

(ii) $s \geq 3$ のとき

$Nf_5(s-1) \neq 0$ より

符号語長 $s-1$ の符号が存在する

よって復号木のレベルは $s-1$ 以上である

$Nf_5(1)=0$ と仮定すると

復号木のレベル1の節点は

それぞれ枝を持つ

ハフマンの符号化より、

出る枝が1本しかない中間節点は存在しないので

復号木のレベルは $s-1$ より小さくなる

よって $Nf_5(s-1) \neq 0$ に矛盾する

つまり、$Nf_5(1) \neq 0$ であり、

符号語長1となる符号が最低1つ存在する

また、その符号語長1となる符号以外の符号は

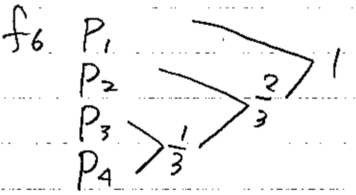
必ず長さ2以上になるので

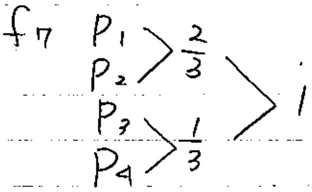
$Nf_5(1)=1$ である

以上より、 $s=2$ のとき $Nf_5(1)=2$

$s \geq 3$ のとき $Nf_5(1)=1$

(3-2) $P_1=\frac{1}{3}$, $P_2=\frac{1}{3}$, $P_3=\frac{1}{6}$, $P_4=\frac{1}{6}$





$f_6(a_1)=0$

$f_6(a_2)=10$

$f_6(a_3)=110$

$f_6(a_4)=111$

$f_7(a_1)=00$

$f_7(a_2)=01$

$f_7(a_3)=10$

$f_7(a_4)=11$

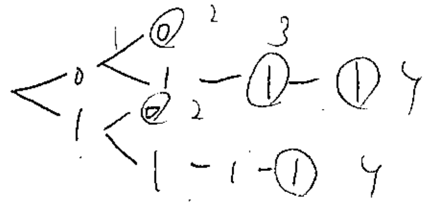
$\boxed{10}$

(1) $f1(a_4)\,f_1(a_4) = f_1(a_3)\,f_1(a_2)\,f_1(a_1)$ より、

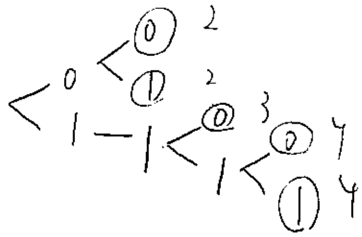
f_1 による符号化は一意ん復号可能ではない

(2)

f_2



f_3



(2-1) f_2 と f_3 んよる符号化は、一意ん復号可能であるという点んおいて優れつかつけがたいが、f_2 は瞬時ん復号不能であるのん対し f_3 は瞬時ん復号可能であるから f_3 による符号化のほうが優れているといえる。

$f_4(a_1)=0$、 $f_4(a_2)=10$ 　 $f_4(a_3)=110$ 　 $f_4(a_4)=1110$ 　 $f_4(a_5)=11111$

$\{a_1, a_2, a_3, a_4, a_5\} \in A$ とすると、任意の $\alpha, \beta \in A^+$ について、$\alpha \neq \beta$ ならば $f(\alpha) \neq f(\beta)$

より、f_4 は一意ん復号可能 —①

$$\sum_{i=1}^{5} p_i f_4(a_i) = p_1 + 2p_2 + 3p_3 + 4p_4 + 4p_5$$

$$\sum_{i=1}^{5} p_i f_3(a_i) = 2p_1 + 2p_2 + 3p_3 + 4p_4 + 4p_5$$

$$\sum_{i=1}^{5} p_i f_2(a_i) = 2p_1 + 2p_2 + 3p_3 + 4p_4 + 4p_5$$

p_1

$) =$

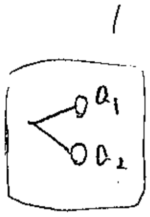
$p_i > 0$ より f_4 による符号化が f_2、f_3 による符号化の平均符号語長よりも短い

(3)

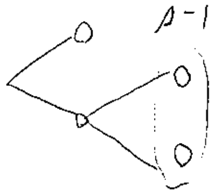
(3-1)

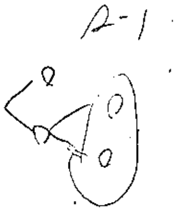
i) $n=2$ のとき

符号木は となり、$N_{f5}(1)=2$



ii) $n>2$ のとき





f5はハフマン符号化の一つであることから、

出る枝が1本しかない中間節点は存在しない。

ゆえに $N_{f5}(n-1)=1$ とはならず $N_{f5}(n-1)=2$

また、$N_{f5}(n-1)=2$ であり、出る枝が1本しかない中間節点は存在しないことから、$1\le r<n-1$ であるrについて、$N_{f5}(r)=1$

