Proc. CMG '88, Dallas, Texas, December 12-16, 1988.

SO, YOU WANNA RUN A BENCHMARK?

Ronald G. Nichols
National Advanced Systems

ABSTRACT

Despite the efforts of vendors to discourage benchmarks in all but a few cases, the number of benchmarks conducted by customers and prospects is on the rise. Management, sometimes disregarding the advice of their own technical staffs, has decided to arrange a benchmark as a definitive demonstration of performance. Once you have decided to benchmark, there is much to be done. The time, money, and effort required to conduct a successful benchmark should be subjected to the same management and financial reviews as any other data processing project. This paper describes the practical aspects of conducting a benchmark from the perspectives of cost, staff involvement, and the probability of achieving your desired result.

INTRODUCTION

Of all the points of confusion that exist about benchmarking, the one that stands out the most is its basic definition. If you were to ask ten different people to define a benchmark, you would probably get ten different answers.

In its simplest form, a benchmark is nothing more than running a particular software workload on a certain hardware configuration. Benchmarks are part art, part science, and, like statistics, can be used to substantiate any point of view. Benchmarks imitate art in that they are totally free-form, can be constructed using any workload or combination of workloads, can be as rigorous as you want to make them, and, like many works of art, you don't know what you have until it's complete. Conversely, benchmarks exhibit "scientific" characteristics in that, once completed, jobstreams yield results that are measurable, predictable, and repeatable.

Commercial benchmarks, especially processor benchmarks, lend themselves to measurements stated in terms of transaction rates or number of batch jobs. But in reality, benchmark results can be stated in any metric that is meaningful to you.

There are many things a benchmark is not. It is certainly not an indicator or predictor for a range of performance. A single benchmark run yields a single performance point. If trend information is desired, multiple runs are required, with either the workload or the hardware remaining constant. Then various measurement points can be graphed to provide an indication of expected performance. However, unless the environment is totally unconstrained (i.e., no hardware or software bottlenecks), the graph will not be linear and, in order to answer the "what can I expect" question, you must make an educated guess based on observed trends. To extrapolate potential performance from a known workload is tenuous at best, and is certainly more art than science. If you want an absolute answer, measure. If you want an indication, extrapolate.

PRE-BENCHMARK SETUP

A benchmark should be planned, constructed, and executed in the same manner as any other complex project. It is an erroneous assumption that a benchmark conducted on short notice with little or no controls can still produce meaningful results. Benchmarks are also expensive and time-consuming. Constructing an MVS benchmark from scratch can take up to six months. Updating an existing benchmark usually takes two months.

Given all the time and expense, why do organizations require benchmarks in the first place? The government, for instance, insists on a demonstration of capability for a large percentage of its processor acquisitions. The private sector often runs benchmarks, not so much to determine absolute performance or capacity as to ascertain relative performance between competing solutions.

Once you decide to run a benchmark, you must choose between vendor offerings. Each vendor offers something different. A user must be able to differentiate between systems utilizing varying amounts of real and expanded storage; between 3.0, 4.5, and 6.0 MB-per-second data transfer rates; between solid-state disks and cache devices; between single-, double-, and triple-capacity DASD, and between processor complexes having anywhere from one to six CPUs. The combinations are endless. The overall effects vary depending upon the workload in question. When faced with such product diversity, the usual reaction is to benchmark. Done correctly, benchmarking can provide answers to performance and capacity questions. But correctly-done benchmarks are not cheap, either in terms of personnel or in dollars.

In general terms, there are three types of benchmarks: the speedometer test, in which raw speed is the prime consideration; the functional test, in which compatibility is the focus; and the capacity test, which measures the ability to process higher transaction levels and more concurrent batch jobs. The speedometer and functional tests tend to be small, and once set up, can usually be run in a few hours. Capacity tests usually require more setup and run time. It is not at all uncommon to spend weeks at the benchmark facility conducting capacity-oriented benchmarks.

