9. ファイルの内容に関する実例研究

東京工業大学　理学部

木村　泉・飯島淳子・辻　尚史

概要

ある中型計算機の一般ユーザー向けファイル装置の使用状況を，全数検査によって調査した．その結果，

(1)全記憶バイトの40％以上はEBCDICのスペースコードであった．

(2)オペレーティングシステム(OS)の機能向上にともなって，ファイル使用状況が大中に変化している．古い時期に作られたファイルにはゴミを多量に含むものが多かった．これは，めんどうを避けるためにユーザーが領域を大き目にとっていたためである．

等の事実が判明した．これらの知見に基づいて，基本ソフトウェアの設計について若干の考察をこころみる．

＊　　＊　　＊　　＊　　＊

1．背景・動機

本文は東京工業大学理学部情報科学科設置のFACOM 230-45S電子計算機(以下F45Sと記す)についておこなった，ある実例研究に関するものである．その主題は，ファイル装置の使用状況にある．ここでファイル装置とは，F472Lディスクパック装置であって，目下3ドライブあり，うち2ドライブを一般ユーザーに開放している．ドライブあたりの記憶容量は58MB(メガバイト．Bはバイトをあらわす．以下同じ)である．なおF45Sの主記憶は1語16ビット，サイクルタイム0.7μ秒のコアであり，224KB実装してある．

この計算機の稼動は，昭和48年3月に開始された．以来，学部学生のプログラミング教育に1/3，各種ソフトウェアの開発実験に1/3，オペレーションズリサーチ，グラフ理論等の研究に残り1/3の比重を置いて利用され，現在に至っている．ファイル装置に関しては，機器構成決定に際して意識的にやや大きめのものを選んだこともあり，一方ユーザー

集団は比較的小規模（約150名）であったので，頭初にかなりの余裕があった。そこで，ファイル装置の利用を事実上野放しとし，各ユーザーの良識に信頼する，という方針をとった。くわしく言うと，われわれのF45Sの3ドライブのディスクパック装置のうち，いわゆるシステムレジデンス用の1ドライブ分（以下K0であらわす）は，一般ユーザーは作業場所としてのみ利用し，保存ファイルを作るのには使わない約束とするが，他の2ドライブ分（K1，K2であらわす）は，保存ファイルを作るのに自由に使ってよいとした。また，作ったファイルは規則上はいつ消されても文句は言えないこととし，ただし運用上は保有者に断りなく消すことはしないようにした。

しかし，当然予想されたことながら，次第に保存ファイルの便利さが広くユーザー間に知れ渡るようになり，その結果保存ファイルの数，占有領域が増してきて，時にはK1，K2の容量の80％以上がユーザーの保存ファイルによって占有されることもあるようになり，大きな作業用ファイルはK0に作るしか方法がない，というような状況になってきた。そこで，保存ファイルは果して有効に使われているのだろうか，調べてみよう，と思い立ったというのが，この研究の一つの動機である。もう一つの動機が著者らの好奇心にあったことは言うまでもない。

本文を草している時点ですんでいるのは基礎実験だけであるが，すでに見方によっては大変興味深い結果が得られている。本文では，これらの結果について述べるとともに，現在進行中の実験についても簡単に説明し，またこれに関連して，基本ソフトウェアの設計について若干の考察をこころみたい。

2．方法・方針

基本となる考え方はごく簡単である。F45Sにはシステムユティリティの一種としてVSAVという名のプログラムが用意されており，これを使うとディスクパック上の全保存ファイルの内容を，システム用の領域まで含めて一切合財，磁気テープ上に書き出すことができる。この研究は，その出力磁気テープを，記録内容の具体的意味になるべく立ち入らないように心がけながら解析し，その範囲で言えることを言おうという

立場に立つ．全数検査を重視し，ファイルの典型的な使いかたとはどんなものかを問うことは一応さて措いて，上記の利用状況下で特に人工の手を加えることなく自然にできあがってきた実態を，ありのままに眺め渡すことに重点を置く．具体的には，制御用の情報を含めた全バイトの内容を累計する（3節），単に確保されているだけで有効情報を保持しているのではないことが明白な領域を除いて同様の累計を作る（4節）などである．このうち後者では，ファイル編成に関するオペレーティングシステム（OS）上の約束に，ほんの少しだけ立ち入る必要があり，見方は前者ほど完全に巨視的ではない．このように，多少微視的な見方も必要になることはもちろんであるが，たとえばほとんど同じ内容をもつファイルをあまり意味なく多数保有しているユーザーがいたとしても，この研究では問題にはしない．

