# HP 3000 - Sizing and Performance
# More Machine for Your Dollar, or "Where Did All My Hardware Go?"

by John S. Parkinson

## 1. Introduction - Why Sizing and Performance?

One of the commonest and most widely propounded theories of the computer industry over the last ten years (probably longer) concerns the falling real cost of computer hardware. We are always being told that any day "soon", hardware will be so cheap that it will be "given away" with application software. If this was true, there would be no need to worry about how much hardware was needed to process a given (and always increasing) workload, since a grateful manufacturer would simply present you with more equipment, whenever the need arose!

You have probably found it difficult to identify which particular hardware suppliers are taking this approach, and how many of those who do (if any) are still in business. As an HP 3000 user, you will have been attracted to Hewlett Packard computers by their ability to meet your particular needs, and to deliver the required performance at the right price. In doing so you will have already discovered the first principle on which this paper is based.

> TANATAAFL - or "There aint no such thing as a free lunch"

Unless you have an unlimited source of funds for acquiring computer equipment (unlikely but possible), a decision to purchase or upgrade a system requires justification. Unless you are very lucky, you will only be allowed to buy as much equipment as you need to process your workload. You therefore need to get your configuration right, and this is where sizing and performance assessment comes in.

HP, who are as honest a group of computer sales people as you are likely to meet, will be quite happy to tell you what you need, but how do you assess whether their advice is adequate or correct? This paper gives you some guidelines, and suggest some other profitable sources of information and advice.

One final point. This paper is being written in November 1983 and hence refers to the state of play as of now. I have tried to include what I know about MPE-V, disc cache etc, but the delays and changes in the release of MPE-V make it difficult to be exact. By the time you see this, some of you MIGHT have MPE-V in use. If so you will know if I got it right (or even if HP did). So please treat the sections that deal with possible developments with caution. Also, be aware that most of the material in this paper is a simplification of a complex process. I have included general examples wherever possible, but they ARE generalisations and may not apply to your particular site. If you do have a problem, ask some one; HP; another user or the user group. Do go to user group meetings. I doubt if your problem is unique.

## 2. Workload Categorisation

The first place to start a sizing and performance exercise is with the work you expect your computer to process. This may seen absurdly obvious, but I have seen a lot of sites where what a system is actually used for varies quite a lot from its stated raison d'etre.

Minicomputers like the HP 3000 are extremely powerful general purpose tools that are DESIGNED to handle a wide variety of possible tasks. It is even possible to do SOME of these tasks AT THE SAME TIME. If, however, you expect to be able to do ALL of the possible

kinds of work simultaneously, you are either going to be disappointed or to buy a lot more hardware than you thought. Ten years ago there was no problem, since computer systems only did one kind of job - batch processing. Thanks to progress, your HP3000 could now be used for some or all of the following:

Batch Processing (this one will never go away) Online transaction processing Online data capture Electronic mail Word processing Communicating with other computers Telex management Document design and printing

and last but not least, software development and maintenance.

This is a lot to ask of a computer, and to enable all these facilites, the designers have to compromise so that, although each will work, none works as well as it would alone. Since different classes of task have different processing requirements, no single architecture will satisfy all equally well, and anyway (whisper it) some of the system software designers were better than others(!) and hence some applications work better than others too.

So the first step we need to take is to figure out what our system is supposed to do. Recognise that this is only a CURRENT viewpoint, and will change as we progress, so also expect to keep an eye on what the system is actually doing from time to time. If we can use the system for several different jobs with the same characteristics, we can tune the machine to its optimum for these jobs and so get more work through a given configuration. If the job mix is very varied, we can review the desireability of processing it all on a single system, and so on. If, however, we don't know what the system is to do (or doing) none of this is possible.

Workload categorisation is not easy, especially for users with no previous computer systems experience, but is worth doing, even if you ignore the rest of this paper. If you don't feel able to do it yourself, get some outside help. You will almost always save money in the medium to long term. At the very least, write down a list of applications and see which category from the list shown above they fit into.

