Massachusetts Institute of Technology
Information Processing Center
Cambridge, Massachusetts 02139

April 27, 1971

To: Multics Administrative Distribution

From: T. H. Van Vleck

Subj: Disk

This memo lists a bunch of thoughts about disk storage. It is more or less inspired by John Gintell's memo on "Secondary Storage Planning for Multics."

1. Some quick remarks on performance:
   a) Think we should get something written down about how performance on Multics depends on various disk issues. Not a thesis, not for publication -- just to inform ourselves.
   b) Maybe more study of performance.
   c) What, for instance, might 2314 seek-optimization buy us? Best case? Worst case?

2. Concerning capacity:
   a) Raising the cost of disk to force users off is the worst solution. (There are many non-solutions, such as a delete-by-random program.) Price changes by the IPC always raise a howl; and many users are telling us that they want and can afford disk during next fiscal year.
   b) Charging for quota, etc., only work anyway if the programmer is price-conscious. This can be done by making the price high enough. How else can we "get his attention?" Record-quota overflow, system crash, delete-by-date? These are pretty severe.
   c) Delete by date might be a good idea. We need a "removed" mode bit in the branch, which causes a fault if one references the segment.
   d) What about demountable disk packs?
      1) keep free storage map on pack
      2) primitive attaches pack to any spot in file system as if it were a directory

3) movequota won't cross upward from or downward to the "pack directory"

4) When pack is dismounted a trick branch with "removed" mode is set up. Includes date last mounted, pack label, etc.

5) Dumper no dump. Search rules ignore (?) if pack not up.

6) Many users share one facility if they are separated in time. For example, could dismount >ldd during certain hours.

e) Would like to hear more about tape archive. Sounds great.

f) Secondary (i.e. deleteable) mode sounds great.

g) We already do oversell disk, and hope it won't crash. But right now, the system doesn't help much on such a crash: operations has no idea of what to do if we run out of disk other than to call Roger. The reloader, if run in non-trim mode, would probably fill the disk, or come pretty close.... Perhaps we need a reloader with some discrimination on dates -- reloads and does sort of delete-by-date, then can fix it up later by using different args.

h) I think we should get a second 2314. I think we can sell it, to ordinary users plus MacAIMS and the Cambridge Project. It'd be nice if we could lower the price. Some big disk users are complaining already.

3. Charging for quota can be done by one of two methods:

(1) Change the file system to keep a quota integrator just like the current usage integrator in the branch.

Since this changes the branch declaration, it is a nontrivial change. Probably we should do it on the next go-around on branch structure.

(2) Simulate the above by doing all movequotas between >udd and the project directories through a special program which keeps records.

The salvager destroys quota when it zaps a scrambled directory. The reloader sometimes generates quota. Both of these actions would bother an excessively simple-minded quota-keeping program.

Furthermore, detecting these situations soon after they occur is so desirable (it costs money) that the program should be run at least once a day. This could be made part of the "crank", when absentee gets done.

The question is, will economic incentives operate to discourage disk quota use when they don't seem to affect record use?

4. The real need in the quota area is for some better "temporary quota" mechanism. The process directory and its quota allow users to use more storage than their record quotas permit, for the time that they are logged in. The access on the process directory makes it unusable for some applications, and the fact that it lasts only during a session makes other applications difficult.

   Here is a proposal for disk charging which would be possible with no more extensive change to the file system than that for unused-quota metering. This proposal is submitted as a discussion-starter rather than a complete plan. Briefly, disk charge would consist of

   A. A minimum charge, based on the user's quota.

   plus, B. A surcharge for usage above quota, to some maximum. This charge might have a step in it.

   plus, C. A surcharge for unused quota records in excess of half the quota.

   For example, suppose that the charges were as follows:

   A. $1 per page per month for each quota record

   B. $1 per page per month up to twice quota
      $3 per page per month up to four times quota

   C. $1 per page per month for unused quota exceeding half

   Then the charges for a user who had a quota of 100 would appear as in Figure 1.

   The following remarks apply:

   a) The flat region from 50-100 records makes it likely that a delete won't cost the user.

   b) The rise below 50 records encourages the user with small space requirements to have a small quota.

   c) Above his quota, the user pays linearly, up to twice quota. This area is meant for "working" data.

   d) The region from twice to four times quota is expensive, but usable. It is meant for overnight storage, or even more temporary stuff. It should keep users from blowing up due to "rqover" errors.

