数字向きバイト単位コードを用いた連接数字圧縮法†

坪田信孝†† 奥田久徳††

通常用いているデータファイルには，高頻度に数字および空白が出現し，かつこれらは連接して出現する頻度が高い．この点に注目して，連接数字圧縮法を考案した．この方式は，連接した2個の数字または空白を1個のJISコード文字におきかえることを基本とし，ランレングスコードも考慮され良好な圧縮効果を得ることができる．また，数字向きであるがバイト単位に構成されたコードを用いている．したがってBCDコードなどとは異なり，他の文字が混在したデータに対しても応用でき，さらに通信プロトコルに依存しない通信が可能である．本論文では，連接数字圧縮法の詳細を述べるとともに，実現例を示し，この方式が容易に実現できかつ広く応用可能であることを示す．

1. はじめに

パーソナル・コンピュータなどの小型コンピュータが普及し，コンピュータ間通信の需要は急速に高まってきた．しかしながら，コンピュータの機能に対し最適の通信回線を確保することはしばしば困難であり，回線速度がシステム全体の処理効率の決定要因となりやすい．したがって，より高速の回線を考える一方で，データ圧縮などによる効率のよい回線利用法も考えられるべきである．

最短のデータ圧縮はShannonらの情報理論によって理論づけられている[1]．これは「より高頻度に出現する情報要素および情報要素の組合せに，より短い伝送コードを当てはめる」ことを原則とするものである．すなわち，理論的に最短のデータ圧縮の実現のためには，情報要素およびその組合せの出現頻度を前もって知っておくことが必要である．このことは，コンピュータ間データ伝送を考えた場合，特殊な例を除き非現実的である．したがって現状では7または8ビットで定義されたJISコード文字などによるデータ伝送が広く用いられている．しかしながら，JISコード文字のなかでも空白および数字は他の文字に比べ高頻度に出現する．この点に注目し，空白および数字に短いコードを与えれば効果的なデータ圧縮が可能である．

従来から，空白に関しては，空白がとくにプリンタイメージファイルのなかで高頻度にかつ連続して出現する性質を用いたランレングスコードによる圧縮法が用いられている．本論文で述べる方式は，空白圧縮に数字列の圧縮を加えたもので適用データの範囲が広く，圧縮効果も大きい．

以下では，この方式の詳細を述べるとともに，この方式の特徴と実際のコンピュータ間通信における実現例について述べる．

2. 連接数字圧縮法

数字はコンピュータ処理の入力となるデータファイルのほか，中間ファイル，プリンタイメージファイルなど多くのファイルのなかで高頻度に出現する．多くのキーボード入力装置が数字専用キーをもっている事実は，コンピュータ処理に用いられる文字のなかで数字が他の文字より高頻度に出現すること，およびそれゆえ，特別の取扱いをすることに利点があることを実証している．

数字のみからなるデータ伝送では，JISの文字コードより短いコードとしてBCDコード，バイナリコードを使用できるが，数字以外の文字が混在するデータの場合には使用しにくい．また，通信制御文字を用いたバイトオリエンテッドな通信手順のもとでは使用しにくい．さらにここで，数字は数値としての大きさを表現するという特性のほかに，文字としての特性を合わせもつことに注意しなければならない．後者の特性をもつコードは前者の特性を包含するが，前者の特性のみをもつコードは書式情報を付加して初めて後者の特性をもつ．以下では前者の特性のみを表す場合に数値といい，後者の特性を含めて表す場合に数字と呼ぶ．BCDコード，バイナリコードは数値を伝達するが，数字を伝達するものではない．したがって，新しい数字伝達のためのコード化に意義があるといえる．

数字データの新しいコード化を考える際に，数字は高頻度に出現するだけでなく，出現する場合にはしば

† Conjunctive Number Compression with Number Oriented Byte Unit Code by NOBUTAKA TSUBOTA and HISANORI OKUDA (Department of Hygiene, School of Medicine, Hiroshima University).