なお，もちいたデータは，特にことわらない限り，1975年10月17日現在の状況に基づくものである．

3．バイトの累計

図1にK1内の全保存ファイルについての，バイトデータの出現頻度の累計，およびその百分比をあらわす．百分比の末位は（さしたる理由はないが，たまたま）切捨てとしてある．各バイトの上位4ビットを16進表示の行の見出し0，1，…，Fで，下位4ビットをやはり16進表示の列の見出し0，1，…，Fであらわし，行ごと，および列ごとの総和を添えてある．これを見ると，上位5者は

'40'	が	12,471,264 B	で 42.16%
'AA'	が	4,762,301 B	で 16.10%
'00'	が	3,192,600 B	で 10.79%
'FF'	が	1,141,015 B	で 3.85%
'F0'	が	480,004 B	で 1.62%

であり，これらの合計22,047,184 Bは，全バイト数29,574,274 Bの，実に74.5%にあたる．他のバイトデータの出現頻度は，いずれも1%未満である．上記5者のうち，'40'はEBCDICのスペースコードにあたり，'00'は2進データによくありそうな組合せ，'FF'はファイルの

********** SESSION SUMMARY **********

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	3 192600	196603	100593	44478	62229	34631	29709	23145	88468	27375	23453	19896	35488	16640	33341	49010	3 977659
1	119428	30131	20150	13948	22171	13560	17095	31382	49440	16996	14257	12598	12506	10438	13911	8013	406024
2	194173	34857	9588	9372	18805	8735	8949	7543	29515	9974	9853	105588	14476	12252	6978	174438	655096
3	25425	15907	8425	100771	8741	7559	7525	55061	43089	14294	7532	26987	14010	6621	14339	11094	367380
4	12 471264	27330	5641	5641	13504	5363	6589	5631	8146	4301	5002	84875	22222	79327	15577	10265	12 770678
5	11051	7014	45469	12145	11065	11296	9609	15780	8646	4827	6443	24443	118873	80772	27118	7754	402305
6	127657	20665	3930	3135	4237	4438	5797	3651	9261	3812	4790	105060	12206	85471	21324	12609	428043
7	10352	7174	4530	10504	3642	3958	2839	5726	37422	59913	15790	8560	16026	43772	53700	9754	293662
8	21501	4954	3760	4970	3088	1700	3078	4039	5508	5396	3624	4819	2825	2881	2697	2212	77052
9	5279	3100	5281	5804	3121	1934	2178	15633	3747	3284	4245	2263	2735	2317	2391	4886	68198
A	69975	17042	3583	1917	3423	2093	3123	2437	4081	4498	4 762301	4626	3423	3493	3211	4159	4 893385
B	20703	11814	20919	10390	2420	2284	2655	4474	4269	2842	3702	3405	3403	2639	2914	2757	101590
C	10695	274141	82835	130075	94182	192313	102854	46626	48806	140025	3094	3075	3160	3275	2494	3453	1 141103
D	4363	18366	45861	120561	86775	136482	136421	81424	18533	158358	4999	74851	2474	2221	2785	3151	897625
E	8481	2983	148430	165707	62019	25709	28455	32338	26440	8205	5076	5843	3582	2257	2469	2752	530746
F	480004	237118	165166	125049	98620	71032	60814	60472	55970	49899	3099	2972	5583	2718	4197	141015	1 563728
	16 772951	909199	674161	764467	498042	523087	427690	395362	441341	513999	4 877260	489861	272992	357094	209446	1 447322	29 574274

