電気通信大学 電気通信学部 情報通信工学科

情報通信工学実験 A・B
実験項目 1. 情報通信 — 情報・セキュリティ —
課題 「データ圧縮の理解と実装」 — ハフマン符号の理解と実装 (プログラミング) —

[1]

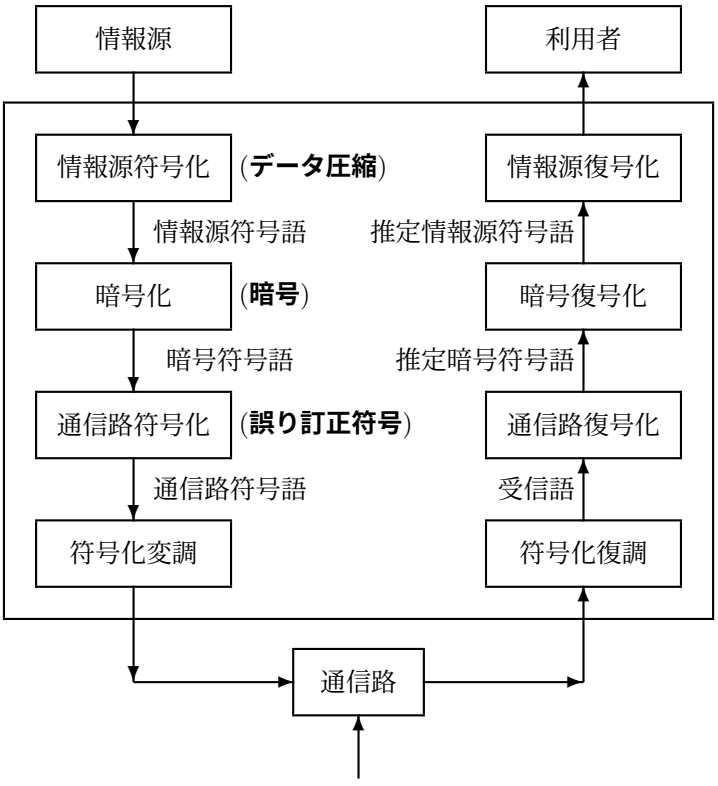
本実験項目「情報・セキュリティ」を履修する学生は、実験担当教員の指示に従って以下の三課題の中から一つの課題を選択し実験を行なう。

本テキストは、課題 **「データ圧縮の理解と実装」— ハフマン符号の理解と実装 (プログラミング)** — に関するテキストである。

1. 「データ圧縮の理解と実装」— ハフマン符号の理解と実装 (プログラミング) —
2. 「暗号化の理解と実装」— RSA 暗号の理解と実装 (プログラミング) —
3. 「誤り訂正符号の理解と実装」— リード・ソロモン符号と最小距離復号の理解と実装 (プログラミング) —

実験項目「情報・セキュリティ」では、下記図に示す 3 種類の**符号化**に関するテーマを扱う。一般に、情報を送信者 (情報源) から受信者 (利用者) に送信する際、次のような 3 種類の符号化を行なう。はじめに、効率性を目的として、**情報源符号化（データ圧縮）**を行なう。次に、安全性を目的として、**暗号化（暗号化）**を行なう。最後に、信頼性を目的として、**通信路符号化（誤り訂正符号）**を行なう。本実験では、いずれかの符号化の具体的手法について理解し、実際に計算機上でプログラムを作成し、実装を行なう。

ディジタル伝送・記憶システム



[1] ©電気通信大学 情報通信工学科 栗原正純 (e-mail: kuri@ice.uec.ac.jp), 2003 – 2009. (2009/3/26/9:05 修正)
本テキストは、http://www.code.ice.uec.ac.jp/kuri/C3/ より入手可能である。
(46: /doc/tex/uec/jikken/2009text/istext2009sourcecoding.tex)

目的 (課題)

1. データ圧縮法 (情報源符号化法) の一つであるハフマン (Huffman) 符号について調べ，符号化と復号化のアルゴリズムを理解する．
2. 次に，ハフマン符号の符号化と復号化のアルゴリズムをプログラミングし，計算機上で実装する．

