# ON BENCHMARKS: DYNAMICALLY IMPROVING EXPERIMENTAL COMPARISONS

by

Leonard Uhr

Computer Sciences Technical Report #571

December 1984

# ON BENCHMARKS: DYNAMICALLY IMPROVING EXPERIMENTAL COMPARISONS

*Leonard Uhr*

Department of Computer Sciences
University of Wisconsin
Madison, Wisconsin

Benchmarking lies at the core of the evaluation, development and improvement of systems, both hardware and software. Basically, it is the application of the experimental method in the practical world of real tasks, real programs, real algorithms, and real computer structures. To be successful and useful, the benchmarking process must be a dynamic, changing, adapting enterprise that engages the interest and the efforts of the whole community of researchers involved.

This paper describes a set of procedures for implementing a continuing cycle of benchmarks on a variety of different systems, whether (as deemed most appropriate or feasible by the individuals involved) by actually running the benchmarks on an existing system, running them on a simulation, or (more or less roughly) estimating how fast they would run. It examines how benchmarks cut in both directions - for they evaluate and compare the programs and algorithms used to benchmark as well as the hardware structures on which these benchmark programs are executed. And it explores some of the philosophical issues that underlie benchmarking and the total experimental evaluation methodology of which benchmarking forms a vital part.

## 1. INTRODUCTION

The development of multi-computers for image processing, pattern recognition, computer vision and scene perception (or indeed for any set of problems) will proceed more quickly to the extent that researchers and potential users can reliably and validly benchmark and assess new systems, uncover and display their strengths and weaknesses, and as a result design, specify and request next-generation improved systems. More immediately, benchmarking plays (or should play) a major role when trying to decide whether to use, or to buy, or to build, or to design, a particular computer system, whether architecture/language or program/algorithm.

Benchmarking is a particular instance of experimental testing and evaluation, which in its turn is the validating portion of the hypothetico-deductive scientific method. Far too often we read that computers have been benchmarked on one particular toy program, one that contains a certain mix of supposedly "typical" instructions. Rarely if ever are we told why an instruction is considered typical, or what evidence there is that the program is of any general significance. Indeed too often it appears that the program was chosen chiefly because it was easy to code!

To adequately benchmark a set of alternative systems one must have a firm grasp of each system's range of potential capabilities, and of the range of problems that these capabilities

can help to solve. But whenever new systems are being developed (as is certainly the situation with today's rapid development of multi-computers for a new research area such as image perception) there are only relatively distant goals and tasks - "recognize and track objects of importance and special interest" - and no clear-cut set of programs, algorithms or instruction mixes. The evaluation process that benchmarking crystallizes is simply a powerful tool, one type of step, toward that goal. This is a situation for successive approximations - that is, for applying the continuing process of science.

## 2. AN OVERVIEW OF PROBLEMS IN DEVELOPING AND USING BENCHMARKS

It is very difficult to develop a set of benchmarks that capture, characterize, and cover a complex (and largely unsolved) set of problems of the sort encountered in image processing/pattern perception. What benchmark (or set of benchmarks) will satisfy everybody who wants a hardware computer, or/and a software program to execute on that computer, that successfully "processes images," "recognizes patterned objects," "sees" in ways we might characterize as "computer vision," and, in general, is a successful "perceiver?"

The total set of problems to be handled is not well defined, or even enumerated or pointed to in any satisfactory way. Different people take different approaches that are more or less suited to different particular problems. Image enhancement uses many different techniques and algorithms, depending upon the particular types of images being processed and their particular problems. Different types of objects to be recognized need different types of information, both object-specific and specific to certain key, signature-like characteristics - e.g., color, texture, sharpness of edges, smoothness of gradients, or jaggedness of contours. Programs that attempt to handle practical real-world applications use problem-specific techniques that may not work nearly as well on different applications, even applications that appear to be closely related.

Almost certainly no single individual is in a position to develop an adequate, comprehensive, and general set of benchmarks. For almost every individual is committed to a particular approach, whether with respect to algorithms/programs or architectures/languages. Indeed, just as a set of benchmarks must fairly cover the range of the benchmarked system's important behaviors, so the set of people who suggest and choose the benchmarks, and who evaluate and interpret the significance of their results, should fairly sample the scientific community's range of approaches, interests, and opinions.