ゆえに $N_{f5}(1)=1$ となる。

(3-2)

11

(1)

説 1

I) (a), (e)

(a)：フレーム誤りがあった場合，送信側はどの時点のフレームを再送するのかを識別するため．

(e)：フレーム誤りがあった場合，その時点に戻るために必要．

II) (b), (f)

(b)：フレーム誤りがあった場合，送信側から本来と異なった順番でデータが送信されるため．

(f)：フレーム誤りの際，どの時点のデータを挿入すれば良いのかを識別するため．

説 2

I) (a), (c)

(a)：Stop-and-Wait ARQ 方式では受信側が正しく受信しない限り新たなフレームは送受信されないが，Go-back-N ARQ 方式では先行して送信され，受信側でフレーム誤りを検出した以降のデータフレームを棄却する必要があるため．

(c)：上記の棄却処理の際に，先行送信のフレームと再送されたフレームを識別するため．

II) (b), (d)

(b)：Go-back-N ARQ 方式では，受信側は正しく受信していない最少の番号のフレーム以外は棄却するが，Selective Repeat ARQ 方式では，正しく受信すれば全て採択するため．

(d)：再送を行った後に，Go-back-N ARQ 方式では再送した次のフレームから送信を再開するが，Selective Repeat ARQ 方式では NACK を受信したフレーム以外を送信しないため．

(2)

Stop-and-wait ARQ 方式

送信側がデータフレームを 1 個送信し，対応した ACK フレームの受信を完了するまでにかかる時間は，$((100bit+20bit)/1000bps*1000)+40ms*2 = 200ms$ である．

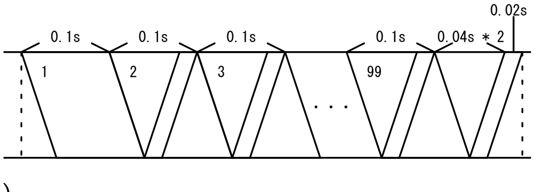
よって，データフレームを 99 個送信した際の時間は，$200ms*99$ であり，送信データ量の合計は $100bit*99$ なので，平均スループットは以下の式で求められる．

$(100bit*99)/(200ms/1000*99) = 500bps$

Go-back-N ARQ 方式

片方向伝搬遅延 $40ms$，ACK フレームの受信時間 $20bit/1000bps*1000 = 20ms$ より，送信側は送信待ちを行わない．よって，下図より，

$(100bit*99)/(0.1s*99+0.1s) = 990bps$



(3)

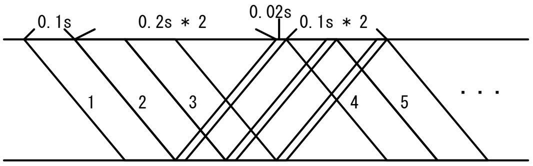
Stop-and-wait ARQ 方式

(2) と同様に，

$(100bit*99)/(0.52s*99) = 192bps$

Go-back-N ARQ 方式

片方向伝搬遅延 $200ms$ より，(2) と異なり下図のような送信待ちを行う．



図より，第 1 フレームの ACK が返るまでの時間が $0.52s$ なので，第 97 フレームの ACK が返るまでの時間は $0.52s*33$ となる．残りの 2 フレームの ACK が帰るまでの待ち時間が $0.2s$ なので，

$(100bit*99)/(0.52s*33+0.2s) = 570bps$

(4)

Stop-and-wait ARQ 方式

1 フレームの送信回数の期待値は，

$\sum_{k=1}^{\infty} k(\frac{1}{10})^{k-1}(\frac{9}{10}) = \frac{10}{9}$ 回[*1]

よって，

$(100bit*99)/(0.2s*99*\frac{10}{9}) = 450bps$

[*1] $S_n = \sum_{k=1}^{n} kp^{k-1}$ は，$S_n - pS_n$ より計算

Go-back-N ARQ 方式

上記より，1 フレームの誤り回数の期待値は $\frac{1}{9}$ 回であり，誤り 1 回あたり 2 フレーム相当の再送，$0.2s$ のロスとなる (最終フレームの場合再送は 1 フレームだがロスする時間は不変) ため，

$(100bit * 99)/((0.1s + 0.2s * \frac{1}{9}) * 99 + 0.1s) = 811bps$

(5)

横軸 : フレーム誤り率
グラフ (上から) : Selective Repeat ARQ
Go-back-N ARQ
Stop-and-Wait ARQ

フレーム誤りが無い場合において，Selective Repeat ARQ 方式と Go-back-N ARQ 方式は全く同等の性能であり，Stop-and-Wait ARQ 方式は他の二つと比較して性能が劣る．フレーム誤りが発生した場合，Selective Repeat ARQ 方式は Go-back-N ARQ 方式よりも性能が優れる．これらの理由より，上記の解答が導き出せる．