提出レポートの内容

1. ハフマン符号およびデータ圧縮について調べたことを１ページ以内（A4 サイズ）に簡単にまとめ，記述する．
2. ハフマン符号の符号化・復号化を実装するに当たり，プログラミングで工夫した点を記述する．
3. ハフマン符号（アルゴリズム）の改良点などアイデアがあれば記述する．
4. プログラムのソースを印刷し，プログラムの各行（あるいは各部分）が何を実行する箇所なのかなどの注釈を記入しものをレポートに添付する．
5. 作成したプログラムが正しく動作していることを確認する方法を記述し、その確認結果も示すこと．
6. 次に示す ied のディレクトリに置かれている book1 と book2 というファイル名のファイルを実際に個々が作成したプログラムを用いて圧縮し，その圧縮率を示せ．

   /home/staff/kuri/corpus

   これらのファイルは Calgary Corpus というファイル群の一部のファイルである．これらのファイルはインターネットを通じて得ることができる．たとえば，

   http://corpus.canterbury.ac.nz/resources/calgary.tar.gz

   ただし、tar.gz ファイルのサイズは 1.02 M bytes 程度である。Download する場合は，個々の利用可能なディスク容量に対応できるかなどを各自で判断してから実行すること．分からなければ教員に尋ねて下さい．また、

   http://corpus.canterbury.ac.nz/

   などを参考にせよ．
7. レポートには “参考文献” という節（あるいは項目）を設け，主に参考にした文献を明記する．ここで，文献とは，書籍に限らない．インターネットなどを利用して得られたものも明記すること．その場合，URL を明記する．また，そのページのタイトルがあればそれも明記する．自分のアイデアと他人のアイデアを明確に区別すること．
8. 紙のレポートとは別にソースプログラムをメイルにて担当教員宛に送る．その際，そのファイルのコンパイル方法，実行方法も忘れずにメイルにて送る．

   Subject 欄には半角英数字を用いて “3jikken(name)” と記述し，メイル文章の初めに，氏名と学籍番号を書くこと．ここで、 “name” は各自の氏名のローマ字表記を書くこと。

1 テキストデータ圧縮

1.1 符号化，記号，アルファベット，符号語

情報理論において，データ圧縮は（**情報源**）**符号化**として扱われる．符号化とは，文章や文字列，プログラム，データなど（これらを総称して**メッセージ**ということにする）を “1” と “0”（符号文字という）のビット列によって表すことである．したがって，データ圧縮とは，ビット列の長さ

をできるだけ短くするような効率的な符号化ということができる．メッセージに現れる文字のことを**記号** (**symbol**) といい，文字列のことを**記号列**ということにする．

符号化について説明するため，入力記号列の例として記号列 "$aabbcccddddeeeee$" を考える．この記号列は，16 個の記号から成り，記号列中に出現する記号は，$\{a, b, c, d, e\}$ の５種類である．このように，記号列中に出現しているか，または出現する可能性のある記号の集合のことを（**情報源**）**アルファベット**という．一般に，ここでは，テキスト文章中の記号は，１バイト（８ビット）の ASCII コードによって，記憶されたり，２進表現されていると考える．したがって，前記の記号列は，

$$\underbrace{01100001}_{a}\underbrace{01100001}_{a}\underbrace{01100010}_{b}\underbrace{01100010}_{b}\underbrace{01100011}_{c}\cdots\underbrace{01100101}_{e} \tag{1}$$

によって表されていると考える．ASCII コードの場合，各記号は**符号語** (**code word**) という８ビットのビット列で表されるので，この記号列を表すのに必要なビット長は $8 \times 16 = 128$ ビットになる．各符号語（この場合はすべて８ビット）のビット長のことを符号語長という．

1.2 簡単な圧縮法の例

前節に挙げた記号列の例のように，事前に情報源アルファベットの記号の種類が５種類のみであることが既知であれば，それらの記号は３ビットで表現可能である．例えば，各記号の符号語を次の表のように対応させる．

Symbol	Codeword
a	000
b	001
c	010
d	011
e	100

(2)

すると，先の記号列は，

$$\underbrace{000}_{a}\underbrace{000}_{a}\underbrace{001}_{b}\underbrace{001}_{b}\underbrace{010}_{c}\cdots\underbrace{100}_{e} \tag{3}$$

によって表される．この記号列を表すのに必要なビット長は $3 \times 16 = 48$ ビットになる．

1.3 データ圧縮の評価：圧縮率

ここでは，元の記号列のビット長と圧縮（符号化）されて得られた符号語列のビット長との比として，**圧縮率** (**compression ratio**) を定義する．具体的には，

$$\text{圧縮率} := \frac{\text{符号化された記号列のビット長}}{\text{入力記号列のビット長}} \times 100 \quad (\%) \tag{4}$$