†† 広島大学医学部衛生学教室

しば連接して出現するという性質を用いる．以下では連続という言葉は同一文字の連続（ラン）を想起させやすいので，異なる文字の連なりを含めた意味で連接という．さらに0～9の数字に空白を加えた11種の文字を考えると，連接して出現する確率はきわめて高くなる．

11種の文字に短いコードを個々に与える方式では，他の文字をも含むデータの伝送に際し，非等長コード系となり，従来の文字コードを主体とした伝送手順の下での応用が困難である．そこで，上に述べたように11種の文字が高い確率で連接する性質を導入し，連接した数字または空白を圧縮することを考えた．この点から，本論文で述べるデータ圧縮法を連接数字圧縮法 (Conjunctive Number Compression, 略して CNC) と呼ぶ．

連接して出現するという性質から，11種の文字の2種の組合せ（11×11=121 通り）それぞれに1個の伝送コードを与えることの意義が説明される．奇数個の連接に対しては，最後の1文字には別のコードを与えなければならない．したがって132種のコードですべての連接した数字および空白を表現できる．これは8ビットJISコード文字から機能キャラクタを除く部分に十分割当てできる．さらに余裕があるので，数字と連接して出現する頻度の高い小数点と正負荷号にそれぞれ8ビットのコードを与える．残りは，同一文字の連続数を表現するコード（ランレングスコード）として用いる．ランレングスコードはこれに続くJISコード文字がランレングスコードで示される数だけ続くことを意味する．ただし，長さ4以下のランはランレングスコードを用いる意義がない（3章で後述）ので長さ5以上のランを対象とする．

以上の視点から考案したコード表を**表1**に示した．このコードはBCDコードと同等の符号化効率をもつだけでなく，数字のもつ文字としての特性をも伝達する．また，バイト単位で表現されているため多くの通信プロトコルの下で実用化できる．この特徴から，表1のコードを数字向きバイト単位コード (Number Oriented Byte Unit Code, 略して NOBU Code) と呼ぶ．表1のコード表は JIS C 6220 のローマ文字用図形キャラクタ集合（G 0 集合）と片仮名用図形キャラクタ集合（G 1 集合）に，上述の132種のコードおよび小数点，正負符号を割り当て，残りにランレングスコードを割り当てたものである．JIS の未定義部分 'E0～'FE および 'FF（以下，' は16進数を表現する）はシステムによっては無視するものもあるので使用してもしなくてもよいこととする．この部分はランレングスコードの拡張部分であるため，デコード手順（4章に後述）は 'E0～'FF の使用のいかんにかかわらず

表1 数字向きバイト単位コード (NOBU コード) と JIS コード
Table 1 Number oriented byte unit code (NOBU code) and JIS code.

	1	2	3	4	5	6	7	A	B	C	D	E	F
0		␣	0 ␣4	@ 00	P 16	‘ 32	p 48	n.d.	ー 78	タ 94	ミ '12	'28	'44
1	DC 1	! 0	1 ␣5	A 01	Q 17	a 33	q 49	。 63	ア 79	チ 95	ム '13	'29	'45
2	DC 2	" 1	2 ␣6	B 02	R 18	b 34	r 50	「 64	イ 80	ツ 96	メ '14	'30	'46
3		# 2	3 ␣7	C 03	S 19	c 35	s 51	」 65	ウ 81	テ 97	モ '15	'31	'47
4		$ 3	4 ␣8	D 04	T 20	d 36	t 52	、 66	エ 82	ト 98	ヤ '16	'32	'48
5		% 4	5 ␣9	E 05	U 21	e 37	u 53	・ 67	オ 83	ナ 99	ユ '17	'33	'49
6		& 5	6 0␣	F 06	V 22	f 38	v 54	ヲ 68	カ 84	ニ .	ヨ '18	'34	'50
7		' 6	7 1␣	G 07	W 23	g 39	w 55	ァ 69	キ 85	ヌ -	ラ '19	'35	'51
8		(7	8 2␣	H 08	X 24	h 40	x 56	ィ 70	ク 86	ネ +	リ '20	'36	'52
9		) 8	9 3␣	I 09	Y 25	i 41	y 57	ゥ 71	ケ 87	ノ '5	ル '21	'37	'53
A		* 9	: 4␣	J 10	Z 26	j 42	z 58	ェ 72	コ 88	ハ '6	レ '22	'38	'54
B		+ ␣␣	; 5␣	K 11	[27	k 43	{ 59	ォ 73	サ 89	ヒ '7	ロ '23	'39	'55
C		, ␣0	< 6␣	L 12	¥ 28	l 44	¦ 60	ャ 74	シ 90	フ '8	ワ '24	'40	'56
D		- ␣1	= 7␣	M 13	] 29	m 45	} 61	ュ 75	ス 91	ヘ '9	ン '25	'41	'57
E		. ␣2	> 8␣	N 14	^ 30	n 46	~ 62	ョ 76	セ 92	ホ '10	゛ '26	'42	'58
F		/ ␣3	? 9␣	O 15	_ 31	o 47	n.d.	ッ 77	ソ 93	マ '11	° '27	'43	'59