SETTING BENCHMARK OBJECTIVES

The point is this: it is more difficult to construct a benchmark in terms of workloads than it is to define the product of a benchmark.

In reality, the benchmark's composition matters very little, as long as everyone understands the scope, limitations, and purpose.

Without question, the most important facet of conducting a successful benchmark is understanding what you want to prove. It seems obvious, but all the preparation and expense that goes into benchmarking is for naught if the objective is not clearly defined before starting the benchmark. By first defining the desired outcome, it is much easier to set up the benchmark, as well as to provide a better description to the vendors of the resources required. Also, with an objective firmly implanted in everyone's mind, the benchmark team is more focused when they are at the benchmark facility.

With the objective defined, the next step is to get management approval and commitment for the project. They need to understand the costs associated with conducting a successful benchmark, as well as the expectations. Since nontrivial amounts are involved, you'll have to sell it up the chain.

Borrow a technique from the sales world, where sales reps are taught to describe their products or services in one sentence. The rationale here is this: if you can't describe it in a sentence, you probably don't understand what you're trying to sell. The same approach should be used when suggesting a benchmark to upper management. If you can't consolidate your reasons into a single sentence, you should either rethink your objective or restructure the proposed benchmark into something that can be easily quantified. The statement "I want to run a benchmark in order to show how our workload will run on the Latest and Greatest-80" indicates much too broad a requirement and has very little chance of reaching a successful conclusion in a reasonable amount of time. On the other hand, the statement "I want to determine if I can support 400 TSO users with subsecond response time" indicates a much more realistic goal and has a better chance of success. Simply put: how do you know when you're done if you don't know what you need to do? The more "yes/no" kinds of questions you ask, the easier it will be to interpret the results.

BUILDING THE BENCHMARK TEAM

Once management approval has been granted, start organizing the benchmark team and begin accounting for all resources expended. Project tracking and reporting should be implemented as soon as possible. Undertaking a benchmark is an involved, and sometimes protracted, process. Just as the DP organization would be very meticulous in defining the requirements for a new payroll system, it should also pay close attention to the requirements for setting up a benchmark.

The benchmark team should consist of representatives of all affected organizations within the DP community. *It is a mistake to consider benchmarking solely a technical support function.*

As the process evolves, it's evident that Operations must be involved, that the systems analysts must be involved, that the applications programmers must be involved, and, heaven forbid, even the end users must be involved. Certainly Tech Support plays a vital role, but don't lose sight of the fact that the data processing facility exists to service the needs of end users, not the tech support staff. After all, applications and user transactions are being benchmarked, not MVS and IMS. To conduct a benchmark without the input of the designers, builders, and users of these systems just doesn't make sense.

BENCHMARK PROCESS

A typical benchmark scenario is comprised of many steps. The first step is to determine whether a benchmark is really required or desired. You will then be asked by the vendor to complete some basic benchmark information, including the requested hardware and software configurations, the time frame for the benchmark, and the measure of benchmark success. Any configuration restrictions should be noted at this time. For instance, must the hardware be set up with 128MB of real storage and 128MB of expanded storage, or can the vendor determine the optimum configuration? Can the vendor "tune" the environment to best showcase their product? All existing limitations should be discussed before arriving at the benchmark site.

This information is forwarded from the local marketing rep to the benchmark center for evaluation and scheduling. Part of the evaluation process is a broad determination of the probability of success given the configuration and the measurement criteria. It makes little sense to spend the time, effort, and money to conduct a benchmark with a performance requirement that can't be met.

In scheduling the benchmark, many factors are considered:

- Real and expanded memory requirements
- How much DASD is required (by type)?
- What about front-end processors?
- Is a driver system required?
- The time frame, both in terms of requested dates and the number of hours to be used each day
- Software requirements, especially when dealing with products not currently installed
- Availability of benchmark staff
- Local marketing requirements
- Prior benchmark center usage commitments.

