REPRINT

The views, conclusions, or recommendations expressed in this document do not necessarily reflect the official views or policies of agencies of the United States Government.

This document was produced by SDC in performance of contract SD-97

9/13

TECH MEMO

a working paper

System Development Corporation / 2500 Colorado Ave. / Santa Monica, California

TM- 1456/008/00

AUTHOR R. L. Patrick

TECHNICAL

RELEASE S. M. Cooney

for D. L. Drukey

DATE 5 Sep 63 PAGE 1 OF 23 PAGES

AD662955

DEVELOPMENT AND MANAGEMENT OF A COMPUTER-CENTERED DATA BASE: PROCEEDINGS OF THE SYMPOSIUM (10-11 June 1963)*

Part 8: Adapting Mass Storage Equipment for the Handling of a Data Base

R. L. Patrick

DEC 28 1967

*This symposium was conducted by the Command Research Project, in cooperation with the Advanced Research Projects Agency. This talk, one of the presentations at the symposium, is to be incorporated in a Technical Memorandum containing the complete proceedings.

A-2450 10/62

Reproduced by the CLEARINGHOUSE for Federal Scientific & Technical Information Springfield Va 22151

ADAPTING MASS STORAGE EQUIPMENT FOR THE HANDLING OF A DATA BASE

Robert L. Patrick*

ABSTRACT

The introduction elaborates the title of the oral paper, and lists the hardware that is available for mass storage, describing its characteristics. Definitions of a few necessary terms are presented, and operations on files are listed and discussed. A discussion of the important basic system considerations of I/O balance, timing, and available design strategies follows.

System-analysis criteria are supplied so that the designer, once he has defined his task and chosen his strategy, may proceed to design within the constraints chosen. Several additional considerations are also given.

Following a summary of the material a startling time-comparison from a recent technical paper is presented.

---

*Mr. Patrick, Computer Specialist Consultant, has been active in resolving problems of computer applications in the analysis and design of large-scale information-handling systems.

INTRODUCTION

In titling this report "Adapting Mass Storage Equipment for the Handling of a Data Base," I don't mean hardware adaptations, but "software." I suggest we adapt the devices that are commercially available to our problems. There are people who are working so far out in front that they can set specifications on the hardware they require, and get it built in time. There are other times when specifications are set for brand new hardware and it doesn't quite materialize, or something goes wrong, and it costs IBM 17 million dollars. Clearly, most of us are going to have to live with generally available commercial products.

First, one more point of clarification: This is a state-of-the-art talk, but let me modify it by inserting an adjective: This is a state-of-the-art practice. What some talk about as state-of-the-art is everyday labor; others mean things that we think we can do and are trying to demonstrate in the lab.

Let us not think that we're building large data bases all by ourselves; no matter how elating this concept, it's not true. There are several excellent, popular efforts going on, and some of them predate our work by 5 or 6 years. As an example, the business-data processing field processes large files and works with a data base. Nor let us think that we're alone in working with hugh files. I recently attended a meeting sponsored by the Library of Congress and the National Science Foundation. We discussed, among other things, the National Union Catalog, an index file. It is a large data base, containing 14 million cards. This building would just about hold the National Union Catalog.

In addition, the County of Los Angeles computes our tax billing for 6 million parcels of property on their D-1000 computer. They "goofed" a time or two, but solved their problems. As another example, Title Insurance in Los Angeles has about 6 million legal descriptions on parcels of property. Every time one of us buys or sells a piece of property in Los Angeles, the transaction is processed against that file, on an H-800.

(Note: I will often use IBM as an example, not because I recommend IBM equipment, but because I am familiar with it.)

As I said, I will restrict myself to the state-of-the-art--in the field, if you will. Figure 1 shows the three major types of hardware available: Magnetic tape, an artist's schematic of magnetic drum, and a magnetic disc. (Unfortunately, I know of no useful, associative memories now operating.)

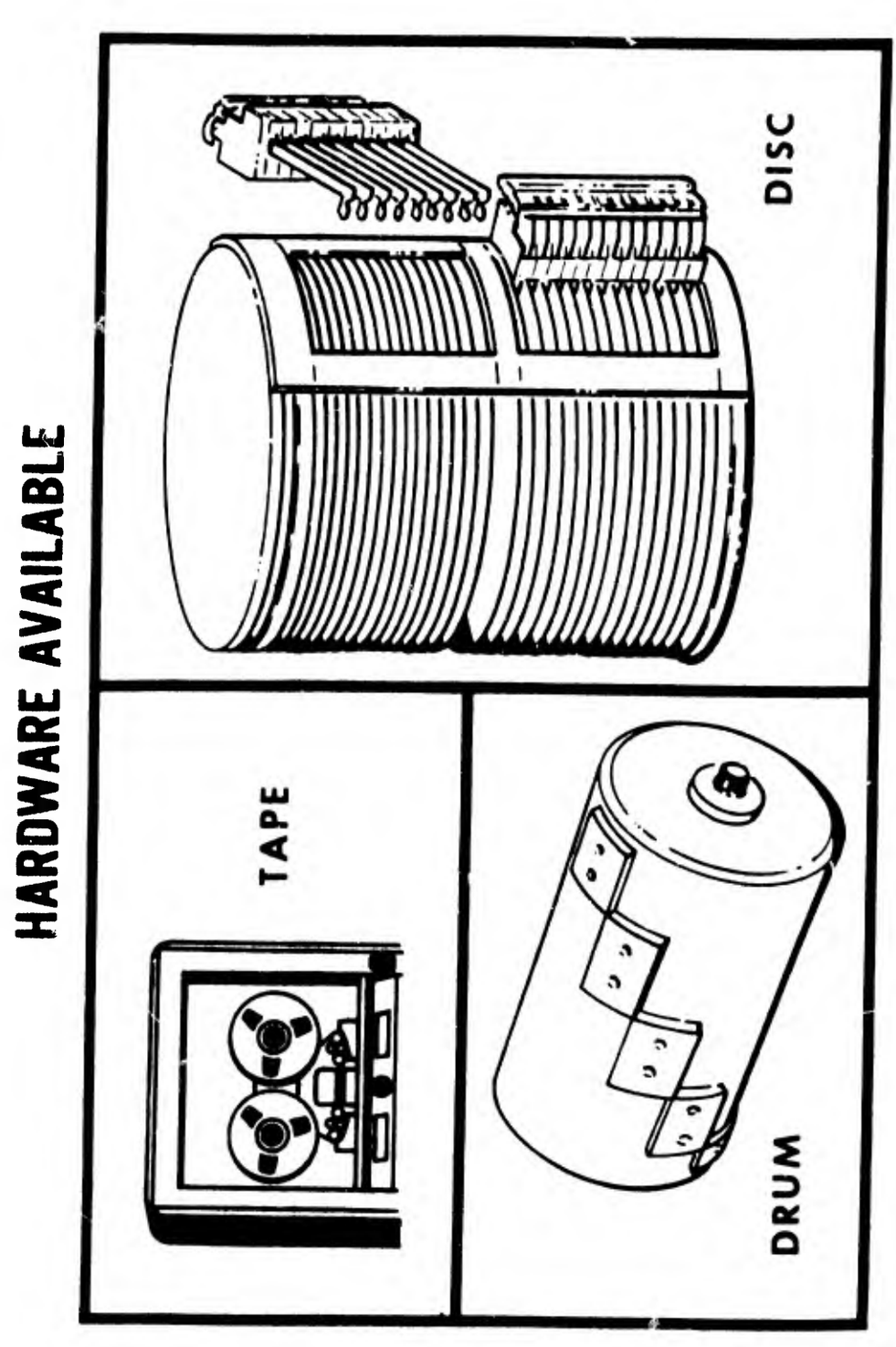


FIGURE 1.