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	10,79	0,66	0,34	0,15	0,21	0,11	0,10	0,07	0,29	0,09	0,07	0,06	0,12	0,05	0,11	0,16	13,45
1	0,40	0,10	0,06	0,04	0,07	0,04	0,05	0,10	0,16	0,05	0,04	0,04	0,04	0,03	0,04	0,02	1,37
2	0,65	0,11	0,03	0,03	0,06	0,03	0,03	0,02	0,10	0,03	0,03	0,35	0,04	0,04	0,02	0,59	2,21
3	0,08	0,05	0,02	0,34	0,03	0,02	0,02	0,18	0,14	0,04	0,02	0,09	0,04	0,02	0,04	0,03	1,24
4	42,16	0,09	0,01	0,01	0,04	0,01	0,02	0,01	0,02	0,01	0,01	0,28	0,07	0,26	0,05	0,03	43,18
5	0,03	0,02	0,15	0,04	0,03	0,03	0,03	0,05	0,02	0,01	0,02	0,08	0,40	0,27	0,09	0,02	1,36
6	0,43	0,07	0,01	0,01	0,01	0,01	0,02	0,01	0,03	0,01	0,01	0,35	0,04	0,28	0,07	0,04	1,44
7	0,03	0,02	0,01	0,03	0,01	0,01	0,01	0,01	0,12	0,20	0,05	0,02	0,05	0,14	0,18	0,03	0,99
8	0,07	0,01	0,01	0,01	0,01	0,00	0,01	0,01	0,01	0,01	0,01	0,01	0,01	0,01	0,00	0,00	0,26
9	0,01	0,01	0,01	0,02	0,01	0,00	0,00	0,05	0,01	0,01	0,01	0,00	0,00	0,00	0,00	0,01	0,23
A	0,23	0,05	0,01	0,00	0,01	0,00	0,01	0,00	0,01	0,01	16,10	0,01	0,01	0,01	0,01	0,01	16,54
B	0,07	0,04	0,07	0,03	0,00	0,00	0,00	0,01	0,01	0,01	0,01	0,01	0,01	0,00	0,01	0,00	0,34
C	0,03	0,92	0,28	0,44	0,31	0,65	0,34	0,15	0,16	0,47	0,01	0,01	0,01	0,01	0,00	0,01	3,85
D	0,01	0,06	0,15	0,40	0,29	0,46	0,46	0,27	0,06	0,53	0,01	0,25	0,00	0,00	0,00	0,01	3,03
E	0,02	0,01	0,50	0,56	0,21	0,08	0,09	0,10	0,08	0,02	0,01	0,02	0,01	0,00	0,00	0,00	1,79
F	1,62	0,80	0,55	0,42	0,33	0,24	0,20	0,20	0,18	0,16	0,01	0,01	0,01	0,00	0,01	3,85	8,66
	56,71	3,07	2,28	2,58	1,68	1,76	1,44	1,33	1,49	1,73	16,49	1,65	0,92	1,20	0,70	4,89	100,00

図1．K1の記憶バイト── 累計および百分比

あき領域のマーカーとしてシステムが使っており，'F0'はEBCDICコードの'0'にあたるので，いかにももっともらしいが，'AA'についてはあまり簡単に説明がつかない．これについては次節で述べる．

なお，図1の集計において処理されたファイルの総数は423個であった．K2についても同様の集計をおこなったが，傾向は全体として上記と似たものであり，'40'の出現頻度は47.09%であった．

図2には，典型的なソースプログラムファイル（大きさ10トラック，62,848B．内容は775行から成るFORTRANプログラム）についての図1と同一形式の累計を示す．ここでは'40'の頻度が79.57%と，大中に増していることがわかる．頻度0の項目が多数あるが，これはEBCDICコードで使われない部分にあたる．なお'00'など，EBCDICコードとしては普通使われないものが1回以上あらわれているのは，この累計が制御情報（ファイルラベル）こみのものであることによる．

図1，2の百分比に示すような送出確率をもつ内部記憶のない情報源を考え，そのエントロピーを算出してみると，それぞれ3.814ビットおよび1.704ビットとなる．これは，バイト間の相関を無視してさえ，8ビットであらわされている情報をほぼこれらのビット数まで圧縮できるはずであることを示している．

4. 有効部分のみの累計

F45SのOSでは，ディスクパック上の領域がOSによってトラック単位で割り当てられ，ユーザーはその領域の中を適宜使用することができるが，その際既製の入出力サブルーチン群（FCP）を使うことを標準とする．FCPによるファイルの構成法のうち，もっとも普通なのは順編成と分割型順編成である．今回の集計でも，これら以外の編成をもつファイルは一つもあらわれなかった．

