Strategies for Re-packing Discs and Image Data Bases

by
Michael Hornsby

Hewlett-Packard
4501 Erskine Road
Cincinnati, Ohio 45242

How do you know when its appropriate to reload your system or re-organize your data bases? Unfortunately, the developers of MPE and Image did not provide a user friendly yellow and red alert message system for this purpose. The answer to this question is highly dependent on the size, number and type of applications resident on the system in question. The purpose of this presentation is to provide insight into the planning and execution of activities required to maintain an optimum performance level on your system, while utilizing available disc space as efficiently as possible.

Topics to be discussed in this presentation:

How to review your system for over configuration of directory and virtual memory disc space.

Planning for disc space growth requirements.

Archiving files and IMAGE data bases on a periodic basis to remove files that aren't needed for online access.

Disc compression and reloading strategies.

IMAGE master data set disc space requirements and performance tradeoff's.

IMAGE detail data set packing strategies.

Using facilities of Turbo IMAGE to reclaim disc space.

I. In the Beginning

Your HP3000 arrives and your Systems or Customer Engineer installs the operating system and whatever subsystem software products you have purchased on the system.

A. The Initial Configuration

The initial configuration of your system will involve using disc space for the system directory, usually 6,000 to 10,000 sectors, and virtual memory 25,000 to 35,000 sectors per drive. Will your system require more or less than this? Only time will tell. The amount of space configured for the system directory and virtual memory on ldev 1 can only be changed by reloading the system, so these parameters are almost always over configured. More realistic values can be set after the system has been in operation for six to twelve months. A sysdump $null will divulge the amount of space configured and available in the system directory. A run of tuner will display the amount of space configured and the high water mark of space consumed in virtual memory. Disc space can be reclaimed from non ldev 1 virtual memory by down sizing its configured value on a cool start.

B. The HP Initialed Accounting Structure

When the system is installed or updated, the installation process leaves behind accounts, groups and files that you do not need for everyday operation. These not only take up space but also are a security problem, as they are found on all HP3000's. The files in the HPPL@, TELESUP,and SUPPORT accounts and the files in the USL, CREATOR, and DOCUMENT groups of the SYS account can be stored off and purged. The installation process also leaves a group of environment files in the ENV group that are not needed if you do not have an HP2680 or HP2688 laser printer connected to your system. A last note, the install job creates many accounts and groups that may not contain any files. These can and should be deleted to tighten security on your system.

C. Planning for Growth

When the system is new and many millions of sectors of space are available, the last thing anyone is concerned about is planning for the day when you'll be out of disc space. This is the time to implement a monitoring and tracking process for growth. Remember, even if the number of users and applications on your system remain stable, you will eventually run out of disc space if no plan is put in place to manage it! This plan must contain a strategy for naming of accounts, groups and files. If all users logon as USER and all files are located in the PUB group of accounts, it will be very difficult to determine appropriate disc space management procedures.

II. The Mature System

Once a system has been in operation for twelve to twenty-four months the discs will be filled and it will be someone's task to account for this scarce resource. A common solution is to simply purchase another disc drive; however, this strategy is only a short term fix. What follows is a brief lesson in disc drive archaeology.

A. States of Disc Space

Disc space can exist in one of three domains: Configured, System, and Lost. Configured disc space consists of the file directory, Virtual memory, and Spooled input and output files. The System disc space consists of Mpe files and free or available space. And finally Lost disc space are those sectors which have been deallocated from the free space maps but aren't recorded as any of the aforementioned entities.

