W H E R E , O H W H E R E H A V E M Y D S T ' S G O N E ?

Steven M. Cooper

Allegro Consultants, Inc.
2055 Woodside Road, Suite 170
Redwood City, CA 94061

Where, Oh Where Have My DSTs Gone?

by Steven M. Cooper

Introduction

Most of us in the HP3000 world have long ago learned of the 192 limit on CST entries. Many have encountered the 255 data items in a data base limit or the 65,535 limit to the number of entries in a single chain. But running out of DST entries? Never, you say. Well, a few years ago, maybe. But lately, more and more large sites are hitting the maximum of 1024 DST entries. You may have 110 terminals and 110 ports, excess CPU and disc channel capacity, and plenty of work needing to get done, but if user number 63 is unable to log on because he or she is "UNABLE TO OBTAIN DST ENTRY", you're stuck until you figure out how to free up some entries. Unfortunately, no manual gives you so much as a hint as to how to do this.

This paper investigates the DST, how it is used and when, and most importantly, what can be done to free up entries before it becomes a brick wall on your system.

What's a DST, anyway?

The Data Segment Table, commonly known as the DST, is one of the MPE tables that the operating system uses to keep track of what it is doing. One configures its size in SYSDUMP when cutting a new COLDLOAD tape, but in no case (at least until MPE Ve) can it contain more than 1024 entries. Each entry is four words long and contains information about one data segment currently defined on the system. A data segment can be a stack, an extra data segment, or an MPE table. (In fact, the third entry in the table contains information about the DST itself! Fans of self-references should like that one.) If the segment is in memory, the entry contains the bank number and bank address where that segment can be found. If it is not in memory, then the entry contains the address in virtual memory (on disc) where the segment can be found.

Where are they going?

The system uses about 60 entries for MPE tables. Another 30 or so entries are used by system processes for stacks and extra data segments; more are used if your system has communication lines (DS, RJE, MRJE, etc.) or spooled devices. So, figure about 100 are used before anyone even logs on.

When someone does log on, four more entries are used, assuming that no UDC files are used. If UDC files are used, add two more for the first UDC file and then one for every UDC file thereafter. In other words, if a system-wide UDC file is in effect, an account-wide UDC file is in effect, and two user-level UDC files have been specified, that user will use nine DST entries just logging on, before even running a program. It doesn't matter how many users are using the same UDC file; each user still gets a data segment for that file and hence a DST entry.

Eventually the user may run a program. At a minimum the process will need a stack using an entry. Then for each "normal" file opened, another entry is used. There are exceptions to this file rule, however. (Hence the use of "normal".) If a file is opened NOBUF (without buffers), then the extra data segment that normally contains buffers will not be needed and the file will not require a DST entry. Be prepared, though, to read a block at a time, not just a record, and do your own deblocking if you use this option. If a file is opened with the GMULTI option, then all users with the file opened this way will share the same buffers. Hence, the first opener will obtain a data segment for buffers and use a DST entry, but subsequent openers will share this segment and will not need any more for themselves. Though a good solution in many cases and an underutilized file feature, GMULTI does not come without its gotchas. More on GMULTI files later.

IMAGE data bases are a bit more complicated. The first opener for a particular data base uses an entry for the Data Base Control Block (DBCB) and another for a User Local Control Block (ULCB). If Intrinsic Level Recovery is being used, add one more for the ILR file. Subsequent openers will just add one more each for their own ULCB. The first time a data set is accessed by anybody, another entry is used that will store as many of the file control blocks for the data sets as will fit in a data segment. If enough open data sets exist, more segments may be needed to store these file control blocks. To put it all together, if a data base using ILR is opened n times, there will be n + 3 entries used on behalf of all the users for that data base, plus possibly a few more if the data base contains many data sets.