と定義する．したがって，圧縮率の値が小さいほど効率のよい符号化が行なえたことになる．先の例の圧縮率は，$(48/128) \times 100 = 37.5\%$ となる．

2 ハフマン符号

2.1 木による符号の表現

本節では，符号の木について説明する．一般に，符号語の集合を**符号**という．符号の木の例を図 1 に示す．この木において，各線分を枝とよび，枝の両端の点を節点とよぶ．また，木は上から下に伸びるているとし，枝は上の節点から下の節点へ伸びていると考える．さらに，その節点から枝

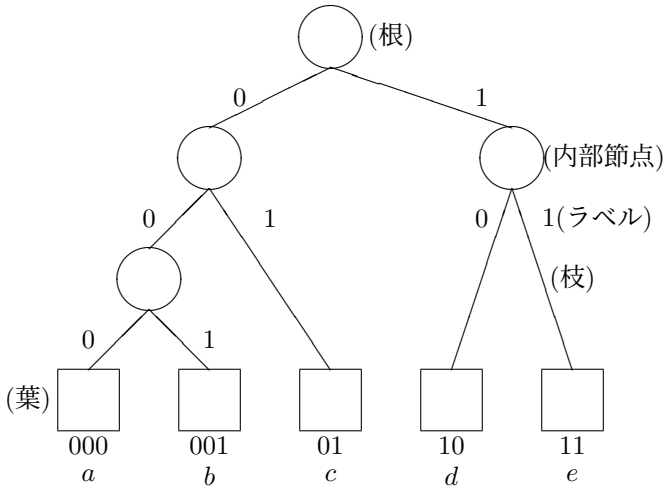


図 1: 符号の木

が一本も下に伸びていない節点（図では□で表示）を葉とよび，葉でない節点（図では○で表示）を内部節点，一番上の節点を根という． 符号の木では，一つの節点から（下に）伸びる枝（2 本ある）には，それぞれラベル０と１が対応づけられている．根から出発して，葉まで枝をたどっていけば、１と０からなる系列が得られ，これがそれぞれの葉に対応した符号語となる．図 1 では葉の下に表示．したがって，葉に情報源アルファベットの１記号を対応させることで、木を用いてすべての記号と符号語との対応を記述することができる．図 1 の符号の木によって定まる符号を表 1 に示す． もし，すべての符号語が葉に対応していれば，この木を利用して符号語の系列を一意的に

記号	符号語
a	000
b	001
c	01
d	10
e	11

表 1: 符号の木から定まる符号

復号することが可能になる．実際，与えられた符号語系列に現れる記号 1 と 0 に従って，根から順に枝をたどり葉に到着した時点で、そこまでの 1 と 0 の系列が一つの符号語であることがわかるので，どの葉に対応している記号，すなわちその符号語に対応する記号を出力する．次の１記号を復号するためには，再び残りの符号語系列に従って根から枝をたどっていけばよいことになる．

2.2 符号の木を使った復号

たとえば、図 1 の符号の木によって、符号語系列 “10001010011110” を復号することを考える．最初の２ビットの “10” まで根から枝をたどるとラベル d の葉に到達する．これから，まず記号 “d” が復号される．次の，残りの符号語系列 “00101001111” に対して再び根から枝をたどれば，最初の３ビットの “001” をたどったところでラベル b の葉に到達する．これから２番目の記号 “b” が復号される．同様の操作を繰り返すことで，与えられた符号語系列は，

$$\underbrace{10}_{d}\,\underbrace{001}_{b}\,\underbrace{01}_{c}\,\underbrace{001}_{b}\,\underbrace{11}_{e}\,\underbrace{10}_{d} \tag{5}$$

のように復号される．

2.3 ハフマン木の構成アルゴリズム

初期条件として,「情報源アルファベット」と「入力記号列の中の記号の生起確率または出現頻度」を既知とする.

1. 各記号に対する葉を作る．それぞれの葉には，その記号の生起確率または出現頻度を書いておく.
2. 生起確率（または出現頻度）のもっとも小さい二つの葉に対し，新しい節点を一つ作り，その節点と 2 枚の葉を枝で結ぶ.

   この二つの枝の一方には 0 を，他方には 1 のラベルをつける.

   さらに，この新しい節点に，2 枚の葉の確率（または頻度）の和を書き，この節点を新たな葉と考える．すなわち，この節点から伸びている枝を取り除いたと考える.
3. 葉が 1 枚になるまで，上記 2 の手順を繰り返す.
4. 根から記号に対応する葉まで枝をたどったときに得られる 1 と 0 の系列をその記号に対する符号語とする.