Let's start with magnetic tape, for example, an IBM 729 Mod IV tape, which passes tape under heads at 112.5 inches per second. It uses six channels, plus parity, to make up a frame or a character; reads and writes 62.5 thousand characters per second; and starts in 7.3 milliseconds. You don't need to worry about the stop time because that is "free" (the computer is released). It has 556 characters per inch. If you had a 1668-character record, you would be using the tape at 80% efficiency. At this efficiency, the 2000-foot spool of tape would hold 10 million characters, and could be read in less than 4 minutes.

Now direct your attention to the magnetic drum, a form of cyclic storage. The principal provider of this type of storage is Remington Rand. A typical present-day device is their "FAST-RAND" device, which rotates every 70 milliseconds, or 860 rpm. On a "FAST-RAND" drum, 64 million characters can be stored. It's big, and it has 151 kc transfer rate (151 thousand characters per second transfer rate), and 670,000 characters can fit in a cylinder in one setting of the head servo, before the heads have to be moved. If you must move the heads, then a "servo" penalty is required--30 milliseconds if you move them to the next track and 86 milliseconds if you move them full width. A current modern-day drum is a little different from preconceived notions of a drum. These huge drums do not have heads fixed to the machine frame, but are "servo" controlled; they slide back and forth, to have less electronics for a big storage volume.

On the right is a disc storage. There are many people that make these, but the pacesetter is the IBM 1301 which they call "random access." It rotates every 34 milliseconds, or 1760 rpm. It has 56 million characters on it. Now we can compare. On tape, we have 10 million characters; on the "FAST-RAND," we have 64 million characters; and on the double module 1301, we have 56 million characters. It has a character-transfer rate of 90 kc--90 thousand characters per second. You can get only 224 thousand characters without moving the heads. If you want to move the servo to the next cylinder, it takes 50 milli-seconds to move one track and 180 milliseconds for the full swing.

Thus, these three create a profusion of types of hardware, with similar abstract characteristics, but dissimilar specifics. So we say to ourselves, "What things should we note?" With tape, the first thing you should note and be aware of is that it is a "serial" storage medium; this medium has a set of characteristics all of its own.

First, note the start-stop time, because you don't want it to take forever to start. Notice the transfer rate; the checking features that are in that tape; and, finally, the compatibility--and by compatibility, I mean brute hardware compatibility. Put 7 channels on the machine and record them the same way. Put 3/4 inch gaps on them. Thus, you can take a file from Machine A to Machine B; you can pass files up to a higher-level command and down.

In bulk storage--either drum or disc--you are interested in a different slant. Note, of course, the rotation rate, because that determines how long you must wait, on the average from the time you request some information off that disc to the time the information flows to you. It's cyclic. Observe also the transfer rate--how long does it take you to get a block after you request it. Note the "servo" time--how long it takes to move these heads back and forth--and look at the ratio of the heads times and track capacity to the number of characters stored, because that is one measure of the probability that you'll have to stand a head "servo" movement while you are reading a lengthy block of data.

Finally, examine the reliability statistics on the machine because there is no means of checking. There's only one head. With magnetic tape (with our read-after-write devices) by the time you finish writing a piece of magnetic tape, you are assured (if you don't get a parity error and if the hardware is working properly) that that tape was correctly written. The only way to check a disc is to wait a full revolution, and then read it back. This is a severe penalty, but it is the current state of the art.

Some manufacturers just hang any old file on any old computer and say, "We've got bulk storage." This is often very clumsy, and results in poor performance. The electronic channel that connects the physically rotating mechanical device to the computer is very important, and sometimes high-priced channels are worth the money. The interrupt logic of the central computer is also extremely important; often it determines what else you could be doing and how close you have to watch the input-output device.

Now let us define a few little terms. Item A on Figure 2 is the familiar 3-drawer file, from which we removed the coffee cup and the old potted plant before we photographed it. It's an excellent random-access device, with an access mechanism that is sometimes quite picturesque! For low-activity-ratio files (what I like to think of as dormant files) it's excellent. It is a poor device if time currency is required, if the activity is high, or if the files must be neat.

On the right side, B, is available hardware. This is characterized by high performance and high cost. The tapes, of course, have the advantage of being removable, shipable, and storable, and excellent for serial files. The disc and drums are valuable, compared to tape, because although they increase the minimum access time to an item of information, they decrease the maximum access for an item of information, when the requirement is random, i.e., not preplanned. Furthermore, they are valuable when the requirement dictates that the file must be on-line and that batching cannot be tolerated.