Your benchmark time will consist of two elements: the preparation phase and the actual benchmark run. During the prep phase, all DASD used for the benchmark is allocated and loaded with data and programs brought from your site. The machine is configured based on your benchmark requirements, and, time permitting, a first "sizing run" is conducted to determine if any bottlenecks exist. The hardware and software configurations are then tuned to give the maximum performance possible during the actual benchmark run, given the constraints and the measurement criteria.

After the final benchmark performance run, any available data is analyzed again for bottlenecks or any other factor that would have an impact on the final performance results. A comparison is also made between the actual results and the expected results. That's why it is very important that you know the measures of benchmark success before you go to the benchmark center. It's very difficult to rerun jobs to get more information once you get back home.

THIRD-PARTY SOFTWARE

One of the areas that has the highest potential for delays while running the benchmark is the availability of third-party software. Depending upon your software environment, the benchmark center may not have the software package(s) that are crucial to your operation. If that's the case, what do you do?

There are usually three options available:

1. Ask the software vendor for a temporary "benchmark license" if one is not already included as part of the master license agreement.
2. Ask the hardware vendor to provide the software.
3. Bring your own copy. If you can't get a temporary license from the software supplier, you must ensure that only one copy of the software is running at a time.

No matter which avenue is chosen, it is important that software that isn't part of the benchmark center's configuration be given to the vendor as soon as possible. Installation of new products must be planned and coordinated at the benchmark center just as they are at your site. In addition, spending benchmark prep time installing required software is a very poor use of a critical resource.

Also, when planning to benchmark software that is not part of the benchmark center floor system, be sure to bring someone as part of the benchmark team who is knowledgeable in that product. It stands to reason that the benchmark center will not be very familiar with a product they don't have installed.

WORKLOAD CHARACTERIZATION

There are two distinct workload types: real and synthetic. A real workload is an actual subset of your current production environment. It may be a just a few of the many CICS transactions, for instance. Or it may be a portion of your overnight processing. It is, first and foremost, "real" in that it reflects something that is run every day.

Synthetics, on the other hand, are artificially constructed workloads. Instead of supplying an actual CICS transaction, you may elect to model its characteristics in order to produce something close to the real thing. Let's say transaction ABC takes 300 milliseconds to execute and makes ten data base inquiries and one data base update. If you were going to include this transaction as part of the benchmark mix, you could either bring transaction ABC, or write a new transaction having the characteristics of ABC.

Why would anyone write a synthetic if the real workload is available? Some installations have security requirements forbidding sensitive applications from leaving the site. Maybe the data base being accessed contains classified information. For any number of reasons, a decision must be made early on concerning benchmark composition. Most benchmarks being run today are real workloads, and the percentage of those containing synthetic applications is decreasing.

Assuming that you have decided to benchmark a real workload, the principal task is to gather those applications and systems that reflect either your current environment or one you anticipate. This is the situation where applications analysts and programmers provide an invaluable service.

The Applications group can tell you which systems are easily transportable, which ones have test data bases that can serve as a sufficient test bed, and which are likely to grow over time. Equally important, they will know which are **not** candidates for benchmarking due to complexity or major changes being made to the system.

Another area that should be consulted is Computer Operations. The operators can tell you more about the functional characteristics of a particular system than almost anyone else. Operations can relate the impact of the system to other jobs, and can tell if its usage requirements have expanded or if its run time has started to elongate.

Many papers have been written on workload characterization, with the underlying goal of being able to understand what work is actually being done during a "typical" day. The results have been used primarily in capacity planning, but the techniques apply equally well to constructing a representative workload for benchmarking. The objective is to pick a subset of the entire job or transaction mix that will be fairly typical of the distinct workloads that comprise the system, as well as being significant consumers (proportionally) of I/O and CPU resource.

The same technique must be used for data files. It does little good to go through the work of identifying a subset of the total system if you have to end up bringing the entire 60-volume data base to the benchmark center. Make sure that the data base requirement has been reduced, even if it means leaving some of the biggest jobs or transactions behind. It's much more difficult to get large DASD farms to run a benchmark than it is to get a big CPU. Depending upon the current benchmark center configuration, it may not be possible to add 30 more DASD strings to accommodate your entire database. From the user perspective, it's also more difficult to tune a 50-string data base than a 20-string one.