n.d.: 未定義，␣: 空白
DC 1 と DC 2 は制御文字として用いる．DC 1 はこれに続く文字が NOBU コードであることを示し，DC 2 は JIS コードであることを示す．
(') の付いた数字はランレングスコードで，これに続く JIS コード文字のランの長さを示す．
n.d.: not defined, ␣: space.
DC 1 and DC 2 are used as control characters.
DC 1 indicates the following characters coded with NOBU Code table.
DC 2 indicates the following characters coded with JIS Code table.
Numbers with (') are used as run length codes, and the number means the run length of the following character defined by JIS Code table.

同一でよい.

CNC は，原データに含まれる連接した数字または空白を NOBU コードに変換することによって圧縮効果を得る圧縮法である．エンコーダは原データの連接した数字または空白および連接のなかに含まれる（数字または空白に接する）小数点，正負符号を NOBU コードに変換し，他の文字は JIS コードをそのまま用いる．したがって，JIS コード状態から NOBU コード状態への移動とその逆方向への移動をデコーダに伝達する制御文字（以下，切換制御文字）が必要である．表1ではこれに DC1('11) および DC2('12) を用いている．この点は著者らのシステムで好都合であったためにすぎない．切換制御文字に何を用いるかは，利用システムの特性に合わせて決定すればよい．ただし，エンコーダ，デコーダの独立性を高めるためには，それぞれ切換制御文字を容易に変更できることが望ましい（6章に後述）.

他の制御文字は NOBU コード状態であっても JIS と同一の意味をもつものとする．すなわち，CR や LF はコードの状態に関係なく使用できることとする．切換制御文字を本来の JIS コードとしての意味をもたせて伝達することは，表1の例で述べると，NOBU コード状態で DC1 を JIS コード状態で DC2 を伝達することにしておけば可能である.

なお，CNC の基底状態は JIS コード状態としておく．これによって，データは CNC を行っていない場合でもデコーダを正しく通過する（ただし JIS から NOBU への切換制御文字を除く）.

NOBU コードは上述したように JIS コード文字との対応関係で定義したものである．したがって JIS 以外のコードを用いるシステムでは，JIS コードとそのシステムのコードとの変換テーブルによって NOBU コードも変換されるものとする．EBCDIC コードを用いるシステムではとくにこの点に注意する必要がある.

3. エンコーダと圧縮率

エンコーダを設計する際には，圧縮が逆効果にならないよう多少の注意が必要である．以下では，原データを場合分けして，圧縮効果を得るための条件について述べる.

1)　ランのない数字または空白の連接

連接の長さを N とすると，圧縮が逆効果にならないための条件は，

① N が偶数のとき

$$2+\frac{N}{2}\leqq N \rightarrow N\geqq 4$$

② N が奇数のとき

$$2+\frac{N+1}{2}\leqq N \rightarrow N\geqq 5$$

となる．左辺が NOBU コード化したときの長さである．また，2は切換制御文字を使用することによる.

2)　ランのみで構成される場合

これは，通常文字のランにも適用可能で，ランの長さを N_R とすると，圧縮が逆効果にならないための条件は，

$$2+2\leqq N_R \rightarrow N_R\geqq 4$$

となる.

3)　連接の中に含まれるラン

この場合，切換制御文字は不用として，長さ N_R のランをランレングスコードを用いた場合と用いない場合とを比較し，ランレングスコードを用いたほうが他の NOBU コードに比べ同等以上の圧縮となる条件は，

① N_R が偶数のとき

$$2\leqq\frac{N_R}{2} \rightarrow N_R\geqq 4$$

② N_R が奇数のとき

$$2\leqq\frac{N_R+1}{2} \rightarrow N_R\geqq 3$$

となる.