Configured disc space can be calculated in the following fashion:

1. Directory Size
   Sysdump $null _____

2. Virtual Memory
   Sysdump $null _____

3. Spooled Input
   Showin status _____

4. Spooled Output
   Showout status _____

Total Configured Sectors: _____

System Disc Space can be calculated in the following fashion:

1. Free Sectors
   Run FREE5.PUB.SYS _____

2. Total Allocated Sectors
   Report ?.@ _____

Total System Disc Sectors: _____

Lost Disc space can be calculated as follows:

1. Total Disc Sectors Available:

Drive Model		Formatted Sectors	
#HP7908's	____ *	64,750	_____
#HP7911's	____ *	109,824	_____
#HP7912's	____ *	256,256	_____
#HP7914's	____ *	516,096	_____
#HP7920's	____ *	195,600	_____
#HP7925's	____ *	469,440	_____
#HP7933's	____ *	1,579,916	_____
#HP7935's	____ *	1,579,916	_____
#HP7936's	____ *	1,200,984	_____
#HP7937's	____ *	2,233,752	_____
#HP7941's	____ *	92,928	_____
#HP7945's	____ *	216,832	_____
#HP7957's	____ *	319,095	_____
#HP7958's	____ *	510,552	_____

Total Available Sectors _____

2. Total of Above
   Configured Sectors _____
   System Sectors _____

Total Known Sectors: _____

3. Total Lost Disc Space
   Subtract (2) from (1) _____

B. Wasted disc space

Aside from being Lost disc space can be wasted in many other ways. Poor blocking factors, Editor "K" files, HPWORD WPSPACEF files, NMLG log files, system logfiles, old source files, and above all mis-sized data bases to name a few. An effort should periodically be made to correct these problems, because not only do they waste disc space but they extend the length of backups, and reloads.

Other frequently overlooked options are the compression of source and program files, and variable length records. The contributed library contains utilities for compressing and unpressing files. If you must store source online thess are methods to minimize the impact.

C. Archival

So what can be done about these files that waste space and time backing them up? The solution is simple store them off to tape with a date= option of the STORE command! This little known parameter makes it possible to identify, store and delete files that haven't been accessed since a given date. I have seen this free up as much as a half million sectors on more than one HP3000!

The topic of mis-sized data bases will be discussed shortly, needless to say Image data sets should not be deleted in the above fashion.

III. Compaction Options

Once files have been archived, and the extent of lost disc space has been ascertained, we have several methods available to contend with the fragmentation of the disc space it's self. Disc free space becomes fragmented over a period of time determined by the volatility of builds and purges of files. How do I determine if a fragmentation problem exists? For each disc drive divide the largest free area into the total free area. If this ratio is 1 then all free space is in one chunk, however if this ratio is less than 20% then your largest chunk pales in comparison to the many smaller less useful chunks out on the drive.

A way to determine fragmentation is to issue the following command:

BUILD TAKSPACE;REC=128,1,F,BINARY;DISC=15000,1,1;DEV=1

If this command fails, your system does not have a contiguous chunk of 15,000 sectors available on LDEV 1. Not having 15,000 contiguous sectors on LDEV 1 will present serious problems if you attempt an update of your system. Therefore it is highly recommended that after an accounts reload you build the above file. By purging this file prior to an update, you will not have to subject yourself to the time and effort to create the space required. Purging this file also is a quick fix for the out of disc space all spool queues shut situation.

One other note about updates and LDEV 1. During an update the system SL is read in from tape and placed on the disc at the first contiguous area large enough to hold it. In the other load operation types the system SL is located next to the system directory, since both are highly accessed, this is an attempt to keep the disc head over the same location. Sever performance degradation can be experienced after an update if the system SL happens to be placed far away from the system directory. Disc caching has lessened the impact of this problem ,but it is a good practice to check the placement of the system Sl using listdir5. This could also explain why some sites experience different performance symptoms after updates to new versions of MPE.

To recompact your disc you have two dramatic options, the full reload, and the Vinit compress option, both of which will make your system unavailable for some period of time.