DEFINITIONS OF TERMS

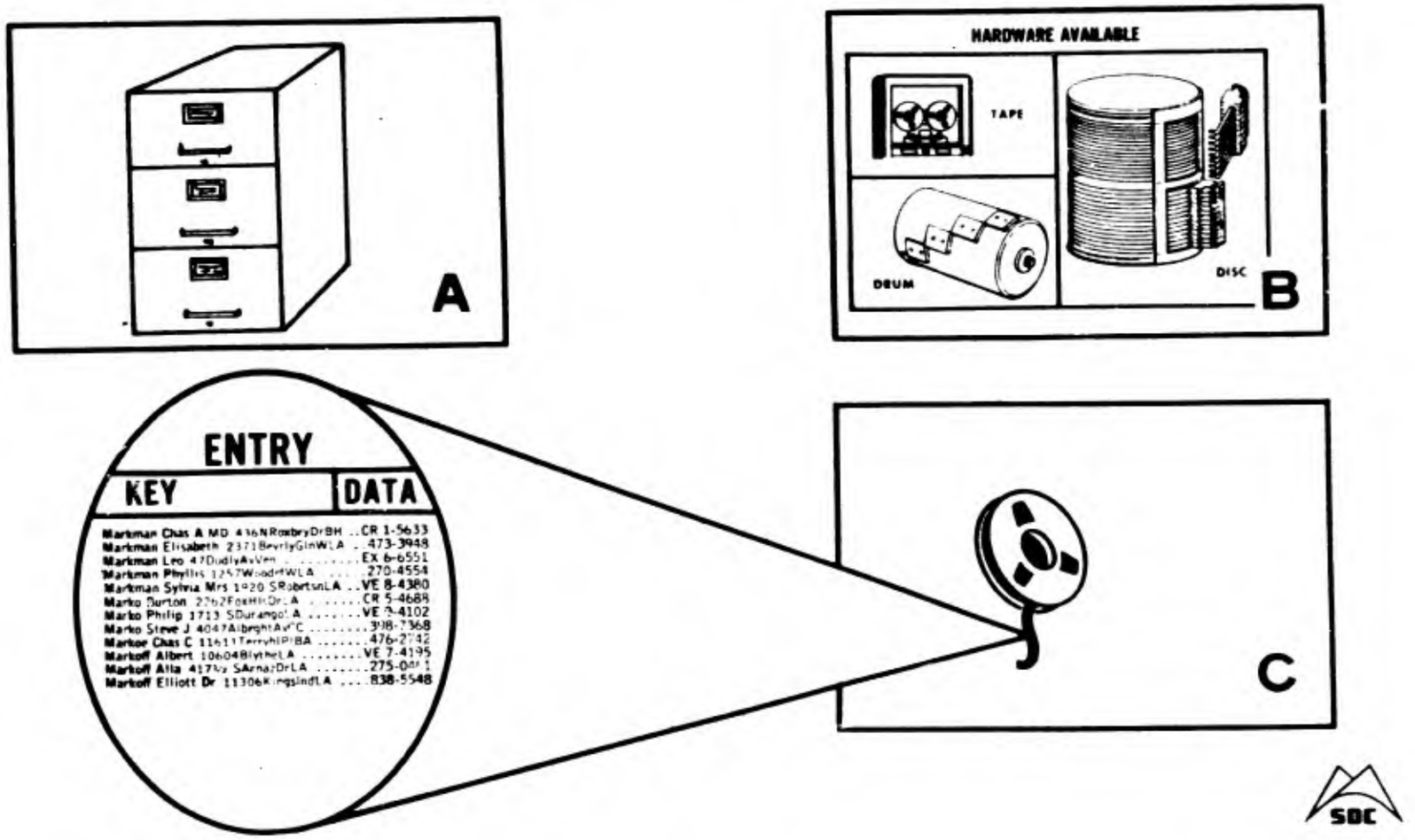


FIGURE 2.

Usually, to back up the disc, as shown in Item C, we have a magnetic tape for our "grandfather" file. In addition, we usually maintain the serial file of incoming transactions even though we are operating with a disc on-line. If a disc or a large drum on-line were constantly updated whenever a message came in from a communication system (e.g., ComLogNet), if orders to ship an item were automatically cut, if the amount available were decreased, and if this were tallied and checked against the re-order level, the disc file would be current at all times. To do this, you would probably keep a magentic tape serial file in contiguous arrival order, in parallel, to keep track of all the transactions processed against the file. Then, in case of failure, you could take a "grandfather" tape copy of the disc file, reprocess the transactions as a batch against it, and regenerate the file.

I'd like to narrow the subject field a little bit more by defining "data base." To me, a data base is a set of files--an ordered collection of entries, each of which consists of a key and some data.

On the lower left of Figure 2 is a section of a telephone book. The key consists of the last name of a party listed in the telephone book. There are certain modifiers to this key, to narrow the definition. These are adjectives that do just what your high school English teacher told you they did--they narrow the scope of the noun.

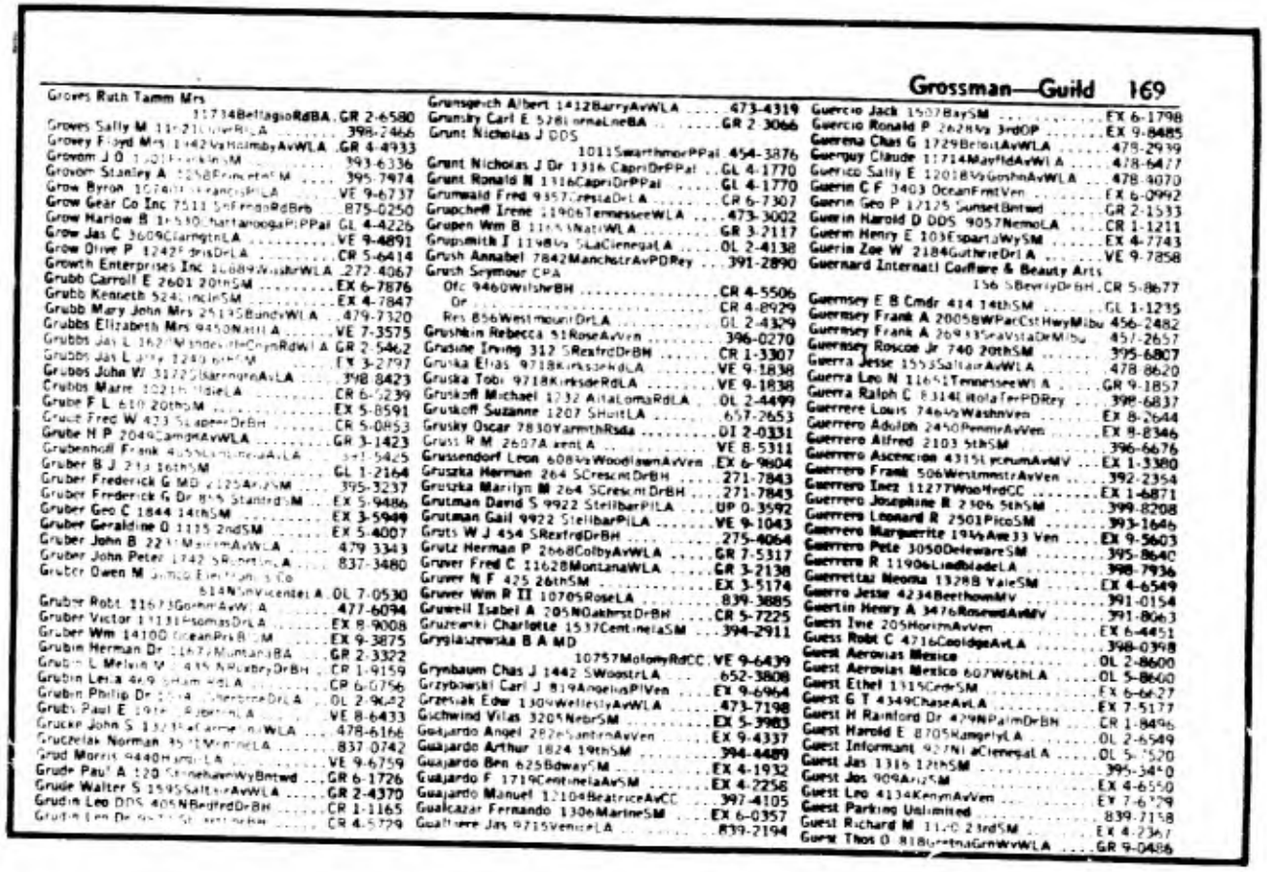
In the example, Leo is an adjective that narrows the set of all Markmans to Leo Markman, and, if there were many Leo Markmans, the middle initial would narrow it further; then the address; and then maybe a "Jr." This sorted file would be entered on last name, first name, and, perhaps, address. And the datum related to that entry--the telephone number--would be obtained.

OPERATIONS ON FILES

The select operation on a file (this is my definition) is the case in which the key that is related to the desired datum is known unambiguously. (See Figure 3.) As an example, if you wanted the phone number of Owen M. Gruber, you would scan the list on the left side and find that Owen M. Gruber was a unique entry in that file.