4)　連接のなかに小数点または正負符号を含む場合

小数点または正負符号によって区切られる部分（以下，小連接）の長さを n として NOBU コードと JIS コードと比較すると，

① n が偶数のとき

$$\frac{n}{2}\leqq n \rightarrow n\geqq 0$$

② n が奇数のとき

$$\frac{n+1}{2}\leqq n \rightarrow n\geqq 1$$

となる．切換制御文字2個を考えると，連接のなかの小連接の長さ n がすべて1以下，もしくは $n=2$ または3の小連接が1個のみの場合は，圧縮が逆効果となる.

以上をまとめると，

a．1)～3)より長さ5以上の連接をエンコードの対象とする．ただし長さ5の場合は，これがランであったときのみ圧縮効果がある，

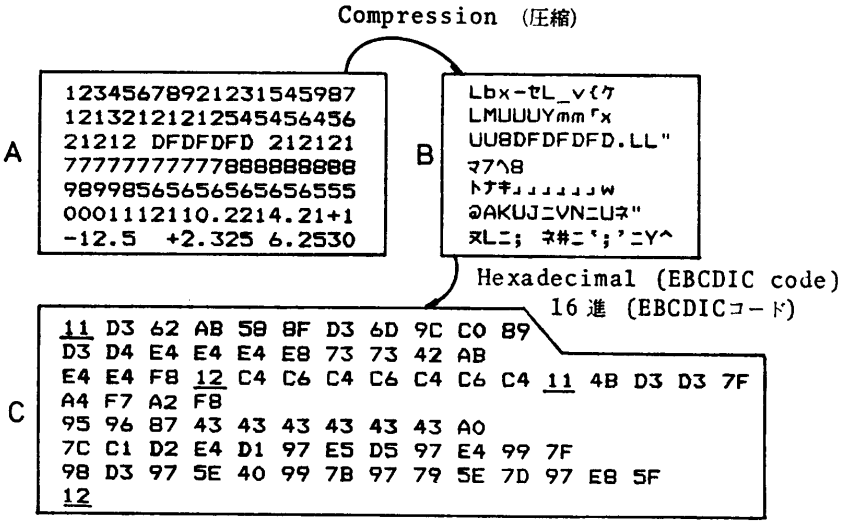

図 1 連接数字圧縮実行例
下線部は DC 1, DC 2. 圧縮率: 55.7%.

Fig. 1 Sample run of conjunctive number compression.
Underlined codes are DC 1 and DC 2.
Compression ratio: 55.7%.

