# Performance Testing of Gridcell-Based GIS

Sherry E. Amundson
University of Hawaii at Hilo
Hilo, HI 96720

## ABSTRACT

The advent of the non-commercial microcomputer-based gridcell GIS brings a host of first-time users who need to know what to expect in terms of processing time. This implies a need for more comprehensive performance evaluation than has been done on GIS in the past. The present study demonstrates the use of the formal performance evaluation methodology in the micro-computer-based setting. It measures the performance of four GIS functions of the OSU MAP-for-the-PC software under a number of varying workload and operating conditions. Commercial performance profiling software is used to monitor program performance internally.

## INTRODUCTION

Several non-commercial PC-based GIS, initially introduced as teaching tools or systems for small government projects, have recently been made available at a modest price. Almost all of them organize data using a gridcell structure.

A thorough evaluation of the performance of these systems -- in terms of processing time and disk utilization -- is needed by a potentially large set of first-time users with a potentially diverse set of requirements. Users need to know what to expect. (Should they wait at the terminal for their results? Should they go out for coffee and come back later? Should they let the program run overnight?)

Processing time varies with the size and complexity of the input data, and with variations in the computing environment. Therefore an assessment of system performance would have to predict processing time under a variety of conditions. Installation managers need performance information when they configure hardware systems or design data sets. Conversely it may be necessary to plan a GIS application to fit within existing hardware or data constraints.

The performance assessment of commercial GIS has traditionally taken the form of application-specific benchmark tests commissioned by individual user agencies. For the sake of economy, measurements are made on only those GIS functions that the agency intends to use most frequently, and the functions are measured using a limited number of real-world data sets that represent typical workloads for the installation or extremely heavy 'worst-case' conditions. The test designer does not have access to the source code because it is proprietary. Likewise the designer owns the test design, and the results are kept confidential.

Application-specific testing is inadequate to evaluate non-commercial GIS; a diverse set of users needs to refer to the same set of results. All GIS functions in the function set must be evaluated. More important, the testing methodology must enable users to predict performance levels under a wide range of data and operating conditions.

## Performance Evaluation Methods

A formal evaluation methodology is used in computer science to analyze and improve computer system performance. It is comprised of a set of quantitative procedures that measure performance in terms of time spent and space utilized in a system. In common practice performance times are measured with internal probes while an application runs on the system. The probes (usually calls to the system clock) may be placed selectively within the program to time specific sections of the code and discover where the system "spends its time".

Performance evaluation studies are applied to entire computer systems, or hardware and software complexes. The interaction between software instructions and the way they are executed in the hardware is system dependent; it is commonly understood that a program cannot be measured outside the context of the computing environment. Program A might run more efficiently than Program B in one environment and less efficiently than B in another environment (Ferrari, 1978).

In a similar fashion the performance of a system may be expected to vary with the application. In this context the term "application" represents a specific workload (in the form of a specific set of tasks and a given data set) that a system is to process. A single evaluation project consists of a number of measured runs which process controlled versions of a synthetic workload. The workload is designed to be modified according to specific parameters; individual performance-influencing factors may be isolated and modified while other characteristics are held constant (DeWitt, 1985; Heidelberger, 1984). This technique can support a full factorial design which measures performance under all combinations of selected factors in an n-tuple structure (Ferrari, 1978).

The complete set of runs tests the strength of the factors as performance predictors. The formulation of test objectives should be based on extensive knowledge of how the system works. Without prior familiarity with the internal organization of the system, an evaluator would not know which facets of the system to test (Heidelberger, 1984).

## Purpose of the Study

Compared to the large commercial GIS, the new non-commercial microcomputer-based GIS are uniquely 'testable'. They can be monitored internally because the source code is readily available. An examination of the code also reveals possible performance-influencing factors. There is no reason to limit testing to specific applications or to use only 'typical' data loads, and test results can be made available to the entire user community. In addition, the new PC-based systems use a gridcell structure, and the regularity of gridcell processes would indicate important and highly reliable predictive factors in the data load.