On the other hand, we perform file operations called "searches." When you "search" a file, you enter a file with a partial key, or with a key that is not unique--an ambiguous key. As an example, if you wished to call an old shipmate of yours, named Guernsey, from the Naval Academy--you called him "Cow" when he was in the Academy and you don't remember his first name, but you know he is in Los Angeles--you would then enter the file, scan down the Guernsey's to see if there were any clue to your friend's first name. You'd find E.B. Guernsey, a Commander, who lives in Santa Monica. With a partial key, you can search a sorted file, and have some chance of achieving a hit.

OPERATIONS ON FILES



Grossman—Guild 169

Groves Ruth Tamm Mrs
11714BellagioRdBA..GR 2-6580
Groves Sally M 11[illegible]RdLA..398-2466
Growey Floyd Mrs 1442VaHolmbyAvWLA..GR 4-4933
Growom J O 1[illegible]SM..393-6336
Growom Stanley A 1[illegible]SM..395-7974
Grow Byron 1074[illegible]RdLA..VE 9-6737
Grow Gear Co Inc 7511 SFreemRdBrb..875-0250
Grow Harlow B 1[illegible]PiPal..GL 4-4226
Grow Jas C 3609CarmgtnLA..VE 9-4891
Grow Olive P 1242[illegible]drisDrLA..CR 5-6414
Growth Enterprises Inc 16889VenturWLA..272-4067
Grubb Carroll E 2601 20thSM..EX 6-7876
Grubb Kenneth 524[illegible]inclnSM..EX 4-7847
Grubb Mary John Mrs 2914SBundyWLA..479-7320
Grubbs Elizabeth Mrs 9450N[illegible]LA..VE 7-3575
Grubbs Jas L 1620M[illegible]nCoynRdWLA..GR 2-5462
Grubbs Jas L [illegible]..EX 3-2797
Grubes John W 31725SarmgtnAvLA..398-8423
Grubbs Marie 1021[illegible]sleLA..CR 6-5239
Grube F L 610 20thSM..EX 5-8591
Grucci Fred W 423 S[illegible]DrBH..CR 5-0853
Grube H P 2049CamdgnAvWLA..GR 3-1423
Grubenhoff Frank [illegible]..[illegible]-5425
Gruber B J 214 16thSM..GL 1-2164
Gruber Frederick G MD 2125AvASM..395-3237
Gruber Frederick G Dr 815 Stanford[illegible]M..EX 5-9486
Gruber Geo C 1844 14thSM..EX 3-5949
Gruber Geraldine O 1115 2ndSM..EX 5-4007
Gruber John B 221[illegible]MarionAvWLA..479-3343
Gruber John Peter 1742 S[illegible]nLA..837-3480
Gruber Owen M [illegible]..
514N[illegible]vicenteLA..OL 7-0530
Gruber Robt 11673GoshenAvWLA..477-6094
Gruber Victor 11131ThomasDrLA..EX 8-9008
Gruber Wm 14100OceanPrsBlvdSM..EX 9-3875
Grubin Herman Dr 1167MontanaBA..GR 2-3322
Grubin L Melvin M [illegible]NPalmDrBH..CR 1-9159
Grubin Leila 469 S[illegible]dLA..CR 6-0756
Gruben Philip Dr [illegible]..OL 2-9042
Grubs Paul E 191[illegible]..VE 8-6433
Grucke John S 1321[illegible]armeLnWLA..478-6166
Gruczelak Norman 851[illegible]nLA..837-0742
Grud Morris 9440H[illegible]LA..VE 9-6759
Grude Paul A 120 S[illegible]WyBntwd..GR 6-1726
Grudie Walter S 1595SaltairAvWLA..GR 2-4370
Grudin Leo DDS 405NBedfrdDrBH..CR 1-1165
Grudin Leo Dr [illegible]..CR 4-5729

Grumsgench Albert 1412BarryAvWLA..473-4319
Grumsky Carl E 5281ormaLneBA..GR 2-3066
Grunt Nicholas J DDS
10115WarthmorPPal..454-3876
Grunt Nicholas J Dr 1316CapriDrPPal..GL 4-1770
Grunt Ronald N 1316CapriDrPPal..GL 4-1770
Grumwald Fred 9857CrestaDrLA..CR 6-7307
Grupchoff Irene 11906TennesseeWLA..473-3002
Grupen Wm B 1[illegible]NatlWLA..GR 3-7117
Grupsmith J 11948[illegible]SLaCienegaLA..OL 2-4138
Grush Annabel 7842ManchstrAvPDRey..391-2890
Grush Seymour CPA
Ofc 9460WilshrBH..CR 4-5506
Or..CR 4-8929
Res 856WestmourDrLA..OL 2-4329
Grushkin Rebecca 51RoseAvVen..396-0270
Grusine Irving 312 SRexfrdDrBH..CR 1-3307
Gruska Elias 9718KirksdeRdLA..VE 9-1838
Gruska Tobi 9718KirksdeRdLA..VE 9-1838
Gruskoff Michael 1232 AltaLomaRdLA..OL 2-4499
Gruskoff Suzanne 1207 SHoltLA..657-2653
Grusky Oscar 7830YarmthRsda..DI 2-0331
Grust R M 2607A[illegible]entLA..VE 8-5311
Grussendorf Leon 6084½WoodlawnAvVen..EX 6-9804
Gruszka Herman 264 SCrescntDrBH..271-7843
Gruszka Marilyn M 264 SCrescntDrBH..271-7843
Grutman David S 9922 StellbarPlLA..UP 0-3592
Grutman Gail 9922 StellbarPlLA..VE 9-1043
Gruts W J 454 SRexfrdDrBH..275-4064
Grutz Herman P 2668ColbyAvWLA..GR 7-5317
Gruver Fred C 11628MontanaWLA..GR 3-2138
Gruver N F 425 26thSM..EX 3-5174
Gruver Wm R II 10705RoseLA..839-3885
Gruwell Isabel A 205NOakhrstDrBH..CR 5-7225
Gruzewski Charlotte 1537CentinelaSM..394-2911
Gryglaszewska B A MD
10757MolonyRdCC..VE 9-6439
Grynbaum Chas J 1442 SWoostrLA..652-3808
Grzybowski Carl J 819AngelusPlVen..EX 9-6964
Grzesiak Edw 1304WellesleyAvWLA..473-7198
Gschwind Vilas 3205NebrSM..EX 5-3983
Guajardo Angel 2826SantmAvVen..EX 9-4337
Guajardo Arthur 1824 19thSM..394-4489
Guajardo Ben 625BdwaySM..EX 4-1932
Guajardo F 1719CentinelaAvSM..EX 4-2256
Guajardo Manuel 12104BeatriceAvCC..397-4105
Gualcazar Fernando 1306MarineSM..EX 6-0357
Gualtieri Jas 9715VeniceLA..839-2194