Once general word of advice. The system development tools on the HP3000 are very good, and tempting to use. However, interactive software development eats up system resources at an alarming rate, and all too often production work suffers. If you are a software house, this is no problem, but if you are really there to process an MRP or Accounting application, keep the programmers off the machine! It will often be cheaper to buy them one of their own, than to provide a configuration that will run both kinds of work side by side without affecting response and run times.

Also remember to allow for MPE and other system products. They all need disc space, memory and CPU cycles and the more facilities you use, the more resources MPE will take up. This is important, because there is relatively little you can do about it.

## 3. Throughput Assessment

Once you have decided what kinds of work are to be run on your system, the next step is to estimate how much of each kind of work needs to be processed. This also seems rather obvious, but it is important to produce estimates that reflect factors relevant to the overall system performance, not just the computer equipment. Various levels of sophistication can be employed in generating estimates, from simple counts to complex stochastic models, depending on how critical a factor the pattern of throughput actually is. You should not produce a model that is more complex than necessary, nor ignore the estimatation process altogether. Quite a good idea is to generate an 'average transaction' for each type of workload and use this, but remember that peak loads can often be many times the average value, and that some processing jobs will be time critical. Once again, if in doubt, get some help.

The throughput assessment tells you several important things. First of all, it will give you an estimate of the MINIMUM number of terminals (VDUs and printers) needed to process the interactive workload. Note that this may be less than the configuration requirement, since it takes no account of location or convenience factors, and is often based on fairly crude estimates of workflow, but it is a lower bound, and determines which series of 3000 computer will suit you.

Second, it will provide an estimate of how much file I/O capacity you will need. At this point you should also estimate how large (and how many) your files will be, and decide on their structure (IMAGE and KSAM files have variable 'overheads' dependant on the complexity of the data structures in use). This will determine your options for disc configuration, and thus how many access paths to the discs.

Finally, you will get some idea of how much memory the system will need. We will look at this later, but in general, always get as much as you can afford. Details of other peripherals (system printers, magnetic tape drives etc) also come out at this stage, but are relatively easy to determine with some simple arithmetic.

If you expect to run a lot of batch work as well as (or alongside) your interactive load, you have probably made the wrong choice of computer. However, this needs to be estimated here too, as it will contributed to memory, CPU and disc usage. Since there is no terminal wait time on batch jobs, they can use up a lot of file I/O, even if they are put in a lower priority queue.

## 4. System Limits

Once we have a picture of the type and volume of processing that our system needs to do, we can look at the various types of HP3000, and see which best fits our needs. The first thing we need to be aware of is that each model in the range has certain limits, some imposed by the architecture and some by HP. We will start with some 'overall' limits and work inwards. Configuring an HP3000 gets more complicated all the time. Five years ago, HP gave you a simple diagram on which you could tick boxes to decide what components you needed, and it was clear 'how many of what' could be added. Now you have a four page long list of questions to fill in, and even this does not cover all the possibilities. This growth in complexity has been, in part, the result of the move from the old Series II/III machines to the HP-IB bus architecture, and the attempt to use "standard" components for interfacing to the bus. all subsequent remarks refer to the HP-IB (and later) machines, and do not to the earlier SIO bus.

The use of the HP-IB introduces our first real limit. The maximum clock rate for the bus is c. 9.5MHz, which allows data transfers (on the IMB) at a maximum rate of 3 Mbyte/sec. The GIC maximum is less than this, at 1Mbyte/sec. To maintain this rate, the bus length must be short, so that the electrical properties of the backplane and cables used do not degrade performance. For high speed devices, therefore, the maximum practical bus length is c. 6m. We will return to this speed limit in later sections.

The other most critical limitations are as follows:

Power Supply - Each board plugged into the backplane draws power from the system processor unit power supply. Even if you are careful to use a sequenced power up on your system this is a major limit. It seems that whoever designs the power supply for HP was never told that anyone might want to actually fill up the card cage, so there was no need to allow for this possibility in design!