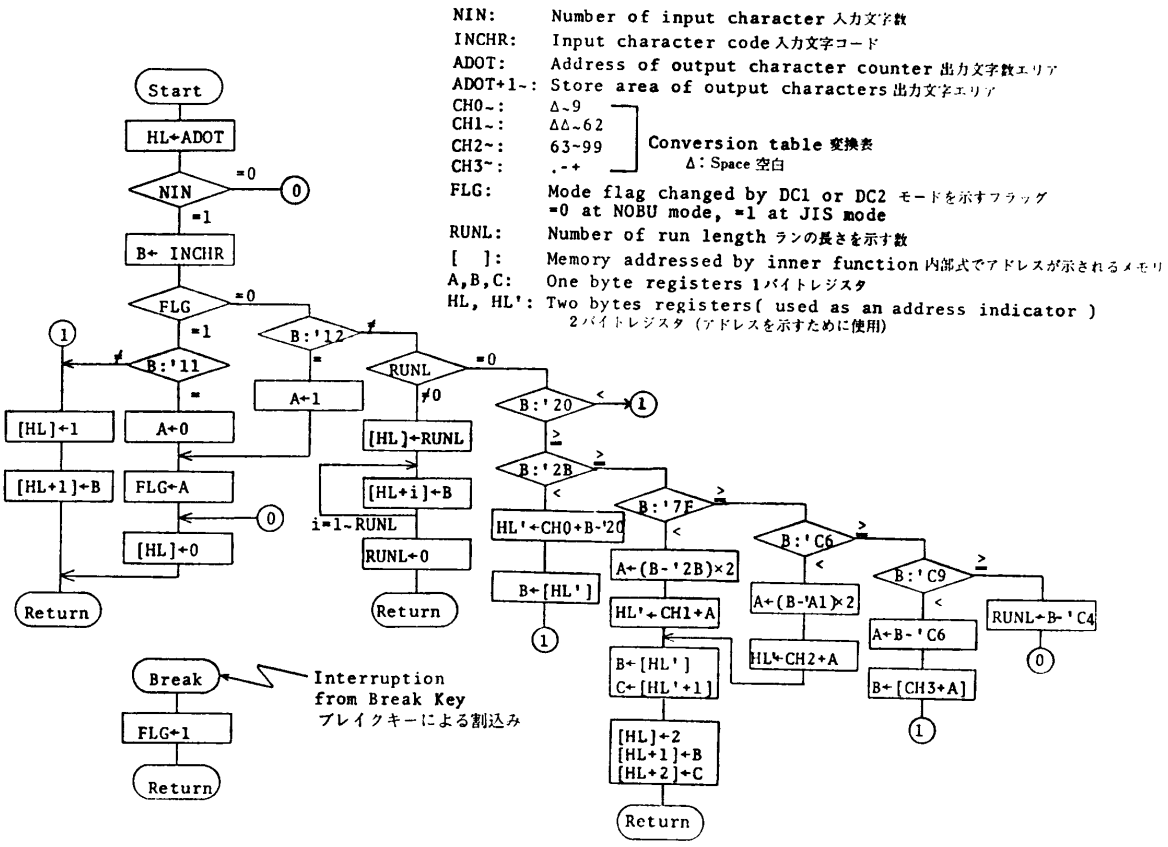

図 2 デコーダのアルゴリズム
Fig. 2 Algorism of decoder.

b．2)，3)よりランは長さ5以上をランレングスコードの対象とする，

に注意してエンコーダを設計すればよい．4)の条件はチェックしなくても，圧縮が逆効果となる場合はきわめてまれである．

実際のエンコーダ設計にあたっては，エンコーダの複雑化にみあった効果が得られるか否かを，対象とする原データおよびエンコーダの開発方法などによって検討する必要がある．エンコーダは圧縮効果からみて最適でなくても，デコーダの方式は左右されないので，実用上は処理速度との関係からエンコード時のチェック部分を少なくしてもかまわない．たとえば上述aのチェックを無視し，1レコードのエンコードのたびに逆効果の発生をチェックし，逆効果のない場合のみエンコード結果を用い，他は原レコードを用いるという方式であっても実用的効果を得ることができる(以下で述べるエンコーダはこの方式を用いた)．

図1は著者らが作成したエンコーダをテストしたものである．Aはエンコード前の原データ，Bはエンコード後のデータをJISコード文字で示したもの(DC 1, DC 2は除かれている)，Cはこれを16進表示したものである．この例の圧縮率は約55.7%である．

4. デコーダ

デコーダのアルゴリズムを**図2**に示した．この例のようにデコーダは入力文字を1文字ずつ順次チェックしていく方式で機能し，複数回のスキャンは不要である．したがって1文字受信後，即座にデコードしてCRTやプリンタなどに出力可能である．FLG=0のときは'12以外のすべてのコード，FLG=1のときは'11以外のすべてのコードを受信できるため，2章で述べたようにNOBUコード状態ではDC 1('11)を，JISコード状態ではDC 2('12)を本来のJISコード文字として受信する．また，ランの長さRUNLは受信文字チェックの最後でチェックされ減算(RUNL←B-'C 4)で求めるためエンコード時に'E0～'FFを使用しているか否かに左右されない．

5. 実現例について

実現例の概略を**図3**に示した．HOSTコンピュータ(HITAC M-200 H，広島大学総合情報処理センター)上に，ファイルを一括して圧縮するプログラム(以下，FCOMP)および一括して圧縮を開くプログラム(以下，FDECOMP)をもつ．ともに，HITAC