Guercio Jack 1507BaySM..EX 6-1798
Guercio Ronald P 2628½ 3rdOP..EX 9-8485
Guerena Chas G 1729BriotAvWLA..478-2939
Guerguy Claude 11714MayfldAvWLA..478-6477
Guericio Sally E 12018½GoshnAvWLA..478-4070
Guerin C F 3403 OceanFrntVen..EX 6-0992
Guerin Geo P 17125 SunsetBntwd..GR 2-1533
Guerin Harold D DDS 9057NemoLA..CR 1-1211
Guerin Henry E 103ESpartaWySM..EX 4-7743
Guerin Zoe W 2184GuthrieDrLA..VE 9-7858
Guernard Internatl Coiffure & Beauty Arts
156 SBevrlyDrBH..CR 5-8677
Guernsey E B Cmdr 414 14thSM..GL 1-1235
Guernsey Frank A 2005BWPacCstHwyMibu..456-2482
Guernsey Frank A 269[illegible]SeaVistaDrMibu..457-2657
Guernsey Roscoe Jr 740 20thSM..395-6807
Guerra Jesse 1553SaltairAvWLA..478-8620
Guerra Leo N 11651TennesseeWLA..GR 9-1857
Guerra Ralph C 8314[illegible]tolaTerPDRey..398-6837
Guerrere Louis 746½WashnVen..EX 8-2644
Guerrero Adolph 2450PenmrAvVen..EX 8-8346
Guerrero Alfred 2103 5thSM..396-6676
Guerrero Ascencion 4315[illegible]rumAveMV..EX 1-3380
Guerrero Frank 506WestmnstrAvVen..392-2354
Guerrero Inez 11277WoolfrdCC..EX 1-6871
Guerrero Josephine R 2306 5thSM..399-8208
Guerrero Leonard R 2501PicoSM..393-1646
Guerrero Marguerite 194½Ave33 Ven..EX 9-5603
Guerrero Pete 3050DelawareSM..395-8640
Guerrero R 11906LindbladeLA..398-7936
Guerrettaz Neoma 1328B YaleSM..EX 4-6549
Guerro Jesse 4234BeethovnMV..391-0154
Guertin Henry A 3476RosewdAveMV..391-8063
Guess Ivie 205HorizmAvVen..EX 6-4451
Guess Robt C 4716CoolidgeAvLA..398-0398
Guest Aerovias Mexico..OL 2-8600
Guest Aerovias Mexico 607W6thLA..OL 5-8600
Guest Ethel 1315CentSM..EX 6-6627
Guest G T 4349ChaseAvLA..EX 7-5177
Guest H Rainford Dr 429NPalmDrBH..CR 1-8496
Guest Harold E 8705KingsleyLA..OL 2-6549
Guest Informant 9270[illegible]aCienegaLA..OL 5-7520
Guest Jas 1316 12thSM..395-3450
Guest Jos 909Ariz[illegible]SM..EX 4-6550
Guest Leo 4134KenynAvVen..EX 7-6729
Guest Parking Unlimited..839-7158
Guest Richard M 11[illegible] 2ndSM..EX 4-2367
Guest Thos O 818[illegible]naGrnWyWLA..GR 9-0486

FIGURE 3.

5 September 1963

9

TM-1456/008/00

As indicated earlier, we perform three major operations on files under the general heading of "update." We insert; e.g., if we wanted to put Sally Gruber in the Gruber's, we'd make an insert in the list of Grubers with Sally's name. We also make changes; e.g., if Commander Guernsey changed his telephone number and moved to Malibu, we would use his existing entry and change the data field associated with it. And, we can make deletions. If Nicholas Grunt gave up dentistry and moved away, we could then take Nicolas Grunt's entry out and delete him from the files. Furthermore, we can steal a bit from those who have gone before us and make use of the search delimiters--the names Grossman-Guild at the upper right of the page--that are conveniently provided.

If you had a manual, 3-drawer file with a girl to access it, the alphabetic labels on the front of the card drawers would be search delimiters. These labels would allow you to split the file. In the case of the telephone book, the search delimiter is a page high-low (here, Grossman and Guild) that tell you, "Don't start your search unless what you're looking for is greater than Grossman and less than Guild."

We have stolen this same technique on the magnetic tape putting tape high-lows in the tape label (on the front of the tape). That way I can make sure that the operators hung the correct tape out of a 50-tape file. If we had a file on disc storage, we would keep an inventory of the cylinders of the disc either in primary memory or on one of the disc tracks, using this inventory to avoid unnecessary positioning of the heads (a potential waste of time).

Figure 4 shows a convenient form of "Gantt" chart, which I find valuable in discussing input-output balance. Reading from the top down, Input T_i is shown as the first series of horizontal bars; Compute, T_c is the middle series; Output T_o is the lower series. Time is the independent X axis. When we start a process, we have nothing to do with the computer until we get our input read in. There's no output available so both the computer and the output channel lie idle.

After we lead the entire input for Item 1, we may then do the computations for Block 1. We are then ready to output Block 1. The process continues cyclicly.

A two-channel buffered computer, is illustrated on this special Gantt chart, permitting two processes simultaneously; e.g., Item 2 is read by the input unit while Item 1 is being processed and is output by the output channel while Item 3 is being processed.

The maximum throughput for a computer system such as this one (the technique, incidentally, is general--you can apply it in both the macro and the micro cases) is in the case of a minimum unexploited resource. The processing for one cycle is seven and one-half time units. You'll see that we have one time unit available on the input channel, since the processing is three units and