ところで，順編成および分割型順編成では，与えられた領域が頭から使われて行き，有効情報の切れ目には特別のEOFがついている．そのうしろはあき領域であり，ゴミが入っている．そこで，ファイル編成についての知識を使ってこのゴミを捨て，有効部分のみについて図1と同様の集計（図示は省略）を作ってみたところ，総バイト数が15,377,964B，

FILE0003: D9667G SEQ

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	59	1	1	0	1	0	0	1	0	1	2	0	0	0	0	0	66
1	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	3
2	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	2
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	49723	0	0	0	0	0	0	0	0	0	0	199	1	374	25	1	50323
5	27	0	0	0	0	0	0	0	0	0	1	1	2129	374	1	0	2533
6	17	266	0	0	0	0	0	0	0	0	0	162	2	0	1	0	448
7	0	0	0	0	0	0	0	0	0	0	0	1	0	44	147	2	194
8	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0	564	188	672	113	456	326	301	104	666	0	0	0	0	0	0	3390
D	0	0	36	787	441	566	502	218	65	337	0	0	0	0	0	0	2952
E	0	0	306	619	312	0	52	6	38	59	0	0	0	0	0	0	1392
F	224	415	123	90	65	79	55	55	41	22	0	0	0	0	0	10	1179
	50054	1246	654	2168	932	1101	935	581	249	1085	3	364	2133	792	174	13	62484

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	0,09	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,10
1	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
2	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
3	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
4	79,57	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,31	0,00	0,59	0,04	0,00	80,53
5	0,04	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	3,40	0,59	0,00	0,00	4,05
6	0,02	0,42	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,25	0,00	0,00	0,00	0,00	0,71
7	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,07	0,23	0,00	0,31
8	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
9	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
A	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
B	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
C	0,00	0,90	0,30	1,07	0,18	0,72	0,52	0,48	0,16	1,06	0,00	0,00	0,00	0,00	0,00	0,00	5,42
D	0,00	0,00	0,05	1,25	0,70	0,90	0,80	0,34	0,10	0,53	0,00	0,00	0,00	0,00	0,00	0,00	4,72
E	0,00	0,00	0,48	0,99	0,49	0,00	0,08	0,00	0,06	0,09	0,00	0,00	0,00	0,00	0,00	0,00	2,22
F	0,35	0,66	0,19	0,14	0,10	0,12	0,08	0,08	0,06	0,03	0,00	0,00	0,00	0,00	0,00	0,01	1,88
	80,10	1,99	1,04	3,46	1,49	1,76	1,49	0,92	0,39	1,73	0,00	0,58	3,41	1,26	0,27	0,02	100,00

図2. 典型的なソースプログラムファイル

ともとの51.99％（未尾切捨て．以下同様）に減じた．また，バイト情報の出現頻度は，上位5者について

'40' が	6,449,081 B	で	41.93 %
'00' が	2,460,424 B	で	16.00 %
'FF' が	960,646 B	で	6.24 %
'F0' が	233,611 B	で	1.51 %
'C1' が	200,134 B	で	1.30 %

となり，5者の総計は 10,303,896 B（67.00％）となった．他のバイトの出現頻度はいずれも1％未満であった．図1の大量の'AA'は，ほぼ完全に消滅して，わずか4404B（0.02863％）を残すのみとなった．

総エントロピーは4.180ビットとなった．図2のファイルはたまたまったくゴミを含んでいなかったので，エントロピーは1.704ビット，と不変であった（前節末尾参照）．

なお，VSAVによってファイルを吸いあげている最中にたまたまそのファイルが更新中であると，順編成または分割型順編成のファイルとしては矛盾した形式のものが取り出されてくることがある．混乱が生じるのを避けるため，そのようなものは有効部分の累計からは除外することにした．実際には，大きさ 55,960 B の分割型順編成ファイル1個が除外された．