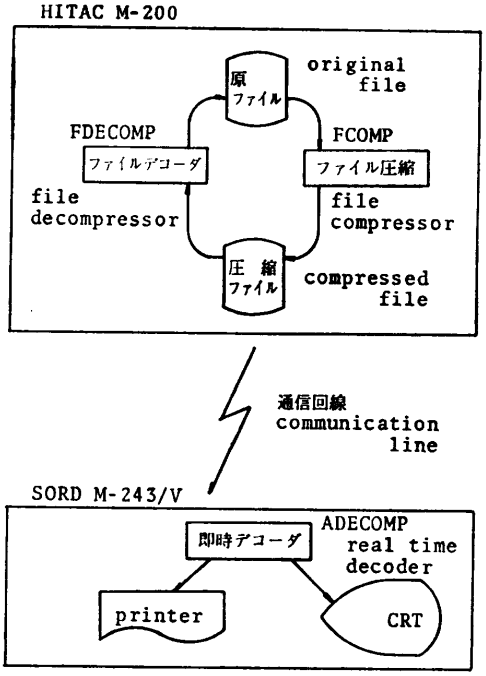

図3 実現例
Fig. 3 Test system.

```
CFile Compression
      CHARACTER*1 A00,DCHR(256),DCHR(256),SNUM,SCHAR
      COMMON ISN
      DATA A00,SNUM,SCHAR/Z00,Z11,Z12/
        :
        :
        :
 1000 CONTINUE
      DO 444 I00=1,256
  444 DCHR(I00)=A00
      READ(1,100,END=222,ERR=333) (DCHR(I),I=1,IRL0)
  100 FORMAT(255A1,A1)
  333 CONTINUE
      DO 445 I01=1,256                          A
      IF (DCHR(I01).EQ.A00) GOTO 456
  445 CONTINUE
  456 ILR=I01-1
      IF (ISP.EQ.1) THEN
      DO 446 I01=1,ILR                          B
      I00=ILR-I01+1
      IF(DCHR(I00).NE.' ') GOTO 447
  446 CONTINUE
  447 ILR=I00
      END IF
        :
        :
        :
        :
```

図4 FORTRAN 77による文字列入力の工夫
A：可変長レコードの入力，B：右方の空白の削除
Fig. 4 Contrivance at input of characters with FORTRAN 77.
A : Input of variable length record.
B : Elimination of right spaces.

表 2 ホストコンピュータ上での処理時間
Table 2 Execution time with host computer.

Data	Program	CPU-time (sec)	Elapse-time (sec)	Compression ratio
R. DATA	FCOMP	4.615	35.13	0.500
	FTEST	1.946	12.74	
	FDECOMP	4.675	9.79	
B. LIST	FCOMP	2.979	19.53	0.587
	FTEST	2.559	16.65	
	FDECOMP	2.246	6.16	

R. DATA: 乱数を書き込んだデータファイル
Data file consisted of random numbers.
B. LIST: 統計パッケージ (BMDP) の出力
Output from the statistical package (BMDP).
FCOMP: 圧縮プログラム
File compression program.
FTEST: 右方の空白除去のみを行って入出力するテスト用プログラム
Test program (read and write only eliminating right spaces).
FDECOMP: 圧縮をデコードするプログラム
File decompression program.

FORTRAN 77[2] で開発した. 図4のプログラム部分Aは, 入力時のエラーを用いてレコード形式 (固定長か可変長か) にかかわらず入力を可能とする例で, FCOMP の出力ファイルは可変長形式にできディスクの使用効率も圧縮率に伴って高くできる. また, 図4のプログラム部分Bは, 原データの各レコードの右方に不用な空白があるとき, これを削除する機能をもたせる例であり, 原データファイルが固定長形式, とくにプリンタイメージファイルのときに出力量を減少させる効果をもつ.

FORTRAN 77 のソース形式で FCOMP は 249 レコード, FDECOMP は 126 レコードとなった. 使用システムは EBCDIC コードを用いており, JIS コードのシステムではより短いプログラムとなる. FCOMP および FDECOMP の処理時間を実行テストし, 表2に示した. 数字データのサンプルとして4万文字の乱数 (以下, R. DATA) およびプリンタイメージファイルのサンプルとして統計パッケージ BMDP の出力例 (以下, B. LIST) を用いた. 比較のため, 入力レコードを圧縮することなく右方の不用な空白のみを削除して出力するプログラム (以下, FTEST) を作成し, この処理時間を併記した. なお処理は TSS 環境で行い, 処理時間が他の利用者との関係で変動するので表2の時間は4回の処理(FCOMP と FTEST は交互に実行) の平均値で示した. また, 圧縮率は入力レコードの右方の空白を除いた文字数で求めた (これを含めると圧縮率はきわめてよくなる). B. LIST は空白のランによる圧縮効果が大きく, 次いで数字列の NOBU コード化による圧縮効果が期待でき, かつ通常文字も多数含まれるファイルである. したがって, 数字列が全体をしめランを期待できない R. DATA のほうが当然ながら長い処理時間を要している. 圧縮による文字数減少に必要な処理時間は,

$$\frac{\text{FCOMP の処理時間} - \text{FTEST の処理時間}}{\text{圧縮前文字数} - \text{圧縮後文字数}} \text{(秒/文字)}$$

と考えると, R. DATA では 11.2×10^{-4} 秒/文字であり, 893.3 文字/秒 (=7.1 kbit/sec) である. B. LIST では 2.5×10^{-4} 秒/文字であり, 3,926.2 文字/秒 (=31.4 kbit/sec) である.