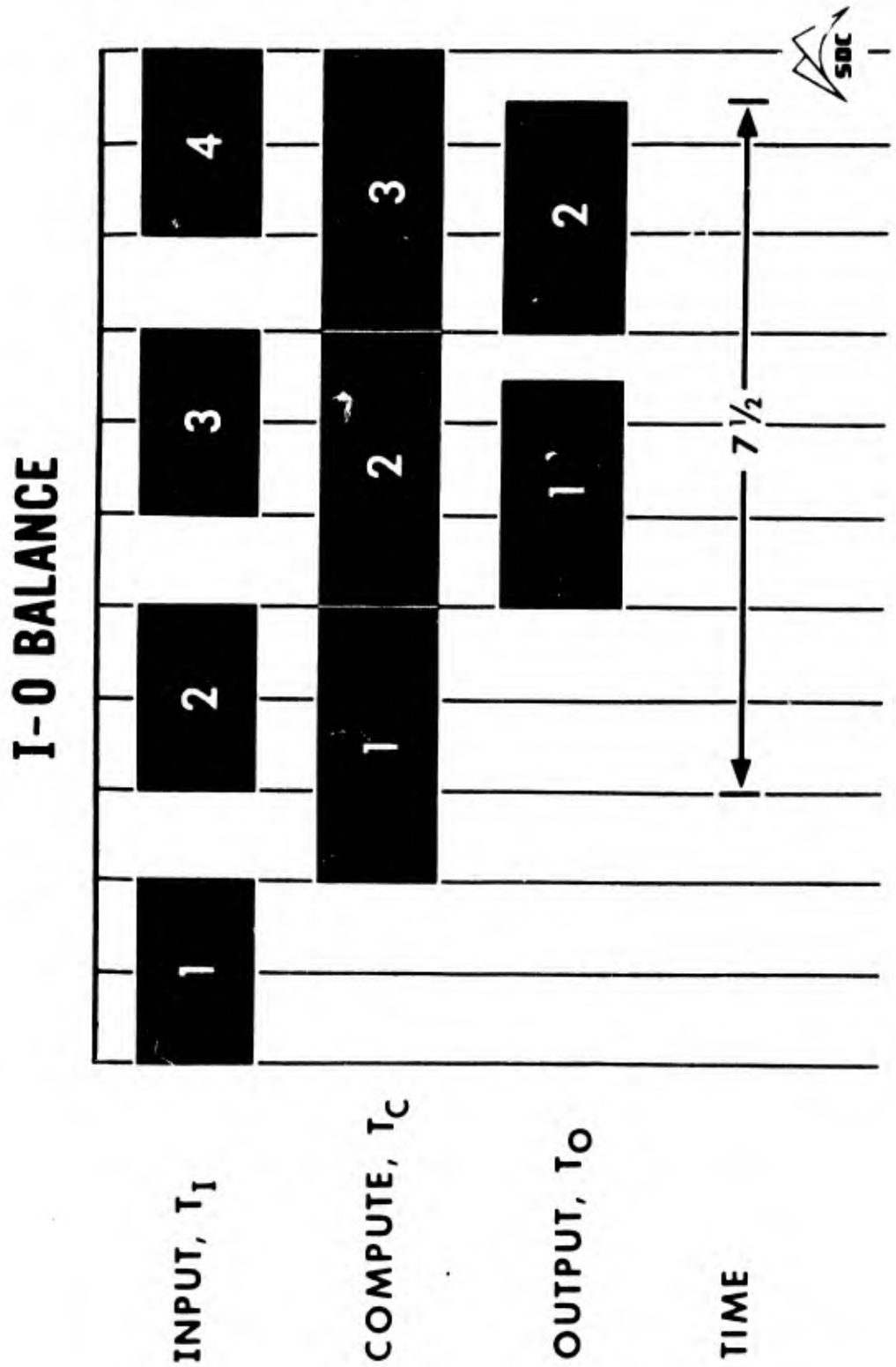

I-O BALANCE
INPUT, T_I
COMPUTE, T_C
OUTPUT, T_O
TIME
1
2
3
4
1
2
3
1'
2
7½
SDC

FIGURE 4.

the input is only two units. The output wastes one-half unit because it's only two and one-half units and the compute is three units. If you can get your job done and reduce the amount of waste space on a chart like this, you are approaching maximum throughput for the hardware you've got installed. It's an extremely powerful technique, one that I am now using for another client in determining the functional allocations between the 7040, a 7094, and shared files.

The cycle time is the processing time--the seven and one-half units shown. Draw one of these charts to determine the natural balance of the operation, and then get a good independent review. (I work with somebody who's capable of saying, "Patrick, you're all wet," and I don't get a good review if there is too big a discrepancy between my work and his check.) After this independent review, we can try to change the functional allocations to bring the chart into balance and raise the efficiency of the operation.

You say, "How can I bring functions into balance--this is a natural rhythm!" Fortunately, it's not. A change in the length of the average entry you read in and write out through recoding and packing can make a sufficiently major change in the I/O balance of a process to justify such an exercise. If you finally reach the limit of such a change--you push your computer to the wall and you still can't get the throughput you want--you must buy some more or different hardware.

The equation in Figure 5 says that the time for the compute should be greater than or equal to the time of input. There is, of course, a similar equation for the output side. You can look at the time for input and (reading from left to right on the lower equation): the time of input is equal to the time for positioning (the servo I mentioned earlier) plus the time for latency (the rotational speed I mentioned earlier); plus a product of the rate of transfer from the device to the main frame.times the length of the record. The only one of these functions not subject to software manipulation is the rate of transfer--R_{tr}--the rate set when you buy the hardware. The others are all either software functions or combination functions affected by both hardware and software. By intelligent program design, you can either materially increase or materially decrease the throughput of the shop. Similar equations written for the output side are just as valuable for analyzing the process.

There are two types of files, as shown on the upper half of Figure 6. The Air Force prefers the combined file in which, eventually, all of the information on each Air Force officer will be gathered and put in one entry under that officer's serial number. There's at least one other way to play the game. This is the concept of the split file indicated by the small Venn diagram on the right.

MACRO TIMING

$$T_C \geq T_I$$

$$T_I = T_P + T_L + R_{TR}(L_R)$$

FIGURE 5.

FIGURE 6.

With this choice: combine the file with everything under one entry name, or split it apart into three separate files (with some overlap, I am willing to admit), the average record length may be adjusted. The advantage of the combined file is, of course, that only one update program and only one lookup program are required. Its disadvantage, obviously, is the length of the record.

In the split file, let's say that File A is telephone directory information and personnel information, File B is a payroll file, and File C contains interests and hobbies. So, to find a man who is a Navy commander, who has some computer experience, who knows something about music, and who reads but does not necessarily speak Russian, I would need information from File C to make a selection. (These are very valuable ways for controlling the record lengths.) But you ask, "How do I know when to do this?"

We do a transaction-frequency analysis, as shown in the lower half of Figure 6. We first list all the transactions--Transaction Class A, Transaction Class B, and Transactions Class C_i, to infinity; determine what transactions this file usually supports; and do a frequency analysis, by class, of the transactions supported, checking to see if there are any others that are significant. In the case illustrated in Figure 6, Transaction Type A occurs with great frequency, and Type B with slightly less frequency. The set of C's is significant as a set, but not as individual transactions; thus, it's clear that you might as well combine them because it's not worth keeping a separate file for each.

In this case, you split the file three ways: Types A, B, and C. Separate data fields for processing Types A and B will result in short record lengths, and if the C's are combined, they can be processed whenever there is spare machine time, since their volume is small.

I detected, this morning, the fact that maybe some of us were talking about the same things, using different nomenclature. This may be from our different backgrounds. As shown in Figure 7, there are alternate strategies available for designing a computerized system for large files. In the civil sector, your concept is minimum dollars. Cum dollars means that you accumulate your expenditures for this function, in an infinite running sum. The mathematician is interested in the minimum of the running integral of dollars over time. In the civil sector, you allow step functions--we call these capital expenditures. You will allow them anytime payoff can be predicted throughout the life of the device. You work very hard to minimize the average case to reduce total cost. You give preferential treatment to the frequent cases. You place them on the front of the tape, or put them in the middle of the servo swing.

In the military case--the military real-time case--you don't have a spare millisecond. You are interested in the "min of the max's." The left hand ordinate is the rate of the effort vs. time. In a real-time case, you must not run late. Often, the computer equipment is dedicated to the task. You can allow higher average processing if it lowers the max. You try to get ready

ALTERNATE STRATEGIES
CIVIL
MIN $
MILITARY
MIN MAX
ULT. CAP.
TIME
TIME
CUM. $
dE/dt

FIGURE 7.

so that the minute you ask the question you can respond right now. You may do that "getting ready" several times, and maybe the question is never asked, but you can answer that question right now if it is asked, and that's important in the military case.