BUILDING THE SYSTEM

Once the benchmark team has been defined, the workload identified, time arranged with the benchmark vendor, and appropriate software arrangements made, it's time to construct the actual benchmark stream you will take with you. The end result is a package containing *everything* necessary to run the workload off-site.

The starting point is the documentation. Not only it is required to gather the correct programs and data, but it will also serve as a guide once the benchmark begins. Develop an audit trail. If you need a subset of your on-line CICS transaction, note why those particular transactions were chosen. Technically, specify how your chosen workload will be split out from all the other transactions. The same applies to the data. Are there special operating instructions? If so, make sure to include them as well.

How will the benchmark system be constructed during the build phase? Will you use spare DASD to create the system a volume at a time and then dump to tape? If so, does the benchmark center have the same backup/restore utility you're using?

After the entire benchmark has been built, schedule some stand-alone time to verify your work. If you're bringing your own operating system software to the benchmark center, download every tape volume and make sure you can IPL from the just-restored DASD. Don't forget -- you must bring your own maintenance libraries if you bring your own system. It's a tough job to put software maintenance on a system without the requisite historical data. Given the headaches associated with bringing your own system, I recommend using benchmark systems whenever possible.

To make sure your system is truly transportable, disable all DASD interfaces to the CPU except those volumes just restored. Check out everything. Do you have a catalog containing data and program pointers that you forgot to include in the benchmark system? What about home-grown utilities or things like customized SPF panels? Do all the jobs and transactions execute *and do they produce correct results?* Nothing is more frustrating than getting to the benchmark center, only to discover you have forgotten something.

This checkout period is also a good time to get some baseline performance data. One of the questions the benchmark staff will ask is how well this workload ran in your environment. If you don't have a clue, it will be very difficult to know if the benchmark is running correctly and if you are getting the best possible results.

One of the mistakes frequently made is what I call the *bigger machine effect*, a corollary to the infamous "large systems effect." In almost all benchmarks, the intent is to see how well a representative workload runs on a bigger machine. But many times the workloads brought to the benchmark center are not capable of being expanded to consume a much larger machine. At home, the benchmark workload may use 95% of the CPU, but it may only use 40% of the benchmark machine. It's important to remember the relative power of the benchmark processor when constructing the benchmark!

RULES OF THUMB

The following data are offered for planning purposes when sizing machine requirements for the benchmark system. These are general rules. If you need something more precise, construct an analytic model tailored to your environment.

- For 3090-class processors, each MIP requires 3.5 MB of real storage and about 4 GB of DASD.
- When using 3380-type DASD, shoot for a balancing number of eight I/Os per second per volume.
- Don't underestimate the DASD requirement. If the benchmark was unconstrained during the checkout and used 100 volumes, increase the DASD by a percentage equal to the "speed ratio" of the benchmark machine to your in-house processor. Expecting to benchmark something 50% faster? Ask for 150 volumes.
- When setting user think times, the average for other than data entry transactions is 25 seconds. Only 20% of the users had think times of less than ten seconds.
- When trying to increase the transaction rate, always add more users rather than reducing the think time. It is unrealistic to assume users will "think faster" just because the underlying delivery vehicle is faster. The nature of much of today's on-line processing is to generate a response based upon received input. It takes the same amount of time to read a screen full of data and formulate a response whether the screen filled in one second or two.

When evaluating a processor complex with more than one CPU, a ballpark figure for multiprocessor performance is given by the formula N^m, where N is the number of processors, and m is the estimation of the MP factor. Assuming an MP factor of .8, dyadic performance is 1.74, triadic is 2.41, and quadradic is 3.03. This implies that a four-way processor complex can be expected to perform at slightly over three times the base uniprocessor performance level.