端末装置としてパーソナルコンピュータ (ソード電算機製, M 243/V) を用いた. 4章で述べたデコードルーチンをアセンブラで開発し (以下, ADECOMP), BASIC インタプリタにリンクした[3),4)]. 次いで, この BASIC 言語により端末装置としての機能とデコード機能をもったプログラムを開発した[5)]. ADECOMP は 453 バイトとなった. ADECOMP のみの処理速度は約3万入力文字/秒 (240 kbit/sec) であったが, インタプリタが ADECOMP へ処理を渡すための時間が長いため, 全体としては約 286 入力文字/秒 (=2.29 kbit/sec) であった. この通信システムは低速回線(300 bit/sec) 用に準備したものであり, この程度の回線速度ではデコード時間は通信全体の処理には何ら悪影響を及ぼさない. このシステムで前述の FCOMP および FTEST を用い出力ファイルを端末に割り当ててデータ転送のテストを行った. 転送速度の改善率を,

$$\frac{\text{FCOMPによる転送処理時間}}{\text{FTEST による転送処理時間}}$$

とすると, B. LIST で 82.65%, R. DATA で 84.33 %であった (処理時間は4回交互に実行したときの平均値). この改善率が圧縮率に比べ悪いのは, 用いた通信手順が7ビットコードを用いており, FCOMP のときに FTEST では出現しないシフトコード(SI, SO) が出現するためであり, 8ビットコードの通信手順下ではさらに改善が期待される.

実際の運用にあたっては, HOST コンピュータ上に入出力ファイルの割当て等を行うコマンドプロシージャを用意し, 通常の TSS コマンドと同様の使用法で運用可能となっている.

6. 考 察

本報告の目的は, 数字および空白が高頻度にかつ互いに連接して出現する性質に注目し, この 11 種の文

字の2個以下の組合せに1バイトのコードを対応させることを基本とする，実用的効果のあるデータ圧縮法を提案することである．高頻度出現要素に短いコードを与える点では1章で述べたように情報理論の原則と一致する．しかしながら，本報告は実用性を重視したものであり，最短圧縮の理論を述べるものではない．最短圧縮の理論は①原データの情報要素およびその組合せの出現頻度が既知であること，②可変長コードを使用可能なことが必要であり，この2点から実用性が低下している．CNC は①をきわめて明確に高頻度出現要素といえる数字および空白に限定し，さらに②を否定して1バイトに固定した．この2点が実用性を確保するための重要な要素となっている．たとえば3個以上の数字または空白にも1バイトのコードを与えようとすると，コードは新たに $11^3=1,331$ 個必要となり，①より高頻度の組合せを1,331個のなかから選択しなくてはならない．特定の組合せの選択は特殊なデータに対しては効果があるが一般性は低下する．選択をやめ1バイトですべてを表現するためには多数の切換制御文字を必要とし，その使用頻度の上昇とエンコーダ，デコーダの複雑化によって実用的意義は低下する．これに対し CNC では，空白または数字が連接して出現する性質をもつため，切換制御文字の使用頻度が低いことも圧縮の原理となっている．また，この原理によって通常文字の混在するデータに対しても応用可能となっている．

表1の NOBU コードは，

1) 数字としての昇順に並べわかりやすくした，
2) 小数点，正負符号を含め切換制御文字の使用頻度を低くできる，
3) ランレングスコードの順序を保持し 'E 0～'FF を使用できないシステムでエンコードしてもデコーダは変更しなくてもよい，

などを考慮して設計した．しかしながら，もちろん '20～'FF の配置を変更しても，あるいはランレングスコードを増減し，他の文字を含めても，小数点または正負符号のいずれかを削除しても CNC の目的とする効果が大きく変化するわけではない．逆にこの配置を変更しなければならない明確な理由もないので，NOBU コードの標準化が必要とされた場合にも表1は原案としての適性をもつと考える．