You must stay below that ultimate capacity line or you're dead--perhaps even literally. In the military environment, where you're minimizing the maximum, you can devote tremendous energies to getting ready and improving your position. Your maximum work force almost equals your minimum work force, and is also available all the time. You work to increase your ultimate capacity, or to decrease the effort required (raising the efficiency) or both, exploiting your resources to get ready for the eventuality.

I had to ask the question in Figure 8 bluntly because, from time to time, even my thinking gets mixed up on the question of batch processing vs. on-line processing. The answer is very clear and crisp: we batch because it's efficient I don't say it will always be more efficient than on-line, but with today's hardware, it is. You can reduce the total effort, both man and machine, by batching. If the environment will tolerate delay (and that's a big "if"), set the batch size to where you use up all the delay they'll tolerate, and run at the maximum efficiency. That's what happens, for example, when you ask for an airline reservation. The girl says, "Sorry, the reservation's desk is busy." They're making use of your tolerance of delay by putting a backlog on the girls that are answering the queries. That raises the efficiency of those girls: they've always got an outstanding query and are running at full capacity. You wait, because if you hang up, you don't get your information, and so you batch, to raise the efficiency

Even in a real-time environment, if you're over-loaded -you have a backlog--you may want to switch your mode of operations and batch to gain efficiency and to catch up. One last point to remember:a real-time, on-line operation is, in present state-of-the-art, running with batches of one.

Now, let's look at some details Here in Figure 9 is the micro-timing of tape (the oblique line) and of magnetic disc (the three parallel lines). The lower parallel line is the time required to input a block from magnetic disc if no servo motion is required (if the heads are already positioned where you want them). The middle parallel line is where minimum servo movement--just a one-track step--is required. The upper parallel line is where maximum movement is required.

The intersections are conveniently marked by vertical lines labeled B, C, and D. The independent variable is block length. Just for fun, let me give you those four ordinates, reading from the bottom to the top. Tape above point A is 7 milliseconds; no movement on the 1301 disc (B) is 17 milliseconds; minimum movement on the disc (C) is 67 milliseconds; and maximum movement on the disc (D) is 197 milliseconds. (These include the servo time plus an average latency time, which is one-half of a revolution.) These, then, are the elapsed times

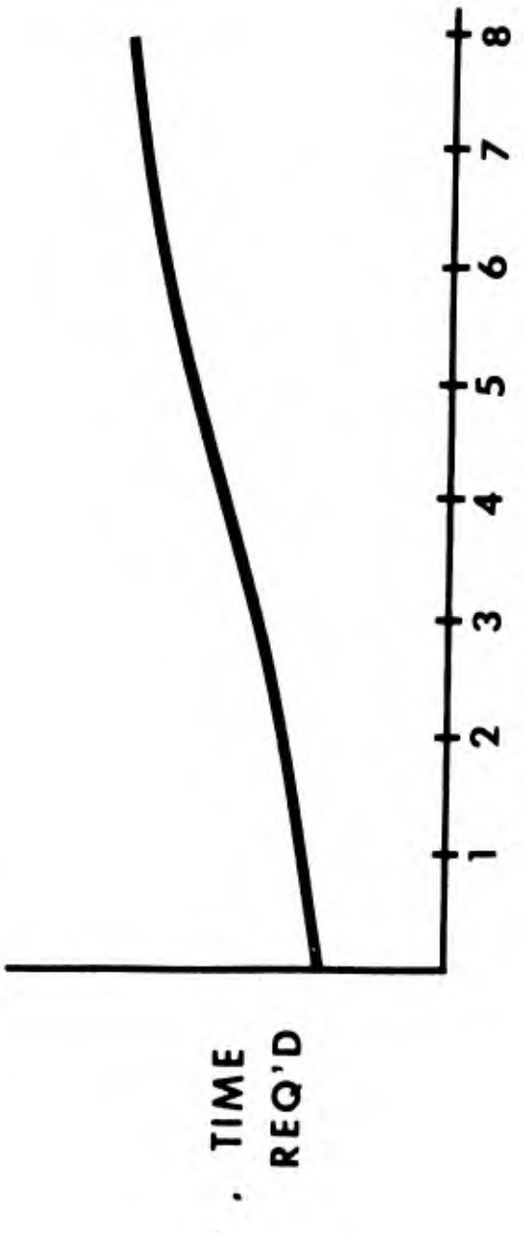


FIGURE 8.