A. Types of Reloads

Reloading the system from a full back up is a time consuming process; however, it provides the best method of reinitializing disc space, in that when complete, the largest free area on each drive will be equal to the total free area on each drive. The reload accounts option is the most frequently used type of reload. This entails a quick reload to get the system up and running,

then a restore to replace all of the user files. The compact option type of a reload can be used on smaller systems with two or three reel full sysdumps. In this type of reload, the system does not become available until the last file is restored from the last reel of tape. And if anything should go wrong, the only option is to start over from the beginning.

B. Vinit Compress

An alternative to reloading the system is the compress option of the Vinit Utility. This will compress a given drive's allocated files in an attempt to coagulate the free chunks. This procedure does not completely achieve the same state in a reload, in that the largest free area may or may not be increased.

C. File Copy Utilities

A free space increase can also be achieved by copying files to other disc drives. There are many utilities available for performing disc to disc copies. However, the same function can be achieved by storing off the desired files or files and then restoring them with a dev= option.

IV. Image Data Bases

On most transaction processing systems, the Image data bases consume the lions share of the available disc space. The following is a brief discussion of how to increase the performance and throughput of a system while recovering wasted disc space as an added bonus.

A. Master Data Sets

Probably the least understood and most maligned topic regarding Image data bases is the master data set, this is normally due to the fact that designers tend to put the cart before the horse and begin the design by specifying all of the possible keys into a detail or by defining stand alone manual masters that substitute for KSAM files. This of course leads to great performance and design problems it that only one unique key is available for direct access to the stand alone master, and all other searches require a serial search of the all the master records.

A different approach to design that leads to greater flexibility is to begin the design by specifying only stand alone detail data sets, and then choosing no more than three paths into any given detail. The utility of the fourth and greater paths usually can not justify the disc space and CPU cycles to maintain them.

1. Sizing Myths

The largest myth concerning master data sets concerns sizing, the thirty percent free space and prime number rules of thumb have had much discussion in the past. It can simply be demonstrated that for many masters these rules of thumb do not apply. Two examples come to mind. Short character keys ie dates tend to generate high percentages of secondaries at low fill levels, due to the sameness of content and low number of bytes to attempt to hash on. On the other end of the scale are masters with capacities in the half million or greater range, here different prime numbers or products of prime numbers can generate poor hashing profiles.

The only way to circumvent these problems is to use a utility to monitor the number and length of secondary chains. My personal preference is DBLOADNG, it not only reports on secondaries but also reports the number of average blocks that have to be read to access one master. Note, the perfect case is when no secondaries occur, and thus only one block need to be processed to access one master entry.

Thus in over sizing certain masters, especially ones that exhibit poor hashing performance, we can trade off disc space for CPU cycles.

2. Combining Automatic Masters

Probably the least understood image procedure is the dbfind intrinsic. This procedure is the heart of the power of image. It is used to set up for a chained read. The parameters of this call reference the detail data set and search item, and from this image knows which master to access. Thus two automatic masters with paths into a given detail can be combined if they have the same data type and size. A good example of this would be an order header detail with order date, ship date and invoice date as automatic masters. The same functionality could be gained with one date master with three paths into the same detail. This not only simplifies the database design but can result in rather substantial disc space savings. Note that this method can also be applied to different data items, for example customer number and invoice number, again if they have the same data type and size definitions.

Surprisingly enough, it turns out that due to the way the dbfind intrinsic works, this is a rather easy change to retroactively implement.

B. Detail Data Sets

As can be gathered from the above discussion I am highly disposed toward detail data sets. Experience has taught me that most Image performance problems can be traced to putting data in master's. The following discussion about detail data sets is the product of years of experience from the school of hard knocks.

1. Picking and Packing the Primary Path

The most important assumption that is made during the design of a data base is the length of an average chain! This value is intimately tied to the application that the data base is apart of, and should be well documented and reviewed during the life of the data base. For example, the number of line items per order, the number of orders per customer, the number of orders per day. If the answer is one, ie the number of customers per customer number, then this path can be said to be perfectly packed because, the number of IO's needed to read the entire chain is always one. However as the number of entries on a given chain rises in respect to the total number of entries in the data set, the usefulness of the path decreases. In fact if the average chain length grows past twenty percent of the number of entries in a detail, it would be more efficient to serially read the entire data set, as fewer blocks would be read.