表1はさらに，切換制御文字として DC 1，DC 2 を用いている．これは2章で述べたように他の制御文字であってもよい．ただし切換制御文字を2個以上の文字，たとえば ESC シーケンスなどで代用すると圧縮効果は低下する．また，2章および4章で述べたようにエンコーダ，デコーダ間で取り決めをしておけば切換制御文字として用いたコードも伝送可能である．したがって，切換制御には JIS の機能キャラクタ集合（C 0 集合，C 1 集合）のなかから利用上便利な2文字を選択して用いればよい．また，多種のシステムで応用可能なエンコーダ，デコーダを設計する際には切換制御文字の再定義を可能とすることが望まれるが，これは，

1) 情報交換用符号の拡張法 (JIS C 6228) に従った複数キャラクタのシーケンスをエンコーダ，デコーダに出力して再定義する，
2) エンコーダ，デコーダにハードウェア（スイッチなど）を用意して再定義する，
3) エンコーダ，デコーダのソフトウェアを再定義が容易なように作成する，

などによって可能である．

CNC はその原理から，数字または空白が多数含まれるデータに対して効果をもつ．しかしながら，他の文字の混在を許しているため多種のデータに対して適用できる．たとえば，数字で構成されるデータ，氏名など一部通常文字を含むが数字の多いデータ，数字と空白の多いプリンタイメージのデータなどが対象となりうる．さらに，最近では行番号をもつプログラム言語が増加し構造化された表記をするために空白が多いものも増加している．これらのソースファイルも対象となりうる（FDECOMP の場合，ソースファイルは 86.26% に圧縮できた）．

特殊な応用として，すべてのビットパターン ('00～'FF) の転送の場合も応用できる．これは，バイトオリエンテッドな通信手順下では当然何らかの変換が必要である．'00～'FF を 000～255 の3個の数字による表現に変換することは容易である．変換後，CNC を行えば圧縮率はほぼ50%になる．したがって CNC の応用によって原データの1.5倍の長さで転送できる．画像イメージの転送などでは同一ビットパターン（とくに背景やぬりつぶしを表現する '00 または 'FF）が多数連続して出現するため，圧縮効果はさらに大きくできる．

以上述べたように多種のデータに対して応用可能であるため，少なくとも補助（外部）記憶の容量的利用効率を高める効果は十分期待できる．通信回線上での実用性を考えた場合にはエンコードの速度およびデ

コードの速度が問題となり，

$$\frac{\text{圧縮によって減少した文字数}}{\text{エンコード+デコード時間}} > \text{通信回線速度}$$

が成立しなければ実用的意義はない．実現例では，端末用パーソナルコンピュータのデコーダ以外の通信処理部を高速化すれば，少なくとも回線速度 4,800 bit/sec 程度までは十分実用性があると考えられる．より高速な回線での応用に際しては FCOMP, FDECOMP をアセンブラなどの高速化可能な言語で開発する，さらに高速性を配慮したファームウェアによるエンコーダ，デコーダを回線の両端に準備することなども可能ではないかと考える．

7. まとめ

通常用いているデータファイルのなかに数字および空白がきわめて高頻度に出現し，かつ出現の際には連接して出現するという特徴に注目して，連接数字圧縮法 (CNC) を考案した．この方式は，連接した 2 個の数字または空白を 1 個の JIS コード文字におきかえることを基本とし，ランレングスコードも考慮されている．このコード化に用いる数字向きバイト単位コード (NOBU コード) は，同じく数字向きの BCD コードなどとは異なり，バイト単位で構成されており，かつ他の文字との混在が可能である．したがって通信プロトコルに依存しない通信が可能であり，多種のデータに対して応用できる．

参考文献

1) 宮川 洋，原島 博，今井秀樹：情報と符号の理論，p. 266，岩波書店，東京 (1982)．
2) 日立製作所：最適化 FORTRAN 77 言語，p. 236，日立製作所，横浜 (1979)．
3) ソード電算機：*M 200 Mark Series RASM*，p. 354，ソード電算機，東京 (1981)．
4) ソード電算機：M 243 シリーズオペレーティング・システム システムコール説明書，p. 165，同上 (1981)．
5) ソード電算機：M 200 シリーズ Communication TBASIC 言語，p. 46，同上 (1981)．

(昭和 57 年 10 月 25 日受付)
(昭和 58 年 4 月 19 日採録)