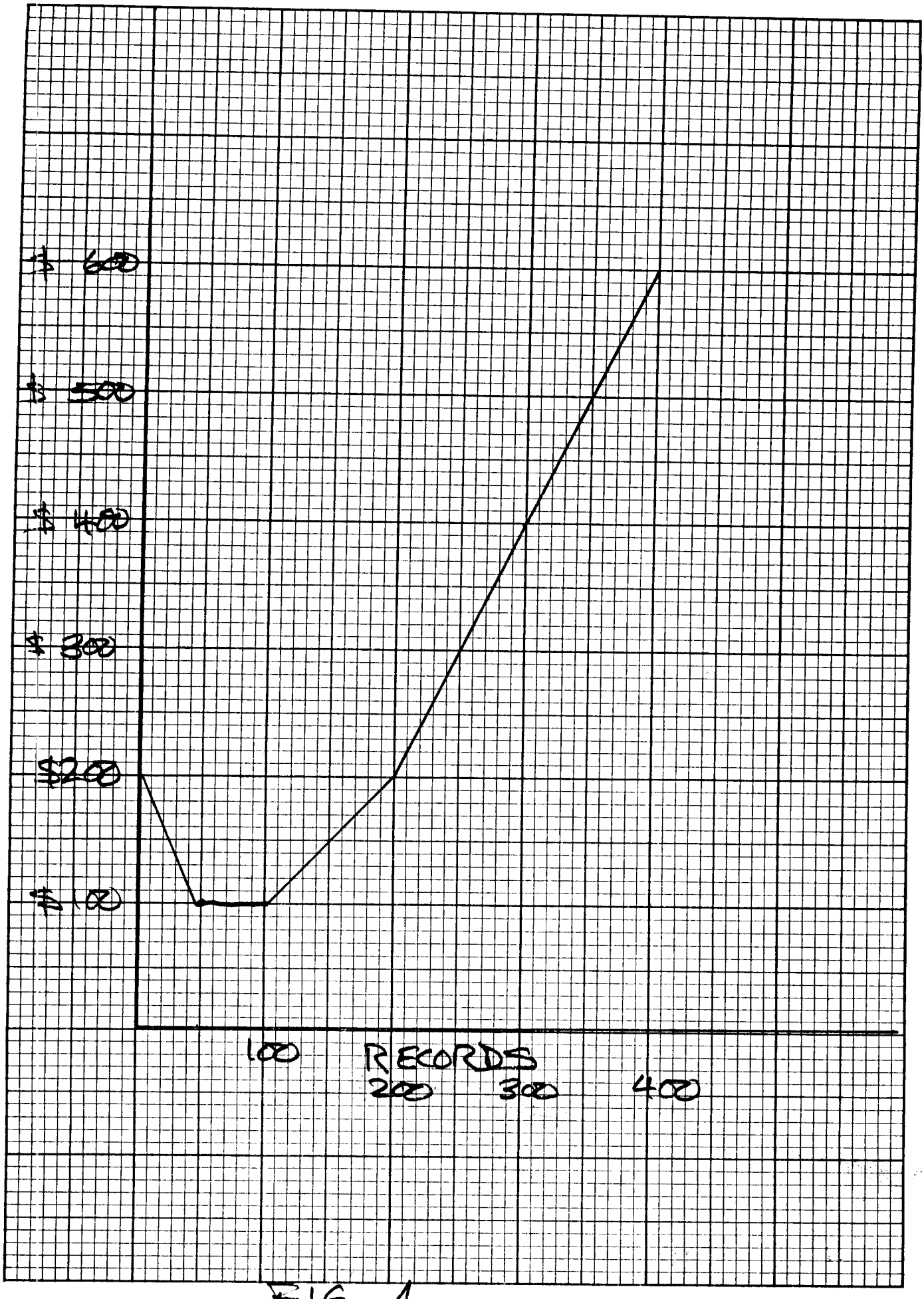

FIG. 1

e) Perhaps the curves should have absolute parameters, not relative to quota; e. g., maybe it should be 64K plus or minus.

f) This scheme enables us to predict pretty well in the one-quota-per-user case. In the many-users-per-quota case, things are a little tricky. Any user of the quota can push the group into the high-price area. Perhaps this argues for parameters being functions of quota, so bigger directories are "safer." The scheme certainly does make the subdividing of quota more possible, since space for listings need no longer be dedicated to each user.

g) Response to requests for increases or decreases is now dollar-valuable to the user. An automatic scheme should be instituted, perhaps handled by the answering service process.

5. Setting up administrative groupings which sub-allocate disk to projects, like CTSS did, sounds like a good idea. This could be done either on paper or by changing the Multics tree structure.

To do it in the file system we would need some tools:

a) re-linker
b) tree-mover
c) administrator program (at least usage report)

None of these are hard. To keep "mail," etc. going, a lot of links would have to be put in UDD also. (This might eliminate some of the need for relinking.) But then we would have to find administrators for each project, train these people, and keep them interested. This was where we broke down on CTSS.