When sizing a workload for potential throughput improvements based on relative CPU power, use the percentage of CPU time to elapsed time as a quick indicator of expected performance. For example, assume a job took 100 seconds to complete using 20 CPU seconds. Its CPU to elapsed time percentage is 20. This means that no matter how fast the new processor will be, it can affect only 20% of the total run time. Adding a CPU 100% faster would halve the CPU time to 10 seconds, but would do little to the elapsed time, dropping it some 10%. In short, adding a CPU that's twice as fast doesn't mean your jobs will run in half the time.

WHAT CAN GO WRONG?

Are you kidding? So far, all we have done is to specify the environment, gather a representative workload, construct a benchmark system, generate a baseline performance value, and get on a plane to the benchmark center. The task now is to actually run the benchmark. Piece of cake.

Rule #1: Expect the unexpected.

Rule #2: If things are going great, see Rule #1.

I won't even discuss things like tape read errors (you do have a backup at home, don't you?) and misplaced documentation. That happens all the time.

Unless you have requested a 24-hour block during the benchmark period, expect to spend some portion of your time doing backup and restores, even if you are supposed to have "dedicated" volumes. The rationale is simple enough: no matter how careful everyone is regarding DASD allocation, sometimes things get trashed. Most benchmark centers have multiple machines with between four and eight paths to each DASD device. Usually these processors are going 24 hours a day with a combination of benchmarking and performance testing. Machine time is a scarce commodity, and many different groups are rotating through the center each day in order to maximize its use.

To further remove the chance of error, write down the exact machine room configuration pertaining to your benchmark. How many channels? How many devices? Controllers? Paths? Expect to spend the first part of each time slot validating the configuration. Chances are nothing has changed, but don't take the chance.

At best, a benchmark is hours of boredom punctuated by periods of frantic activity. Therefore, pay particular attention to the people aspect. Make sure each benchmark attendee has a defined function. Having people sit around with nothing to do is counterproductive. Keep the on-site team small. Depending upon the benchmark length, consider rotating people in and out during the actual run time. Burnout is a real possibility. Don't let it happen to you.

REPORTING THE RESULTS

The end product of the benchmark process is a recommendation to management based on the technical evaluation criteria. As part of the initial decision to go ahead with the benchmark, the team should have a clear understanding of the items that need to be addressed in the final report. If the current hot button in the organization is TSO response time, for instance, then the report should be formulated in terms of how this problem is solved by the new CPU. Upper management does not want to see RMF output, but you will need the backup data for the inevitable challenge by the losing vendor(s).

Start your report the moment you arrive in the benchmark center. Keep a diary of events, a notebook, a log -- anything that will help you remember what transpired during the benchmark. The days will be long and intense, so it's easy to forget something that happened earlier in the week. Write everything down.

As a section of the benchmark is finished, start the analysis and writeup immediately. Why wait? By starting the paperwork while it's fresh in your mind, you're less likely to forget pertinent facts and details. Also, and perhaps more importantly, you can tell if you have enough data to do the analysis. What kinds of data do you need for each run? Do you need job logs? RMF data? SYSLOGs? Whatever it is, make sure you have it in a safe spot after the job has been run. Construct a checklist for the kinds of data needed for evaluation purposes. In most benchmark centers, your team will have a room to themselves, but by the end of the benchmark every square inch of table space will be covered with listings, JCL, SYSOUT, and RMF data. The last thing you need to do is to have to find something on Friday that you should have saved on Monday. Make it easy on yourself.

CONCLUSION

Deciding to benchmark is easy. Translating that desire into a meaningful benchmark is something significantly more difficult to achieve. Despite all the time, effort, and expense involved, a benchmark can yield results important to the acquisition process.

But you must also be pragmatic. Understand the weight given to the technical evaluation. If it's small, you can afford to be less rigorous in your approach. Maybe you can turn a performance benchmark into a functional one. You may be able to reduce the time requirements significantly while still providing results that are good enough for your needs.