'AA'が激減した原因を調べるため，'AA'を全記憶バイトの1％以上含むファイル（K1内の）を抜き出してみたところ，計13個あり，それらに含まれる'AA'の個数は総計 4,756,349 B であった．これはK1全体に含まれる'AA'の，実に 99.87％（!）にあたる．これらのファイルの総記憶バイト数は 6,519,190 B，総有効バイト数はわずか 523,270 B で，後者の前者に占める比率は 8.02％であった．つまりこれらのファイルの内容は，大部分がゴミで，そのゴミの中にK1に含まれる限りのほとんどの'AA'が集中していたことになる．

諸般の事情からみて，ことの原因は次のようなものだったと考えられる．実はこれら13個のファイルは，1個を除いて計算機稼動開始後1年以内に作られたものであり，当時ディスクパックの中は乱々とあいて

いたので，これらのファイルは充分余裕のある広々とした作り方で作られた．ところで当時，ディスクパックの初期設定ルーチンはテストのために全領域に"AA"を書くようにできていたので，初期設定したてのディスクパックは"AA"で充たされており，ファイルを作る際に領域をたっぷりとると，あき領域にこの"AA"が多数取り込まれることになっていた．その"AA"が，あいたまま受け継がれて現在に至ったのである．

実際，これらのファイルは，いずれもシリンダー単位（40トラック単位）の大きさをもっており，大変おおらかな作り方をされたものであることが，よくわかる．この辺のことについては8節で再論する．

5．編成別のエントロピー分布

前節の有効部分のみの累計によって，ファイルごとに3節末尾で述べた意味のエントロピーを求め，これを順編成と分割型順編成にわけてヒストグラムにあらわしたものを図3に示す．分割型順編成に関するヒストグラムには，はっきり二つの峯があるが，これらのうちエントロピーの値の小さい側の峯は相対形式のプログラムファイルによってもたらされたもの，もう一つの峯は実行形式のプログラムファイルによってもたらされたものであると考えられる．

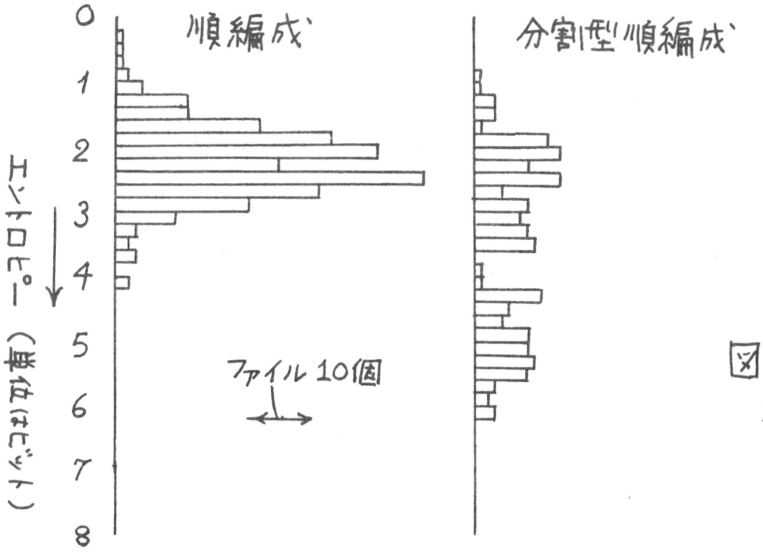

図3．編成別のエントロピー分布

6. 4バイト単位の集計

以上では，バイトを基本単位とみて実験を進めたが，実は各バイトが独立であるはずはなく，たとえばソースプログラムファイルにおいてはスペースの次にはスペースがあらわれやすいはずであるし，実行形式のプログラムファイルでは飛越し命令の命令コードがたくさんあらわれてもふしぎはないと考えられる．そこで，4節と同様の集計を，4バイトを単位とみておこなってみた．すなわち，ファイルは有効部分のみ取り出し，各レコードの頭から4バイト区切りで切り出し，レコードの終端に半端が出ればそれは別個に集計した．K1についての集計結果の一部を図4(a)に示す．また，図2で例示したファイルに関する同様の集計結果を図4(b)に示す．図4(a)，(b)とも，各欄は左から順に，4バイトのパターン（16進表示），出現回数，パターン全体に占める百分比，その累計をあらわし，出現回数の多い順に40パターンずつ示してある．