上記のようにして構成される符号の木をハフマン木という.

2.4 ハフマン符号化の例

先に例として取り挙げた記号列 "*aabbcccddddeeeee*" をハフマン符号化しよう．まず，記号列中の記号の出現頻度と生起確率を求めると表 2 となる.

記号	出現頻度	生起確率
a	2	2/16
b	2	2/16
c	3	3/16
d	4	4/16
e	5	5/16

表 2: 記号列 *aabbcccddddeeeee* 中の各記号の出現頻度と生起確率

初めに，各記号に対応する葉を作る．このとき，葉の中の数字は生起確率を表す（図 2）.

もっとも確率の小さい葉は，記号 "a" と "b" に対応する葉であるから，新しい節点 A を作り，葉 a と b を A に結び，この二つの枝にラベル "0" と "1" を与える．また，節点 A の生起確率として $2/16 + 2/16 = 4/16$ を与える（図 3）.

新たに付け加えた節点 A を葉と考え，葉の集合 $\{A, c, d, e\}$ に対して同様の操作を行なう．この場合，もっとも確率の小さい葉は，記号 "A" と "c" に対応する葉であるから，これらを新しい節点 B に結ぶ.（図 4）.

葉の集合 $\{B, d, e\}$ に対して同様の操作を行なう．この場合，もっとも確率の小さい葉は，記号 "d" と "e" に対応する葉であるから，これらを新しい節点 C に結ぶ.（図 5）.

葉の集合 $\{B, C\}$ に対して同様の操作を行なう．この場合，もっとも確率の小さい葉は，記号 "B" と "C" に対応する葉であるから，これらを新しい節点 D に結び，ハフマン木は完成する（図 6）.

最終的に得られたハフマン木において，根から葉までたどったときに得られる枝のラベル系列が，その葉のラベルに対応する符号語になっており，得られた符号語を表 3 に示す.

得られた符号語を用いて記号列 *aabbcccddddeeeee* を符号化すると

$$\underbrace{000}_{a}\ \underbrace{000}_{a}\ \underbrace{001}_{b}\ \underbrace{001}_{b}\ \underbrace{01}_{c}\ \cdots\ \underbrace{11}_{e} \tag{6}$$

となる．この記号列を表すのに必要なビット長は $3 \times 4 + 2 \times 12 = 36$ ビットになる．したがって，圧縮率は，$(36/128) \times 100 = 28.1\%$ となる.

記号	符号語
a	000
b	001
c	01
d	10
e	11

表 3: ハフマン木から定まる符号

参考文献

[1] 植松友彦, 文章データ圧縮アルゴリズム入門，CQ 出版，1994.