Also understand your current environment. What has been your growth rate since the last CPU purchase? Do you expect it to remain constant? Do you see a shift in the kinds of work being done by your users? If so, make sure the new work is reflected in the benchmark.

Most processor benchmarks are used to set the `minimum` size CPU given current or projected workloads. And, while it is important to determine the minimum processor needed to satisfy your business requirements, it is also important to determine `maximum` requirements as well. Buying a CPU too big for your projected needs is not the best way to utilize scarce corporate capital.

Remember -- benchmarks are supposed to be technical evaluations of proposed solutions. If you benchmark with three vendors, you will get three results. You must be able to reduce the data so that a ranking can be determined. Assuming you are comparing speedometer test results, it makes for a much easier evaluation if you can say the "fastest" vendor is first, and the "slowest" vendor is third. If you're required to make an evaluation that calls for minimal acceptable performance levels on a pass/fail basis, I suggest that the weighting of the technical portion be reduced.

Don't forget that no matter how involved the benchmark, the technical portion is only one segment of the overall selection criteria. Technical people are often surprised when the winning vendor was the one that came in last in the benchmark. Don't be. Other factors such as support, price, contract terms, future products, and discounts on other hardware and software are all part of the process.

One last thing: don't trash all your work once the acquisition decision has been made. Chances are you'll be called upon to verify the newly-installed machine with the benchmark workload. And, if you have constructed the benchmark jobstream properly, you have some valuable insight into the way your customers are using the machine. The raw benchmark results can be plugged directly into a capacity planning model in order to forecast future requirements.

It's been a lot of work, a lot of head scratching, and probably more than a little cussing. But you survived. And sometimes that is the true measure of benchmark success.

GENERAL ACKNOWLEDGMENT

I would like to thank Bernie Domanski and Pat Artis for the following references. Also, their class "Benchmarking MVS Systems" helped crystalize some of the topics included in this paper.

REFERENCES

Listed below are some of the many papers written on the subject of Workload Characterization:

[1] Hughes, "Workload Characterization of Computer Systems," CMG '78 Conference Proceedings, p. 81

[2] Hughes, "A Study of a Procedure for Reducing the Feature Set of Workload Data," CMG '80 Conference Proceedings, p. 16

[3] Wight, "Cluster Analysis for Characterizing Computer System Workloads -- Panacea or Pandora?," CMG '81 Conference Proceedings, p. 183

[4] Jackson, "Workload Characterization and Capacity Planning at a Large IBM Installation - A Case Study," CMG '81 Conference Proceedings, p. 248

[5] Lee, "Workload Characterization of IMS Using Cluster Analysis," CMG '83 Conference Proceedings, p. 341

[6] Jacobs, "Computer Resource Usage Analysis for Use in Characterizing Workloads," CMG '86 Conference Proceedings, p. 157

[7] Smith, "A Technique for Analyzing VM Workloads," CMG '86 Conference Proceedings, p. 474

[8] King, "Workload Characterization," CMG '86 Conference Proceedings, p. 789

[9] Gibson, "Sizing New Automated Systems Having Unknown Workloads: A Structured Approach," CMG '87 Conference Proceedings, p. 177

[10] Samson, "Workload-Oriented Performance Analysis," CMG '87 Conference Proceedings, p. 290

BENCHMARK CHECKLIST

- Make the decision to benchmark
- Determine the criteria for success
- Decide on the type of benchmark to be run
- Secure management approval <u>and</u> funding
- Inform vendor(s) of your intentions and ask for assistance
- Assemble the benchmark construction team
- Implement a project tracking and resource allocation system
- Begin gathering potential workloads and associated data
- Determine third-party software needs
- Initially size your benchmark for machine, software, and time requirements
- Inform vendor(s) of your requirements and time parameters
- Build and test benchmark system
- Choose on-site benchmark team and assign specific tasks
- Use prep time to make sure everything "works" and to get an initial sizing run
- After each portion of the benchmark completes, isolate necessary computer-generated output and begin writing report
- Submit final report to management