Thus many individuals should be involved in order to achieve a robust, representative, useful benchmarking process. These should obviously include people developing hardware architectures, software operating systems and languages, algorithms (whether general or specific), and applications programs (which are, essentially, complex structures of each relatively simpler algorithms), as well as users of every sort. Ideally, the people included should be a good representative sample of the different approaches, points of view, and needs of the groups involved. But such an ideal is extremely hard, if not impossible, to achieve.

The benchmarking procedures described in this paper attempt to overcome the problem of developing comprehensive, representative, unbiased (sets of) benchmarks by asking each individual researcher and user involved to make her/his own decisions about what to propose, and what to do. That is, researchers and users are encouraged to choose for themselves. The dynamics of the interplay between these individual choices will then build toward a total sample that (in a situation of true free competition) would reflect the total group. The problem, therefore, becomes one of structuring and controlling the group to maximize its free competition, and to minimize any other forms of control or bias.

## 3. THE MAJOR STEPS NEEDED FOR BENCHMARK EVALUATIONS AND COMPARISONS

Benchmarking is, basically, the application of the experimental method in order to evaluate and compare different systems. Ideally, benchmarking might best involve the following major steps:

A) A test, or more realistically a set of tests, must be developed. The goal should be to adequately assess the entire range of behaviors of the systems being examined.

B) These tests should be run on the systems being benchmarked, and the results collected and compared.

C) These resulting evaluations and comparisons should be disseminated widely, to the variety of different kinds of people already participating and potentially interested - the researchers originally involved, the larger group of researchers with related interests, and the still larger group of potential users.

D) Systems should be chosen (for purchase and/or use), or modified, and new systems should be designed and built, making use of information gathered in this process.

E) Suggestions should be solicited for improving the total procedure, and a new set of benchmarks should be built.

F) This whole cycle should be repeated, again and again.

But in reality this is an extremely complicated set of procedures to follow, easily said and rarely done. Indeed, when the systems to be benchmarked are complex (and systems for image processing and pattern perception are extremely complex) it is probably impossible to achieve definitively satisfactory conclusions. Realistically, at best a great deal of work may result in rough approximations that throw new light on the systems benchmarked and the benchmarking procedures.

It is important to emphasise that the sequence of steps A through E outlined above is merely one iteration. This is exactly the same situation that we find in the application of the scientific method to help organize the community of researchers' continuing, never-ending search for what one hopes are closer and closer approximations to the "true" (or least-error-filled) state of affairs.

## 4. A SUGGESTED PROCEDURE FOR IMPLEMENTING BENCHMARK COMPARISONS

The benchmarking process should be implemented by a set of more or less independent individuals. Even if there were a single "Institute of Multi-Computers" with a single overall plan for which all architects, programmers and users worked with diligence and devotion, there would still be a need to get individuals who are deeply committed to their own particular projects to develop and run benchmarks that might make their own systems look bad. And it is unlikely that such a monolithic, planned institute would produce a broad range of good work.

It is assumed that the best climate for good scientific and engineering research and development is one where each individual is encouraged to pursue her/his own ideas and goals as vigorously as possible. From this it follows that the ideal pictured in this paper must result from individual actions, rather than from any grand overall design and direction. The total situation should be structured so that each individual is encouraged, or enticed, or challenged, as much as possible to contribute. And free enterprise should reign; but with

careful and stringent controls that keep every participant equal.

The following outlines the needed steps:

1) A preliminary group composed of interested researchers, especially those who:
   a) have built or are building, or
   b) are designing or examining,
      i) tools: hardware architectures, software languages or operating systems, or
      ii) tasks: software algorithms or applications programs,

should propose and list a preliminary set of benchmarks.

This is probably best done in an informal, inclusive way. That is, everybody interested should be encouraged to participate, and everybody's suggestions should be included. It might be done at a face-to-face meeting, or by mail, or by electronic mail or conference. (The participants in the yearly meetings on "Computer Structures and Image Processing" - of which the 1984 meeting formed the basis of the present volume - played this role in the on-going benchmarking enterprise that motivated this paper.)