K1全体についての，パターンの総数は3,848,321，種類数は314,766，エントロピーは4バイトあたり8.560ビットである．また，図4(b)のファイルについては，パターン数15,621，種類類は1008で，エントロピーは3.003ビットであった．ただし，ここでエントロピーとは，これらの4バイトのパターンが出現したと全く同じ比率でこれらのパターンを出す，内部記憶のない情報源のエントロピーをさす．このエントロピーの値が，バイト当りにすれば，4節で得た値と比べて半分以下に下っている，という事実は，バイト間の相関が無視できないほど大きいことを物語るものである．

同様の集計を，2バイト単位についてもおこないつつある．

7. 圧縮のための符号化

エントロピーがかくかくしかじかである，ということから，ただちにファイルはその分無駄使いされている，と結論することはできない．情報圧縮の原理的可能性が保証されていたとしても，そのための符号化があまり複雑では，実用的な意味はない．簡単で，しかも有効な符号化法がないかどうか，考えてみる必要がある．

40404040	1441061	37.44649	37.44649
00000000	442346	11.49452	48.94101
FFFFFFFF	233939	6.07899	55.01999
10000000	14742	0.38308	55.40307
A0A01000	12438	0.32321	55.72628
52085208	11954	0.31063	56.03691
00001000	7304	0.18980	56.22670
0000A0A0	7141	0.18556	56.41226
0F000F00	6448	0.16755	56.57982
5C5C5C5C	6415	0.16670	56.74651
F0F0F0F0	4267	0.11088	56.85739
33200002	4251	0.11046	56.96786
C6C6C6C6	4111	0.10683	57.07468
60606060	4105	0.10667	57.18135
C1C13320	3706	0.09630	57.27765
5E404040	3601	0.09357	57.37123
00010000	3326	0.08643	57.45765
2F790001	3253	0.08453	57.54219
4040C4C3	3208	0.08336	57.62555
C3404040	2992	0.07775	57.70329
4040C9C6	2861	0.07434	57.77764
2B200002	2857	0.07424	57.85188
5D404040	2459	0.06390	57.91578
5C404040	2365	0.06146	57.97723
D4C3D9F0	2347	0.06099	58.03822
C6404040	2323	0.06036	58.09858
C1C12B20	2199	0.05714	58.15573
4040C1C9	1979	0.05143	58.20715
C3D4D5F0	1950	0.05067	58.25782
F1404040	1881	0.04888	58.30670
00030000	1815	0.04716	58.35386
D6D74040	1811	0.04706	58.40092
F1F1F1F1	1791	0.04654	58.44746
B2602F79	1706	0.04433	58.49179
E3C8C540	1687	0.04384	58.53563
00010001	1672	0.04345	58.57908
4040C3C1	1671	0.04342	58.62250
4040D7D4	1659	0.04311	58.66561
C4404040	1652	0.04293	58.70854
4040D9F4	1649	0.04285	58.75139

(a)

40404040	11436	73.20914	73.20914
5C5C5C5C	457	2.92555	76.13469
4040C3C1	200	1.28033	77.41502
4040C3D6	118	0.75539	78.17041
4040C9C6	86	0.55054	78.72095
D5E3C9D5	64	0.40970	79.13066
E4C54040	64	0.40970	79.54036
F1404040	59	0.37770	79.91806
D4D4D6D5	55	0.35209	80.27015
4DD4C6D3	47	0.30088	80.57103
D3D340E3	47	0.30088	80.87190
C1C74BC5	47	0.30088	81.17278
C4404040	43	0.27527	81.44805
40C7D640	41	0.26247	81.71052
4040C5D5	38	0.24326	81.95378
4040D9C5	38	0.24326	82.19704
C35C5C5C	37	0.23686	82.43390
D3D340C3	35	0.22406	82.65796
E3E4D9D5	34	0.21766	82.87562
D3D340D6	32	0.20485	83.08047
E3C9D5C5	31	0.19845	83.27892
C2D9D6E4	31	0.19845	83.47737
4040E2E4	31	0.19845	83.67582
D9D3C640	30	0.19205	83.86787
C1C24040	29	0.18565	84.05352
D84BF05D	29	0.18565	84.23917
5D404040	28	0.17925	84.41841
4040C9D5	27	0.17284	84.59126
D3D340D3	26	0.16644	84.75770
F2404040	24	0.15364	84.91134
5C5C4040	24	0.15364	85.06498
4040D4C6	21	0.13443	85.19941
D84BF15D	21	0.13443	85.33385
D3D340D4	20	0.12803	85.46188
D3C1C77E	20	0.12803	85.58991
405C405C	20	0.12803	85.71794
C3C84DF1	19	0.12163	85.83957
E3C5C7C5	17	0.10883	85.94840
F0404040	17	0.10883	86.05723
D3D340E2	16	0.10243	86.15966