In practice, you can juggle the total power consumption loading a good deal more than HP allow. The configuration manual rules tell you what is formally allowed, but friendly CE0 managers may allow some deviations. (If not, you could consider doing your own maintenance!).

Card Cage - This is what determines (PSU allowing) how many boards can be plugged into the HP-IB backplane. Slots are NOT interchangeable, however, because some boards need to be linked together by ribbon cables and/or have hard wired addresses within the bus controller program. This means that some slots cannot be used. On the series 4X, for instance, there are 17 slots for memory boards (based on the original 16 x 256KB boards + 1 add on controller). With 1024KB boards you only need 4 of these to get the 4Mbyte system maximum, but the remaining 13 cant be used for anything else! With a rumoured 4 Mbyte board (using 256K RAM chips) rumoured for early 1985, this situation will be even more riduculous!

The remaining 26 slots are for ADCC's (or SIB/ATP's), GIC's, INP's etc. However, even here you don't have a free hand. Originally 15 ADCC's could be installed (total 60 ports). With the advent of the ATP, this could be reduced to 7 boards (one ADCC is required plus one SIB and 5 ATP's) for a total of 64 ports and 8 free slots. However not all these can be reused, because of PSU limits!

Once again, we need the configuration rules to tell what is allowed and what is not.

Junction Panels - Each I/O card communicates with the outside world via a cable/connector which HP like to look tidy. This in effect means that each machine uses a different cable (well they have different part nos) which is NOT transferable, even if the board is! The CEO doesn't like cables hanging out of the back of machines (unless they're series 39/40/42!). So you need the correct version, plus a junction panel to screw it to. When you run out of junction panel space - thats it; no more connections allowed.

MPE Kernel - Ever since the 3000 range 'mushroomed' from the old series III, there have been two parts to MPE. This is necessary, since at the hardware level, the machines are radically different. However, HP also chose to incorporate other limits, which determine, for instance, the memory and channel address limits and DRT liits for a particular series machine. I don't know why they did this but there you are. So, even if you COULD put 16 1Mbyte boards in a series 4X it wouldn't do you any good since the 4X version of MPE wont talk to an address above 4Mbytes. If you think this is daft, please write and complain. It might do some good.

System Tables - There are LOTS of limits in this bit. If MPE's to talk to a configured device, there must be a DRT entry for that device. The size of the DRT therefore limits the maximum configuration In practice, c150 entries is the most available under MPE IV which in most cases is more than can actually be used, anyway. Some other interesting limits are:

- Code Segment Table Entries (192)
- Process Identification numbers (256)
- File system directory entries
- Open files per process and overall
- Buffer space allocation

If you come REALLY close to some of these limits, you can run into others as well. However, MPEV will change all

this, so we wont dwell on it here. See below for the latest news.

Program Size - The HP3000 is a 16 bit computer. The maximum directly addressable offset is therefore 64K. Since opcodes only use a maximum of 15 bits for an address, this limits a CODE segment to 32 Kbytes, although data segments can be full 64K. Since you can have up to 64 code segments in a program, the maximum program code size is 2 Mbytes (note: you can actually build programs bigger than this, but not load them). Note also that the limit to VIRTUAL memory is 60,000 sectors of disc, i.e. 15 Mbytes, and that this is less than twice the maximum real memory size on a model 6X.

HP Support - In the last resort, HP will only maintain and support a system configuration that they have actually tried out and therefore know will work. How far you can push this depends on the attitude of your local CEO and what you want to do. For most sites, it is best to stay within the published guidelines and ask politely it you want to do something new.

### 5. Critical Resource Assessment

We now have most of the information needed to begin looking for particular problems. This is where the going gets a bit more difficult, as it is often not obvious which problem we should be looking at, and solving the wrong problem is all too easy. For our purposes we can divide the computer up into a small number of "components", the size or capacity of which might be a problem. In every system, there will be at least one of these that we don't have enough of. This is the CRITICAL MINIMUM resource, and unless we do something about it, other changes will not improve performance or throughput. The trick is to find it! Here are places that it is worth looking in detail.