2) These preliminary benchmarks should be discussed, mulled over, refined, and standardized, by the large group and/or smaller informal sub-groups and individuals willing to spend more time to take a more searching look at each possibility.

The smaller groups are valuable to the extent that people especially interested in and knowledgeable about the problem will spend the time and thought needed to scrutinize, question, and organize the preliminary set of benchmarks. These small groups are probably best chosen simply by having people volunteer - that is, anybody interested in spending the time and effort should be encouraged to do so. These small groups should form, and change, dynamically, so that each individual involved in the total group is kept aware of their composition, and is encouraged to make suggestions and to take action (e.g., volunteering to serve one of these smaller groups) to correct perceived imbalances.

Here, as at most other steps, the composition of this set of more active people should be scrutinized to see that it is as representative as possible. The larger group, to the extent that it validly reflects the total research community that it samples, is involved enough to actively participate, and is listened to (that is, the total system is kept responsive) should have a strong built-in tendency to keep the smaller groups representative. But it is probably necessary to have some kind of steering committee (or individual) to monitor this, and all other situations, where it is important that things remain as representative and as unbiased as possible.

3) Each person (individually) should rank the preliminary set of benchmarks by importance and relevance, and suggest a mix (probably ordered, with a weight that determines each benchmark's role in that total mix).

These rankings should be made by the larger group, or even, using mail polls, by an even larger sample of the total scientific communities involved. If the procedure is not too cumbersome (as it might not be if all of this could take place via electronic mail on a network to which everybody involved had good access) the larger group should be kept up-to-date about everything that is happening, and be encouraged to send their reactions and suggestions, which the smaller steering committee should study.

4) A priori estimates should be made as to how each benchmark would execute on different

systems. If at all feasible, an estimate should also be made as to the reliability of each estimate.

These estimates should be made by a) the people responsible for the system to be benchmarked, b) a disinterested group, c) people skeptical of the system's capabilities, and d) anyone else who might be interested in the system. When these estimates disagree, the interested people should try to resolve them. If they cannot be resolved, all the discrepant estimates should be disseminated.

If all systems were built and running there would be no need to make estimates. But among the most important purposes of benchmarking is to help in the evaluation of whether to build a new system in the first place, and how to improve the design of a system before it is hardened into program code or hardware chips.

5) These a priori estimates of benchmark results, fully specified and described, along with any comments and discussion, should be disseminated to all those interested (including the researchers developing all the relevant hardware/software systems and applications programs/algorithms).

Ideally, these results should be boiled down to a very short, yet fair and full, summary; followed by a longer description; followed by a detailed appendix-like documentation that included as much raw data as possible. Such a compendium would probably be too cumbersome, expensive and time-consuming if printed on paper and mailed via the Post Office. But it might be extremely useful if handled by electronic mail (albeit with systems that are better than those developed to date, in that they allowed users to browse about, using sophisticated sets of descriptors, to whatever level of detail was desired).

6) All individuals should now be encouraged to choose whether to run benchmarks, and which - whether on their own or other systems, whether getting estimates, timings, simulations, or actual computer runs.

Free choice is of the essence of the benchmarking procedure, just as it is basic to the scientific method in general. Only when each knowledgeable individual is allowed, and encouraged, to decide how best to use her/his time and energy will the community of scientists maximize its fruitful productions and results.

It seems unlikely that busy scientists and engineers could ever be got to participate productively in benchmarking that they didn't feel was worthwhile. They might do it for external reasons (e.g., grant money, joining the group, gaining prestige); but then they would be likely to do so in a half-hearted foot-dragging manner. In any case there almost certainly will (and should) be too many suggested benchmarks for everybody to run all of them. But each individual should choose the specific benchmarks that seem most relevant; and the group should continue to cycle through this procedure.

Each individual should try to be as objective and as disinterested as possible. But that is very difficult if not impossible for most people who are also strongly involved in developing systems and implementing what they naturally feel to be good ideas. So it is crucial that the dynamics of the total interplay between individuals cancel out and overcome individual biases.