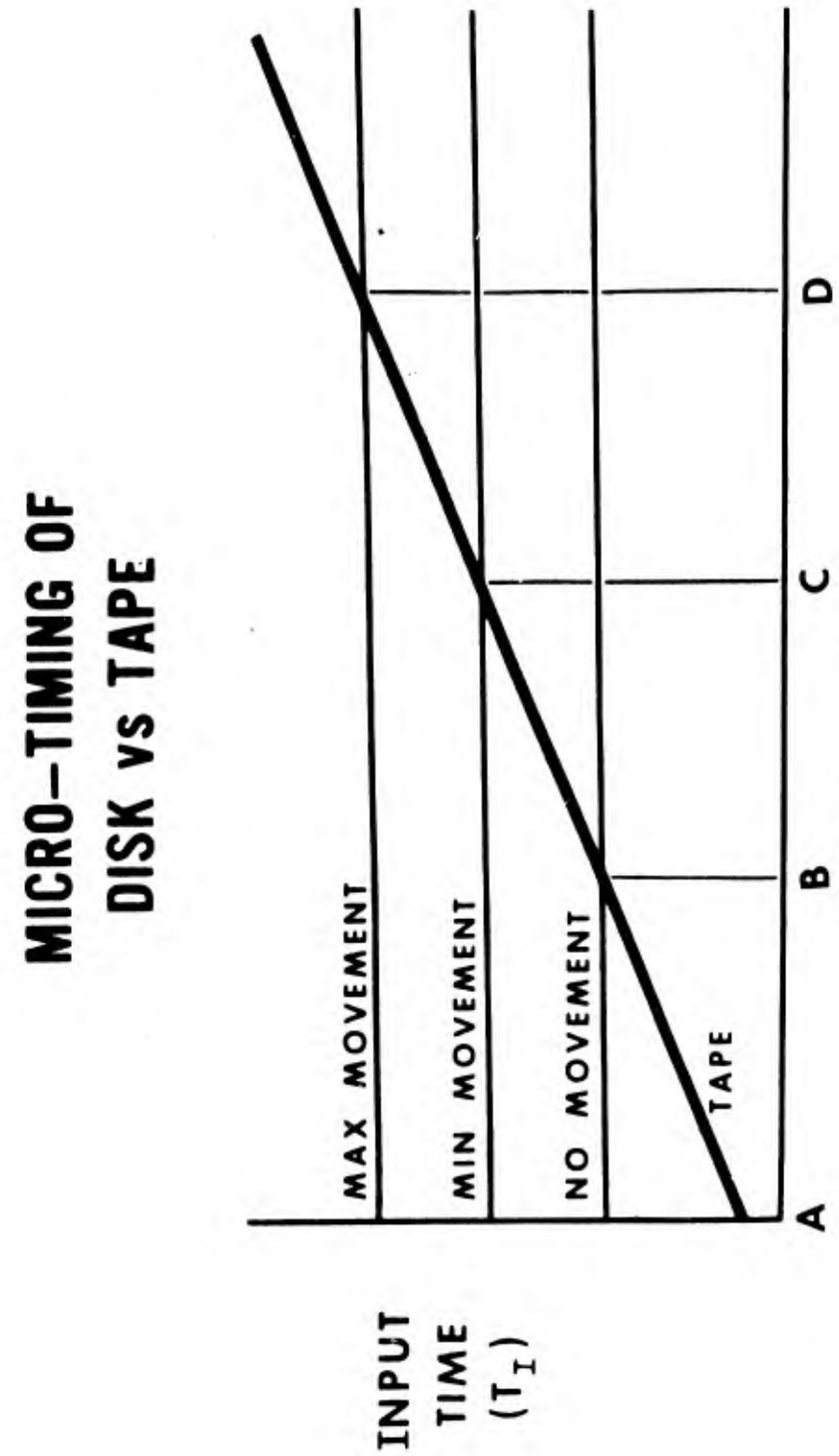

MICRO–TIMING OF
DISK vs TAPE
INPUT
TIME
(T_I)
MAX MOVEMENT
MIN MOVEMENT
NO MOVEMENT
TAPE
A
B
C
D
BLOCK LENGTH

FIGURE 9.

required from the time you give your first command to the unit to the time the first data word comes in. Clearly, I'm being unfair in comparing tape against disc, although some people do this, and for that reason, I think the chart is valuable.

QUESTION: Do those times cover information transfer?

ANSWER: Those numbers I just gave have no information transmitted. This is just the time for the first word to come in, the elapsed time from the time you give the command to the time the first word of memory is changed. Block length is shown on the horizontal. If your block length is in the area from A to B, the tape will transfer the block of information to core memory faster than the disc will--provided there's no arm movement on the disc. If it is from A to C, the tape is faster than disc if minimum motion of the disc arm is required. For block lengths in the range A to D, the tape is faster than disc if maximum disc arm motion is required.

Let me restate those conversely. Above D, disc is always better; above C, disc is better if only minimum arm motion is required; above B, disc is better if, by some happy circumstances or excellent planning, the disc arms were already positioned before you gave the command.

Let me run across the block lengths for you. I recommend that you faster your seat belt--this is state-of-the-art, current in the field. For B, the crossover is 2,040 characters (340 words); for C, 12,280 characters (2,046 words); and for D (remember, above D disc is always better), 38,900 characters (64 hundred 36-bit words). If you are talking about a couple of buffers going in and a couple of buffers going out, and the program alternates between them, the disc is always better, if you use up 25,600 words just on buffers!

QUESTION: That implies that the tape is partly pre-positioned?

ANSWER: Yes sir. That is correct. That implies you're processing a serial file, and you're comparing tape vs. disc for a serial file.

Note that this comparison is not quite true for writing because the dual head on the tape checks while writing, in the same period of time indicated by the oblique line. The disc requires a second rotation to verify, and the crossovers are way off the end of the slide.

So this says, if you are processing a serial file, or conversely, if you are processing a file that can be made serial, tape is much better than disc, present state-of-the-art. By the same token, if you are processing a file that is random--you don't have any idea what you are going to need next--there is no preferred order; you can't sort it on one key, or only a few keys. If all of the frequencies of transactions in Figure 6 are equal, and none are significant, then you've got random-access process, and you must pay the penalties. There's no way out of it.

CLOSING

The purpose of this talk is to scare you a little bit--to indicate that although things are nice, they're not quite as nice as we'd like them to be. Figure 10 was taken from a recent paper given at the Disc File Symposium by Herman Hess of Informatics.[1] He shows the time to sort 60,000 16-word records, using the IBM 1301. Three methods are timed: random storage, simulated tapes, and actual tapes. For random storage, he takes the key that's associated with this data element, mixes up the characters in that key and does the famous "remapping" to find where to store it on the disc. That's what he calls random storage. He's utilizing the random-access features of the device by computing an address from the data elements in the key, storing it at that point, and remembering where he put it.

With simulated tape, he uses the 1301 to simulate 20 magnetic tapes, which are used in the same way as are real magnetic tapes. He says, "Now 1301, you are tape Number 1." He commands the servo to track 1 and it sits there and writes on that cylinder until it is full, then it inches over and writes on the next cylinder, etc. There are 250 cylinders on the 1301 disc. To simulate 20 tapes with them, he makes up 20 logical serial files on 12 1/2 cylinders per each.

In the last case he actually sorted with sixteen 729 Mod IV tapes--actual timing. You'll notice that phase I is the same (that's reading in the basic input file at full read speed and getting your hands on the information to be sorted). Phase II is the internal sort itself--arrangement of items in sorted strings. Phase III is the merge. The total time is the next column.

For sorting with random storage the total time is 93 minutes, and, using a 1301 disc file in this very primitive way, the speed ratio (the rightmost column), to tape time is 6.3 times (630%). (Tape is lower by a factor of 6.) For the simulated tapes, where he's minimizing servo motion, he can shade the regular sort time a little bit, to a ratio of .88 (13.6 minutes). With actual tapes (these are all done on the 7090) 14.78 minutes were required, and that's his reference point.

TIME REQUIRED TO SORT 60,000 16-WORD RECORDS USING THE IBM 1301

METHOD	PHASE I	PHASE II	PHASE III	TOTAL	RATIO OF TIME TO TAPE SORT
	(TIME IN MINUTES)				
RANDOM STORAGE	1.78	1.20	90.10	93.08	6.30
SIMULATED TAPES	1.78	9.34	1.94	13.06	.88
ACTUAL TAPES	1.78	10.04	2.96	14.78	1.00

FIGURE 10.

QUESTION: But who would use random storage on a disc file? I don't understand this at all.

ANSWER: This is kind of--at least in his mind--the way you might think of using disc.

QUESTION: Only if you hadn't seen it intuitively would you go to this extent to find out you couldn't do this on disc?

ANSWER: He probably wasn't as smart as you are.

QUESTION: (I appreciate the remark) But, it's well known in the field.

ANSWER: The reason he did this for the same reason I'm using Figure--to shock! He picked an application that most people thought they understood fairly well (in actuality, people don't understand sorting very well) and people would normally say, "I have a 'random' access device. I'll use it as a random-access device." You remember the plot in Figure 9--if you're writing very short blocks, tape is much better than disc. In the top case here, he's writing very short blocks. Most of the time is spent waiting. The hardware that's provided with the 1301 disc is a single-channel and two servos; there is a minimum of overlap; there is no simultaneity of transfer in the device at all.

QUESTION: I think that approach is unreasonable.

ANSWER: Well, it's unreasonable except that you must recognize that until you understand the process, there are some magnificant conclusions that you can jump to. If you don't lay your data base on the disc file, to minimize the servo arm motion required, quite possibly you'll run longer than you have been running with tape. It can be significant.

In conclusion, performance probably won't come easily. The time penalties that we saw in Figure 10 are astounding if you do it as stupidly as possible. Performance requires careful attention by a well-founded machine man. We do not yet know how to build this kind of "smart" into our compilers. Therefore, through software, a lot of attention, and some hand-written machine code, we must adapt our equipment to the tasks at hand.