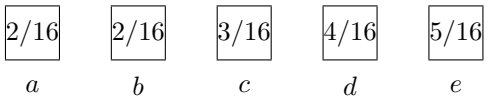


図 2: ハフマン木の構成過程 1

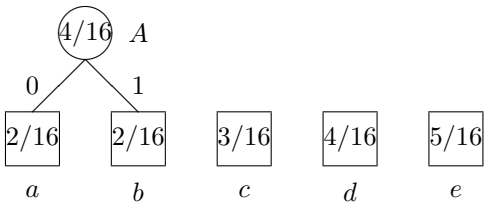


図 3: ハフマン木の構成過程 2

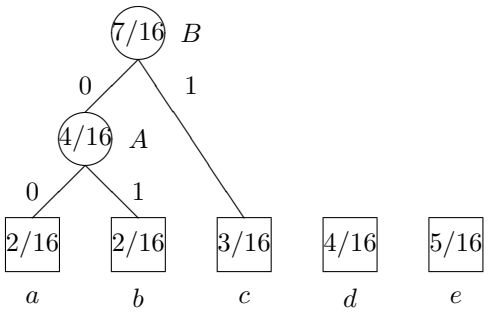


図 4: ハフマン木の構成過程 3

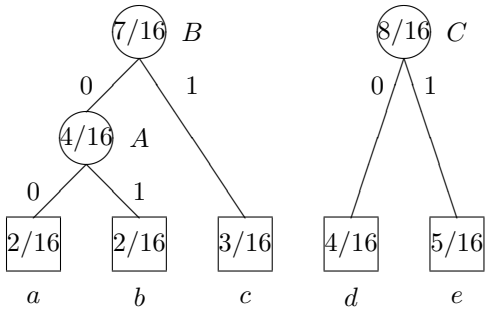


図 5: ハフマン木の構成過程 4

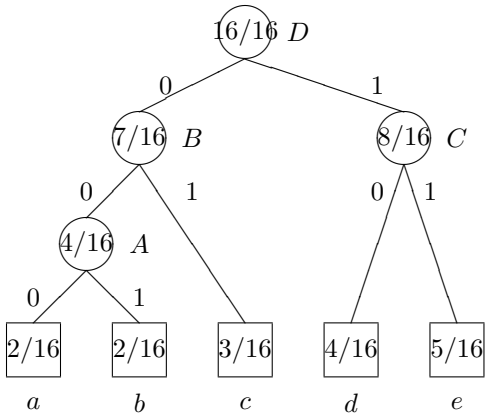


図 6: ハフマン木の構成過程 5（完成）

ASCHII

ASCII(アスキー) とは、American Standard Code for Information Interchange(情報交換用米国標準符号) の略。その中身は、1963 年にアメリカ規格協会 (ANSI: American National Standard Institute) が定めた、7/8 ビット英数字のコード体系の 1 つ。7 ビット版は、128 種類のローマ字、数字、記号、制御コードで構成されている。実際にはコンピュータは 1 文字を 8 ビット (1 バイト) で表現するため、256 種類の文字を扱うことができるが、ASCII が定めていない 128 文字分の拡張領域には、コンピュータメーカーや国によって異なる文字が収録されている。日本では、拡張領域にカナ文字を収録したコード体系が JIS X 0201 として規格化されている。

下記に、7 ビットコードを上位 3 ビットと下位 4 ビットに分けて 16 進表示したものを示す。たとえば、大文字の「Z」は、16 進表示では 5A、2 進表示では 1011010 となる。

ASCHII

下位 \ 上位	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	‘	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	”	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAC	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	’	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF/NL	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

[2] ©電気通信大学 情報通信工学科 栗原正純 (e-mail: kuri@ice.uec.ac.jp), 2005.(2005/05/10)
本資料は参考文献欄に挙げた文献をもとに作成したものである。

制御符号の説明

制御符号	コード	制御符号名	意味
NUL	00	Null	空
SOH	01	Start of Heading	ヘディング開始
STX	02	Start of Text	テキスト開始
ETX	03	End of Text	テキスト終了
EOT	04	End of Transmission	伝送終了
ENQ	05	Enquiry	問い合わせ
ACK	06	Acknowledge	肯定応答
BEL	07	Bell	ベル
BS	08	Backspace	バックスペース（1 文字後退する）
HT	09	Horizontal Tablation	水平タブ
LF/NL	0A	Line Feed/New Line	改行／復行（復帰・改行）
VT	0B	Vertical Tablation	垂直タブ
FF	0C	Form Feed	改頁
CR	0D	Carriage Return	復帰
SO	0E	Shift Out	シフト・アウト
SI	0F	Shift In	シフト・イン
DLE	10	Data Link Escape	データ・リンクでの拡張
DC1	11	Device Control 1(X-ON)	装置制御 1（送信を開始する要求に使用）
DC2	12	Device Control 2	装置制御 2
DC3	13	Device Control 3(X-OFF)	装置制御 3（送信を止める要求に使用）
DC4	14	Device Control 4	装置制御 4
NAC	15	Negative Acknowledge	否定応答
SYN	16	Synchronous Idle	同期文字
ETB	17	End of Transmission Block	伝送ブロック終了
CAN	18	Cancel	取り消し
EM	19	End of Medium	媒体終端
SUB	1A	Substitute Character	（CP/M でファイルのデータ終了記号に使用している）
ESC	1B	Escape	拡張（画面やグラフィックなどの制御コードの拡張に使用している）
FS	1C	File Separator	ファイル・セパレイタ
GS	1D	Group Separator	グループ・セパレイタ
RS	1E	Record Separator	レコード・セパレイタ
US	1F	Unit Separator	ユニット・セパレイタ
SP	20	Space	空白、ブランク、スペース
DEL	7F	Delete	抹消

参考文献

[1] http://e-words.jp/w/ASCII.html

[2] http://www.psl.ne.jp/perl/pdojo00b.html

[3] http://www.komonet.ne.jp/~perl/chap13.htm

[4] 椋田 (むくだ) 實, はじめての C (改訂第三版), 技術評論社, 1995(平成 7 年).

[5] 平林雅英，ANSI C 言語辞典 (初版 第 10 刷発行), 技術評論社, 2000(平成 12 年).