This will result in a sampling, with each system benchmarked on the tests chosen by somebody or other. The dynamics leading to these choices will be extremely complex. The people developing systems will have a strong urge to choose benchmarks that demonstrate how well their systems work, and also how much better their systems are than other sys-

tems. Others will choose to benchmark a particular system because they are in direct competition with it, or are skeptical about its claimed performance, or want to provoke the people developing it to demonstrate its behavior on a wider range of problems. All individual should be encouraged, and helped, to challenge each other to contest-like comparisons. Thus the total situation must be structured so that the dynamics of this continuing interaction turns the natural desire of each individual to show particular systems in a certain (good or bad) light into a self-correcting competitive contest-like procedure that builds toward a balanced overall picture.

7) Results of these runs should be collected, validated, and disseminated to everybody.

The group collecting the results of these benchmarks must make sure they have been done carefully, and mean what they say. This is important both to make sure that everbody engaged in benchmarking works as carefully as possible, and also to convince people to whom the results are disseminated to take them seriously.

8) Each individual might challenge particular systems (e.g., by giving an a priori estimate of how well, or how poorly, they would execute particular algorithms).

This whole process should continue to cycle. And it should be made as automatic and as simple as possible.

## 5. POSSIBLE MECHANISMS FOR EFFICIENT IMPLEMENTATIONS OF BENCHMARKING

People should engage in the benchmarking process voluntarily, because they feel it serves a useful purpose toward their larger research goals. Therefore it should require minimal amounts of extra time and effort.

Yearly meetings (like those held on multi-computers/specialized architectures and image processing/pattern perception of which the meeting from which this volume grew is an example) are ideal for bringing people up-to-date, re-examining issues, and getting everybody involved in a variety of formal and informal discussions.

A periodic mailing (e.g., every 3 months) might serve important purposes. It would put things in writing, and keep everybody more up-to-date. And it could go to many more people.

The use of a computer network with good electronic mail and conferencing procedures could, potentially, be a great help - if everybody had good access to the network, and the network had much better systems than those presently available, systems that allowed each person convenient access to whatever types of information, in whatever detail, she/he chose to look at.

### 5.1 Some Miscellaneous Comments

Benchmarks should be encouraged at several different levels, e.g.: a) application, b) whole program, c) single functions, d) simple algorithms, e) individual instructions (with the above decomposed to a certain percent of instructions of each type).

Great care should be taken to keep the group involved open and representative. It should

try to attract new participants (by publicizing this enterprise and inviting feedback), and to keep participation active.

Just as this proposed system is designed to be as adaptive as possible, so that it can make maximal use of each individual's contributions, so the benchmarking process itself should be made as flexible as possible. Therefore people should be invited and enouraged to make suggestions as to how to modify and improve the benchmarking process, and these should be considered using the same procedures used in benchmarking.

The group should try, through publications and any other means, to make these benchmarks well-known and widely used.

Possibly the most important function of benchmarks is to suggest improvements in architectures and algorithms. The publication and dissemination of information of this sort should be encouraged, both in meetings, publications and network groups, and in less formal interchanges.

It is appropriate to use funding to help people do onerous chores for which they otherwise would not have the time, but not to lure them into getting involved in something that they would not otherwise do.

## 6. BENCHMARKING THE WHOLE SYSTEM: HARDWARE/OPERATING-SYSTEM/LANGUAGE/PROGRAM

The benchmarking process is symmetric with respect to the different parts of the total system on which a benchmark is run. We typically think in terms of benchmarking a particular hardware computer by running some benchmarking program(s). But it can be fruitful to change the perspective, and think of benchmarking the program by asking how well a particular computer can execute it. That may seem a bit odd: but it sounds strange only because we (think we) have programs that serve as standards (and are also simple to code, or have already been coded) but we do not have anything close to a comparable "standard computer" much less a "standard parallel multi-computer." But the development of a taxonomy of multi-computers, and of standard exemplars of each sub-species is one of our key research problems, and should go hand-in-hand with benchamrking. The important point is that the computer can serve to benchmark the program, and the program can serve to benchmark the computer. Each benchmarks the other.

More generally, the total system of hardware computer, operating system (some of which may be in hardware), programming language, and program serves as an environment within which some of its parts can be used to benchmark others of its parts. A benchmark always asks the question: "How does X handle Y in the context of $Z_1$ $Z_2$ ... $Z_N$? Usually it asks, "How does machine X handle program Y when run under operating system $Z_1$, using programming language $Z_2$, and ... ?" But it might just as well ask, "How does program X run on machine Y, ... ?