(b)

図4. 4バイト単位の累計: (a) K1全部,
(b) 典型的なソースプログラムファイル

ソースプログラムファイルに関しては，いくつかの連続したスペースを一つのバイトであらわせるようにする，といったあたりが現実的であろう．図2のファイルについて，そのような方法による圧縮をおこなってみたところ，単純な計算では約1/4までの圧縮ができることがわかった．さらにくわしいことについては，検討中である．

8．考察

最近の大容量記憶装置（ファイル装置）の発達は著しく，したがって8ビット入るはずのところに実効的には4.180ビットまたは1.704ビットしか入っていなかったとしても，どうということはない，という考えにとらわれ勝ちである．ながそれは，8ビット対4.180ビット（または1.704ビット）という損が，それに見合うだけの別の利益（たとえばプログラムの書きやすさ）によって帳消しになっていることを前提とする．同じ比率でファイル装置の台数をふやせばよい，と考えるのは誤りである．同種のファイル装置を考える限りにおいて，1回の呼出して読書きできる情報の量は不変であり，したがって記憶容量の損は，そのまま呼出し回数の損として効いてくるおそれがある．

そこで，たとえば7節で言及したような方法で情報の圧縮をはかろうと考えたとすると，レコード長可変のファイル（可変長ファイル）を扱う必要が生ずる．それはやはりめんどうだから，やめておこう，などとつい言いたくなる．

なが，そのめんどうは，その気になればいわゆるアクセスメソッドの中に閉じ込めることができるものである．コンパイラやユーザーのプログラムから見ると，どう見ても固定長ファイルとしか見えないものが，正体はソフトウェアの衣を着た可変長ファイルであっても，まったくさしつかえないはずである．要はその気になるかならないかの問題である．

もちろんこれは，可変長ファイルを使え，と言っているのではない．可変長ファイルを使うことを含めて，今までおこなわれていなかったやりかたのうちに，検討に値するものがありそうに思われるから考えなおしてみてはどうか，と言っているのである．みだりに伝統にとらわれるべきではない．

そして，この辺のことについての冷静な判断には，本文で一例を示したような大つかみな実例研究が，大いに役立つと考えられる．

もう一つ，本文の研究から浮び上ってくる問題点は，基本ソフトウェアに対するユーザーの反応に関するものである．4節で，普作られたファイルには領域を驚くほど広々と占有しているものが多い，ということを指摘した．そうなった理由は，4節で述べたように当時はそれが可能だったからでもあるが，実はもう一つ事情があって，FACOM 230-45S のOSの当時の版では，ファイルの記憶領域の所要量をユーザーに前もって算定させ，その算定が少しでも低すぎたら，途中までおこなってきたデータ処理を一切おじゃんにすることが建前となっていたのである．そのような設計のもとでは，ユーザーは自衛手段として，領域の算定を大き目にするようになる．自信のないユーザー（それが大部分である！）ほど，その傾向が著しい．ファイルの中に有効部分が8%しかなかったとしても，ユーザーをあまり責める気になれない．

ユーザーというものは，とかく与えられたものを無批判にありがたがり，靴に足をあわせる，のたとえ通り，使いにくいソフトウェアにもいつしか順応してしまうものである．だが，そうだからと言って，悪いソフトウェアでいい，ということにはならない．悪いソフトウェアはまわりまわって，結局システムの，全体としての効率を下げてしまう．大つかみな実例研究は，その辺のことを明るみに引き出すのにも役立つと考えられる．

謝辞

白濱律雄君の助力に感謝する．