CPU - The current CPU's in the 4X and 6X series machines are fairly fast (0.4 mip and 1.0 mip) so it is unlikely that these will cause bottlenecks. Remember, however, that effetive use of the disc cache facility will use 20 - 30% of the available CPU cycles, and that I/O interrupts on the CPU can also degrade performance. Note also that if the capacity required for disc caching is NOT available, then performance will DEGRADE!

Main Memory - This is a hot favourite for problems. You get quite a lot of main memory on current systems (0.5 Mbytes to 8 Mbytes) BECAUSE YOU NEED IT In fact, I don't know why HP sells ANY HP3000 with less than 1Mbyte of memory, since 0.5Mbyte systems run out of steam under virtually any real loading. The 6X Machines also have an 8K high speed cache (why not 64K?) for high speed access. Access time is not really the issue, however, since a system with, say 60 users has a need for a large memory size rather than raw speed. The need to use disc I/O to retrieve virtual memory segments slows things down a LOT (ten to fifty thousand times slower)

so the more active code and data in real memory the better. I have seen 4X based systems where 70% of all disc I/O was virtual memory swaps. Disc cache WILL NOT HELP this, since it needs large chunks of 'spare' memory to hold files, not programs and data, and if there is not enough of this (say 25% min) performance will degrade.

Note that MPE itself uses a good deal of memory (c. 180 Kbytes minimum to c. 850 Kbytes max. I have never seen a 3X or 4X system exceed this figure, although it clearly could do so in theory. 6X system can use up to c. 1.5 Mbytes).

Disc I/O - This is a complicated area. Disc I/Os comprise three main types:

1 - Virtual memory swaps (see above)

2 - System I/O, covering MPE activity, spooling, logging (system and database) directory management etc. Much of this is "invisable" (i.e. not directly caused by applications code) and so gets forgotten. It can add up to 30% to the total so it needs to be included. Disc cache (memory permitting) could help here by holding log files etc in memory but Note also that much of this activity is a WRITE to disc rather than a read, so cache may not help after all!

3 - Application program I/O to data files (IMAGE, KSAM etc). This should be the biggest volume of I/O and is the area that disc cache is designed to help. To do so, your disc activity needs to be 75% reads (or better). In many online applications, this is a good bet, given the overheads involved in directory reads and data structures, and that enquiries and amendment transactions all involve more reads than writes. However some applications, (e.g. sorts, file creation, data capture) may be 50 - 100% writes, and disc cache wont help much here.

Disc I/O efficiency also depends on physical configuration factors. The nearer the ratio of controllers to drives, is to one, the more paths to the data exist, so I/O can be in parallel. The more spindles per volume of data, the less head contention will be present. Multiple GIC's further increase the level of parallelism available. The secret of this is to use as much as possible of the available band width. As was mentioned above, we have 3Mbyte/sec on an IMB. Since disc I/O is fast (1Mbyte/sec in burst mode) we can use a lot of the available band width if we arrange transfers efficiently. In the past, the BEST we could do was c. 30 I/Os per sec, and most systems achieved less than this in

practice. Now, with the new CS80 discs the 4X and 6X machines can delivery 60 - 150+ I/Os sec, but they still need to be utilised to be effective.

The trend is also to provide more disc space per spindle, at a lower cost per bit stored. This is OK in theory, but with more tracks to search, a 400 Mbyte disc had a WORSE average performance than the old 7920/25 discs This has now been put right, so that all the discs (791X, 792X, 793X) have more or less the same performance on a per drive basis. How they perform on a particular system depends on lots of factors, like buffer size, file organisation, use of private volumes etc.

