FROM: C. Garman

DATE: October 25, 1965

SUBJ: Proposal for Paging Disk Track Usage Tables

Summary

Certain advantages in file-system operation can be gained by separating the disk usage table into pages.

Advantages

1. reduced core storage for tables.
2. exact accounting for record usage (1 bit/record).
3. future expansion of hardware configuration.
4. shortened time to update usage information (re-write only current page).

Disadvantages

1. extra programming required for implementation.
2. non-optimum record assignment (increased inter-record seek time).
3. extra overhead taken from users during page-swapping (more-or-less invisible, depending on degree of user/supervisor multi-programming).
4. Track usage analysis programs would no longer work from information in supervisor core; must have ability to read records from M.F.D.

Background

The current disk track management module (TMAN2C) uses 1 bit in its tables to represent each two records; for the current MAC implementation of one 1302 with 4 modules, (80000 records) this amounts to 40000 bits, or 1112_{10} words of dedicated tables in supervisor memory. The current algorithm also permanently loses 10 percent of available disk storage from files which contain an odd number of records, because of the 2 records/bit scheme (based on analysis at Project MAC, mid-October, 1965, done by Noel Morris.)

To add another 1302 disk module to the available resources would force doubling the length of this table, or else further modifying the module, to let each bit represent 4 records; the first alternative is difficult due to current cramped conditions in the supervisor, the second would result in the loss of approximately 27 percent of the 160,000 records of storage, by extrapolation from current statistics. This paper presents a method for reducing the size of the tables resident in supervisor core, their length remaining (almost) constant through changes of configuration, simultaneously regaining 1-for-1 correspondence between bits in table and records used/available for secondary storage.

Method

TMAN2C (the current track management module) initializes by reading two files (secondary name '(FILE)') from the M.F.D. : DRUMUT, the usage table for 7320 drums; and DISKUT, the usage table for 1302 disk (including a table at the end for "cylinder" usage). These two files are kept permanently open in the file system's internal active-file-status-table (AFST), and are re-written on calls to UPDATE and IOFINI. For 40000 bits, re-writing takes:

avg. rotational delay	1/2 rev.
re-write of DRUMUT (1rec)	2 rev.
missed latency from calls to BUCM	1 rev.
re-write of DISKUT (3 rec)	6 rev.
total	9 1/2 rev. @ 17 ms/rev.
$\cong$ 160 ms.	

To obtain a record, GETTRK is called with one argument: the device number for a new file, or the BCD record address of the last record of the file for additional records; the result is the BCD record address of the record to be written. To return a record, DELTRK is called with the BCD record address of the record being returned. Both GETTRK and DELTRK must be called <u>disabled</u> either implicitly from the "trap-side", or explicitly from the "entry side".

The proposed change would be to split DISKUT into "pages" of 10000 bits (278_{10} words), each page a separate file in the M.F.D. Records for each module would be distributed evenly among the pages. For a single 1302, this would result in 8 pages, DISK00 → DISK07. DRUMUT would remain separate, as at present; one additional file, PAGEUT, would be required to indicate pages with available bits.

Operation

GETTRK would operate substantially as usual until a page is exhausted; after assigning the last available record for the page, entries would be made in the file system's internal queues and AFST, to re-write the current page and read in a page containing available records. These operations would be invisible to the user, but would pre-empt all disk/drum I/O after completion of the current record.

DELTRK would operate normally for a record within the current page; for records in a different page it would hold back the record address given, initiate the re-write - read cycle, and return; manipulation of the table would wait until the next call to either routine after completion of the I/O.

Error Checking

Since the I/O to cover page manipulation is invoked during the time traps are disabled, and is being performed by the same strategy module which "track management" services, any I/O errors occurring during paging will be considered "latent errors" by TMAN and will not be noticed until the next call to either GETTRK or DELTRK.

Timing and Efficiency

Once a re-write - read cycle has started, further user I/O is blocked for ~ 60 ms (1/2 drum revolution avg rotational delay, 2 rev re-write, 1 rev readin). Pathological cases could conceivably occur with oscillation between pages as records are assigned and released; these should be infrequent.

No consideration has been given to division of records for a module among the pages; for reasons of optimization of disk seek time, divisions should be made between cylinders or groups of cylinders.

Programming Changes

File co-ordinator: room for one extra permanent AF entry in file system's AFST. (trivial)

Disk/drum strategy: code moved from trap side "setup" to disabled section of entry side when writing first record of a file. (minor) 1

Track management: additional code for manipulating AF entries and inserting queue beads; modifications to bit position algorithms; proper handling of released records. (major)

Disk/drum setup: creation of proper files for empty configuration. (minor)

Salvages: read all pages into core; modify bit index calculations. (medium)

Known Problems, Proposed Solutions

1) request for record address before completion of page swap (due to set-up of next I/O overlapped with initiation of current I/O):

a) synchronization flag to block write-requests until known completion of page I/O.

   b) single record address/page kept in table, refilled on first call after page I/O complete (buffering)

2. Two requests/trap (occurs only on first record of file): code moved from trap side to disabled portion of entry side.

Points for Discussion

1. Maintain two pages in core
   a) efficiency in releasing records,
   b) resolution of certain problems above.

2. Algorithm for swapping pages at some point short of complete exhaustion.

3. 250 cylinders/module is not evenly divisible by 8:
   a) scheme for allocating cylinders to pages, or
   b) split each cylinder among all pages (need further information about average and median lengths of files).

---

1. Page swapping time can be cut to $\sim$ 45 ms by removing one rotation of drum for re-write; full explanation requires khowledge of re-write implementation in Disk/Drum Strategy.