Well, have we accounted for enough to explain how we can run out? If we assume the four UDC files as described above (one system, one account, and two user) and assume that everyone is running QUERY with the same data base opened, let's count and see. QUERY uses one for its stack, one for its message file, one for its FIND file, and of course, one for the data base ULCB; that's four per user. Plus the nine when each user logs on as mentioned above totals 13 per user. If we have configured all 1024 possible entries and the system uses around 100, then user number 72 will not be able to log on, due to insufficient DST entries. In reality, most applications

are more demanding on DST entries than our QUERY example; they may open multiple data bases, KSAM files, and MPE files. If the application uses process handling or makes explicit use of extra data segments, then even more entries will be required. It is obviously not difficult to see where, oh where they have gone.

Bring back, Bring back, Bring back my DSTs to me, to me

One 'solution' would be to merely wait until MPE-Ve comes out. The number of DST entries is scheduled to be dramatically increased in that release. True enough, but remember that MPE-Ve will only be available on Series 42s, 48s, and 68s; not a solution for those of us with older machines. Besides, adding the necessary evil of indirect tables needed to increase these limits must have a cost in overall system performance. Rumor has it that MPE-Ve will run at least 10% slower than MPE-Vp, even if table sizes are not increased. Hence, many users will probably wish to stay on the non-expanded-table MPE, if there is some way to stay within the current limitations.

Our only other choice, then, is to free up some entries. The biggest culprit seems to be our friend the UDC. (Remember what they say about free lunches?) We are all in love with them and few would be willing to give them up, but moderation is advised. On a production machine, where most users log on to a NOBREAK, LOGON UDC that runs an application, and then logs the user off, there is no sense to having a system-wide catalog and an account-wide catalog. Some sites do use a system-wide catalog for an extra layer of security checks and an extra layer of consistency across the machine. However, if you are running low on DST entries, your use of UDCs is the first thing to reconsider. (Perhaps if every reader submitted an Enhancement Request on the subject to his or her S.E., ...)

The next thing to consider is opening commonly accessed files with the GMULTI option. With this option, all users share one extra data segment and one set of buffers. Not only can this reduce DST entry requirements, but can eliminate some concurrent update problems and minimize memory requirements for that file. If you open the file with enough buffers and if the file has a proper blocking factor, opening the file with the GMULTI option is a very nice, user mode way of getting memory resident tables for your application. Since everyone will be looking at the same data segment, it will probably never be swapped out.

Now for the gotcha. Everyone also shares the same Current Record Pointer. This is no problem if everyone is doing random I/O to the file (i.e. FREADDIRs instead of FREADs, FWRITEDIRs instead of FWRITEs, or in COBOL, RANDOM instead of SEQUENTIAL). If someone opens the file GMULTI, sequential, then as he reads, he'll get records 1, 2, 3, then 4. If someone else comes in and reads record 147, then the next record the first person will get will be 148, not 5! So don't add the GMULTI option unless you intend to do some careful testing.

Next, use a tool such as OPT to check the files that are opened for a given application. Eliminate any 'bugs' you may find that open files each time through a loop, never closing them, or leaving files open after they are through with them.

To free up more entries takes a more drastic approach. Some companies have turned to an applications monitor to solve this problem for them. This monitor runs as a job and launches applications as son processes for each terminal. This way, users do not have to log on, freeing up the DST entries associated with the Command Interpretor and UDCs. In one case, with 106 users on-line in applications, less than 800 DST entries were in use. Before applications can be placed under such a monitor, the application needs to be examined for its use of :FILE commands and JCWs, since all applications will now be running in the same job tree. Installing such a monitor is not a transparent, trivial exercise, but depending upon your level of desperation, may be well worth considering.

Conclusion

For those systems with a large number of concurrent users (at least 50), the number of entries in the DST should be monitored, since the limited size of this table may limit the number of users that may use the machine at any given time. Measures can often be taken to prevent this table from filling up. Taking such measures before the limit is reached will prevent the problem from occurring. If it has already occurred, then the measures may allow additional sessions, with minimal expenditure.

***

NOWRUG May 16-18