Finally it is worth mentioning that HP are introducing Rotational Position Sensing (RPS) on their CS80 disc drivrs. This senses when the right sector will come along and be under the Read/Write heads and frees up the GIC while waiting. Although this wait is "small" (c. 10 ms equal to the average rotational latency), the GIC can do a lot of work in the time if it has to service several discs.

Terminal I/O

Terminals and other slow devices work so much slower than everything else that you might think they are no problem. Usually this is so but there are one or two things to remember.

ADCC's generate a CPU interrupt on every 'block' transfer from an attached device. This is bad news if your CPU is heavily loaded but is usually only a problm on 3X series machines. This is why block mode applications use less CPU than character mode. However a block mode transfer can use up lots of buffer space both in main memory and on the ADCC board. We have found that using V/3000 screens at 9600 bps on an ADCC can result in data loss if all 4 channels run like this and the system is heavily loaded. ATP boards don't have this problem. They do DMA transfers and don't involve the CPU.

Other Things - Like system backups to tape. In theory you could put your system log files on tape (does ANYBODY do this?) But in practice you need to use STORE or SYSDUMP now and again. If you have a lot of disc files that change regularly, this can take a LONG time, even if you pay a LOT of money for a 7976 tape deck. Help is on the way in the form of a dual mode MT with fast streaming for backup and start/stop for other uses. (7974). By the time you read this you should know how much it will cost.

System printing via the spooler is another problem. Every spooled printing operation uses disc transfers and GIC bandwidth. Still, you could always buy a laser printer.

In practice, one of CPU, Main Memory and Disc I/O will be the critical resource. When you find out which it is, you can try to do something about it. We will look at tools for this later.

## 6. Complications

So far our approach has been based on a relatively simple single system example. Real life can be much more complicated. Networking, RJE, IML, Graphics, the list of possible complications increases all the time. The biggest complication however is supplied by the applications software being used, and the way in which it is designed, operated and mixed. The first thing to remember is that individual processing loads to not add up linearly. Two applications that each use 20% of a resource do not, when run together, usualy use 40%. The relationship is closer to exponential. This is becasue of the myriad of individual queuing interactions that take place within the system. Sorting this out is a job for experts and specialist tools. In general, however, a small number of applications can support more users than a larger number.

The efficient design of applications makes a big difference too, but not as much as some people think. Most problems come from a few design factors, such as:

- database locking strategies segmentation
- database design trying to be too clever
-

System level software can also cause unexpected complications, as anyone installing HPWORD will have discovered. At least when your own applications are compiled and prepared, you get some clues on how much resource will be needed to run them, and can control segmentation. With HP's own software, this is all outside your control. **Sometimes this shows!**

If you want an excellent view of how to assess the basics of performance/design criteria, try Jim May's "Programming for Performance" in the proceedings of the 1982 IUG conference in San Antonio. There is plenty of other published advice on this around, so read INTERACT regularly and write or call the people who contribute articles. Even better, write and describe your problems, you might get published!

## 7. Performance

Now that we have identified all the main components of sizing and effective use of resources, we can begin to assess what our system will do. I do not intend to go into a lot of detail here, as much has already been written on performance and there are bound to be other contributions during the meeting that will address this directly. However here are some general observations reflecting important trends.

- Anything you do on the HP3000 will go faster if written in SPL. HOWEVER, it wont necessarily go much faster, and in SPL you have to do it RIGHT to get the benefits. There are also lots of ways to do it wrong, and make things worse for everyone else on the system as well. So stick to COBOL (or whatever) where the chances of REALLY bad code effects are reduced.

- If you use a code generator or applications development system, don't expect to get as resource efficient applications as you would if you wrote code. Remember TANSTAAL. What the so called fourth generation approaches give you is applications that arrive faster, not run faster. We have run some tests on several of the available systems and found that, used at their simplest, they are 30%-50% less resource efficient than good COBOL code, while at their most complex, they get to within 10% of the equivalent COBOL. Of course you can go just as badly wrong in application design using these systems as with traditional methods.