So, if the average chain for a given path exceeds twenty percent of the detail data set, the utility of the path is highly suspect, and in all probability it could be eliminated.

To pack a given path, or set of chains we have several alternatives. One method is to unload, erase and reload the entire data base. This can be extremely time consuming. Another method is to use the DETPACK feature build into ADAGER, and the third method is to write your own custom detail packing utility.

2. Small Detail's Tied to Large Masters

Nothing wastes more disc space than tying a small detail to a large master! It is not at all unusual to find a hundred thousand entry master with a path into a detail with capacity for three or four hundred entries. Six words of chain head space is reserved in each master entry. If the

detail is less than 300 entries, make it a stand alone detail, and simply serially search it. If the detail is larger, tie it to a separate smaller automatic master.

3. Unused and Unneeded Paths

After a system has been in production for many years it is very difficult to determine which paths are being used and which ones are a waste of disc space and CPU cycles. One method is to code traces into applications, this method while direct is difficult to implement and adds overhead to the application. Another way to get at the same result is to use the Profiler built into turbo image. The Profiler can be purchased, but it probably would be better if consulting were purchased from HP to interpret the results. Either way the performance improvements from reducing the number of paths into highly volatile details can be dramatic.

4. Unused and Unneeded Data Items

Not too long ago, the only method to modify image data bases was to use the dbunload, delete data base, change schema, build data base, dbload process. This achieved the desired end result, but not only did it consume great amounts of time, but filler data items where required to hold the place of changed or expanded data items. This resulted in a tremendous waste of disc space. These filler data items can easily be removed with the utilities now available.

C. Large Image Logfiles

Prior to turbo image, if your application wanted to implement disc transaction logging, you had to build extremely large logfiles, two or three times larger than ever would be necessary. This was due to the fact that if the logfile filled up, all processing against the data base would halt and the data base could become corrupted. Turbo image has the facility to log to successive numbered logfiles. Thus these logfiles can now be down sized by factors of two or three times.

V. Performance Implications and Conclusions

Review your system configuration of directory and virtual memory on a timely basis, overconfigured values waste disc space, underconfigured values require a reload to expand.

Understand the HP provided accounting structure and remove any accounts, groups and files you don't require, only after backing them up to tape.

Plan for growth of disc space requirements, even if the number of terminals on your system isn't. Running out of disc space can at worst cause a system failure, but at least will interrupt your production system.

Review the states of your disc space on a timely basis. Characterize the need to manage each category on a scheduled basis. Spooled print files tend to grow as the amount of data stored on the system does.

Purge the trash files off of your system on a periodic basis. This will make more valuable disc space available, and in the bargain speed up back ups and realods.

Archive files on a periodic basis to remove files that aren't accessed.

Use VINIT compress as a strategy to deflect the need to reload your system. Periodic compresses will coagulate available space, and thus the period of reloads can be lengthened.

Plan your reloads! It's better to do them on a planned basis than on an emergency one.

Be critical of manual master data sets, make them justify their existence.

Combine automatic masters of same data item type and size.

Track the average and maximum chain lengths in each detail data set. Watch for chains that exceed twenty percent of the detail capacity.

Pack the primary path of your most frequently accessed details. This activity alone can reduce response time by an order of magnitude.

Make small details tied to large masters stand alone, or give them a separate key.

Remove unused or inefficient masters, paths and data items.

If you have converted to turbo image, implement sequential logfile numbering, and down size your transaction logfiles.

These are only a few points to ponder. It is clear that disc space can be traded off for CPU cycles. The most frustrating SE situation is telling some distraught system manager that the only solution to the problem at hand is reloading his or her system, or worse data base. The only way I know to avoid these problems is to take the time to implement some or all of the above.