The important point is that each of these components of the total system can be evaluated (benchmarked) with respect to the others.

This is simply the standard situation for the experimental evaluation of systems. We manipulate one (or more) component(s) - often called the "independent variable(s)" - and the resulting effect on a second component - the "dependent variable" - is examined. And there is always a larger context of related but "controlled variables." When a benchmarking program is run on a particular computer that computers performance is being compared with other computers. But it is important to emphasise that such a benchmark merely tells how well each computer executes that particular program - UNLESS the program has been care-

fully chosen and demonstrated to be an adequate sample of some larger domain of programs. Here is where techniques of experimental design, sampling theory and experimental validation are absolutely necessary.

Years of experience with political polls have demonstrated that small samples, when carefully chosen, can indeed be got and used with reasonable accuracy to predict the behavior of very much larger populations. But the problems of developing benchmarks that are adequate samples of the jobs that a computer should be asked to handle appears to be far more difficult than getting representative samples of the population, for example with representative percentages of people of different ages, sex, income. This is because we know far less about the whole population of programs, especially when these are programs that would be run on a new type of multi-computer that would make new types of programs reasonably economical and feasible.

Yet the basic situation is essentially the same. The benchmarking/experimental evaluation procedure therefore actually serves in the more fundamental enterprise of moving toward a clearer understanding of the total system, its parts, and the interactions among these parts. When a benchmark leads to a choice of one computer, therefore that computer is bought, but then the larger set of users find that computer less useful than a second on which the original benchmark did less well, evidence has been acquired that the benchmark but then is not representative; and that very evidence also contains a lot of information that will help improve that benchmark. Unfortunately, this is far more easily said than done, since computers cost too much to buy several to validate the original benchmarks, and rarely can people afford to set up careful experimental comparisons, and collect and analyze the needed data. Or, more precisely, rarely do the people who buy and run the computer feel that the large amount of extra effort needed serves their purposes.

Note that the operating system, the programming language, and any other part of the total system, might also serve the benchmarking process, as either independent or dependent variable. Normally they serve as controlled variables that are always held constant, and therefore without any influence on the situation. But it would be perfectly viable to, for example, change languages to see what effect a different language had on the program's execution. And certainly comparing different operating systems and their effect on performance is a central problem of interest.

The classic way of handling experiments, polls or benchmarks is to have some individual(s) (the experimenter(s), polster(s), benchmarker(s)) exercise complete control over the process outlined above. This can give good clean results in simple laboratory situations, or in real-world studies and polls where everything is relatively straightforward, or reduced to simple straightforward issues. But the dynamically self-correcting procedures outlined in this paper would appear to offer the possibility of richer, more sensitive, more believable, and more believed, results.

## 7. FINAL COMMENT AND BRIEF SUMMARY

It is probably impossible - and undesirable - to develop a universally agreed-to and satisfactory single benchmark, or even a set of benchmarks - especially for new computers that will make possible whole new classes of algorithms and programs to attack new types of problems, including problems not even contemplated before their advent. Benchmarks cannot be static and immutable. To the contrary, just as the experimental procedure pits hardware computer and software program as benchmarks each against the other, throwing light on each, so the results of this benchmarking enterprise reflects and throws light on the

benchmarks used and the whole benchmarking procedure.

The development of good benchmarks, and of a good benchmarking procedure, is in itself an extremely important and interesting research issue. By attempting in the kinds of ways suggested in this paper to involve as many as possible of the total group of interested researchers and users, and continuing to cycle through the process of evaluation, comparison and specification of new benchmarks, it should be possible to get substantially better benchmarks than are typically used today. And it should be possible to develop a continuing interactive, evolutionary procedure that has a number of advantages over the typical specification of one, or a small handful, of static tests whose importance is too often obscure or questionable.

The results of such a benchmarking enterprise should, potentially, have a much stronger effect than is typically the case, because more people will have been involved in achieving them, and will have reason to trust them.

## 13. ACKNOWLEDGEMENTS

The research on which this paper is based was partially supported by NSF Grant No. DCR-8302397.