- If you have a memory bound system, avoid the use of block mode, which uses lots of memory for buffers. If you still want to run V/3000 applications (you probably will), use something like PREVIEW from TYMLABS to convert to character mode. Remember, however, that ADCC based systems will generate many more CPU interrupts in character mode (TANSTAAFL again), and this may affect disc cache performance.

This is all getting rather complicated. To sort our what is going on on the system, we need some quantative data to identify which resources are over or under utilised, and to direct efforts to the most critical areas. We will look at these next.

## 8. Tools

All the above is of little use (outside of intellectual exericse) if you have no way to measure what is going on in a real situation. To do this you need some tools. Fortunately there are several available, both from HP and from third parties. Most of them work OK and some of them are useable by non experts. Unfortunately, many sites don't think its worth spending money on them. This is silly, since the TOTAL cost of ALL the available (and usable) tools, plus the cost of help in learning to use them is probably less than $50,000. A good representative selection, including some in the contributed library, probably costs less than $10,000. Most systems costs a LOT more than this, and most sites annual upgrade budget is more than this too. Here are some that we use all the time.

- TUNER (contributed library)
- SOO (contributed library)
- PROGINFO (contributed library)
- LOOK

There are quite a few others available too.

We also use the OPT/APS software from HP, but we hve found it rather more comple than the above. Also the best bits are not regularly available to customers, being regarded as consulting tools needing HP's own staff to use them. (This can be expensive).

## 9. MPE-V (I think)

Everytime I think I've got the 3000 sorted out, along comes something new to change things.

Here are some of the major enhancements that MPE-V should bring. Firstly increased table sizes.

CST up from 192 to 2048 BUT any one user can only use 512 entries of which 256 must be HP library segments.

Maximum program size is up from 2MByte to 8Mbyte.

```
PCB up from 256 to 1024
DST up from 1024 to 4096
Disc request queue up from 256 to 900
I/O request queue up from 256 to 1300
More entries in file system director
DRT changes  -  up to a total of c. 500 devices of which
                336 can be local terminals
                50 can be DS sessions.
```

Secondly, disc cache and RPS on the CS/80 discs.

Thirdly, new communications support for DSN/X.25 and (maybe) a local area network.

Fourthly, faster file backup routines.

As a result of the above, MPE-V will also bring

- larger system code segments (all these extra facilities have to go somewhere)
- more memory utilisation (including DEGRADED performance on small memory systems)
- more places to go wrong if you're not careful.

What MPE-V will NOT bring is:

- larger code and data segments
- larger real memory address space (I might be wrong on this. If the 4 Mbyte board really is coming, it will need an enlarged memory address capability)
- downwards compatibility to MPE-IV. There will be an intermediate product (MPE-VR?) for users of 3X systems and the older Series II/III without the increased table size features and the RPS driver modification.

You will, however, be able to get more out of your 4X system without finding the space, power, A/c and money for a 6X upgrade.

## 11. Conclusions

I hope you now feel that the effort of a sizing and performance review is worthwhile. Can I also suggest that it is worth doing BEFORE you have a problem to solve, and that this 'prevention mode' exercise could actually head off a problem? I am sure that installations that know what is going on with their system fare better when bidding for new resources than those that do not. They probably give a better service to application's users as well. This need not cost a lot of money and nearly always yields quick benefits. So think about:

- Workload and Workmix
- Transaction volume
- Good Application Design
- Performance Monitoring tools

and you will be able to SEE where all your hardware went, perhaps even get some of it back to work!

*John S. Parkinson graduated in the UK in 1972 with a BS in mathematics and an MS in information sciences. After 5 years working on a variety of health care related system projects for the UK National Health Service, he joined Sifo Sytems, a specialist health care systems house and consultancy, where he is now General Manager and head of Product Development. His experience with the HP 3000 started in 1978, when SIFO became an OEM and since then he has been responsible for the installation of systems in the UK, Europe and the Middle East. He has presented papers at several UK and European User Group Meetings, and runs a Sizing and Performance Seminar for users in the UK.*

---