This paper demonstrates the use of performance evaluation techniques (in the form of synthetic data set design, internal monitoring of the code, and the use of a factorial testing scheme) in the setting of the non-commercial PC-based GIS. Tests are conducted on one of the Ohio State University versions of the Map Analysis Package, OSU MAP-for-the-PC. This particular GIS was selected because the source code was already in hand. A method for testing the speed of individual GIS functions is introduced and applied to four of the functions.

## Measurable Factors vs. Functional Factors

"Performance" refers to how well a system works. It is based on measurable factors within the system (in terms of the utilization of system resources) and on functional factors such as ease of use, correctness, availability, reliability, training, etc. (usually measured in terms of human resources). In both cases most resources are represented by some form of time expenditure. System resource expenditures might be expressed as throughput or turnaround time; human resource expenditures might be the time required for data collection and editing, or the time required to develop applications. Some functional factors cannot be measured at all; they are simply verifications that the system possesses specified features.

Literature about performance evaluation acknowledges that human resource expenditures are at least as important as computer resource expenditures in evaluating system performance (Stonebraker, 1985). In fact these elements do assume a major role in application-specific GIS benchmark studies, because production schedules are called into play (Goodchild and Rizzo, 1986; Tomlinson, 1981; Greenlee et. al., 1986). However the technical performance literature deliberately excludes the functional factors from formal study because they do not lend themselves to quantitative measurement (Heidelberger, 1984). If performance evaluation methodology is to be used to improve GIS tests, it is more likely to be in the arena of internal measurements of the system itself.

## OSU MAP-FOR-THE-PC

### Operating Environment

OSU MAP-for-the-PC runs on an IBM PC/XT or PC/AT or PS/2 or equivalent machine, using the MS-DOS or PC-DOS operating system. The machine must have at least 512 KB of memory, a hard disk, and one floppy disk drive. 640 KB of memory is required if grids larger than 28,000 cells are to be used. An appropriate math co-processor (the 8087 for the PC/XT, the 80287 for PC/AT-class machines or the 80387 for "386" machines such as the IBM PS/2 80) is optional but highly recommended, especially with the slower machines.

The microprocessor in the machine -- the 8088 chip in the PC/XT, the 80286 chip in the PC/AT and the 80386 chip in the "386" machine -- determines its processing speed. The pace at which instructions are processed is measured in terms of a steady beat supplied by a **clock generator**, which beats at 4.77 MHz in the PC/XT, 6 to 20 MHz in the 80286-based machines, and 16 to 25 MHz in the 80386-based machines. Clearly the processing speed of a system has a strong influence on the performance of any GIS application.

Math co-processors may have a strong impact on the execution speed of programs because they vastly accelerate the processing of floating point operations. Co-processors are not essential to program execution, and their installation is optional. In OSU MAP-for-the-PC most mathematical operations are performed in integer arithmetic. However floating point operations are concentrated in the implementation of a few GIS functions, and users who need to call heavily on these functions may find that a co-processor is beneficial.

Ohio State provides two versions of OSU MAP-for-the-PC based on co-processor options. One version requires a math co-processor and the other version emulates a co-processor regardless of the hardware configuration. The emulation version detects the presence or absence of a co-processor and accesses the chip if it is present.

### Program Structure

The central module of OSU MAP-for-the-PC is a large command interpreter. After it parses a user command it calls the appropriate subprogram to implement the spatial data handling function that has been requested and then writes the results to the database. The function-handling subprograms can be viewed as independent and unrelated spokes extending from the command interpreter "hub".

The gridcell structure simplifies processing because of its regular distribution of data points and because (for many GIS procedures) all cells must be examined in turn regardless of the cell value. In addition the region boundaries in OSU MAP-for-the-PC must be identical for all the layers. This regularity in the data organization and the data handling processes implies that the number of cells in the grid has a strong influence on performance. It also suggests that the relationship between performance and the number of gridcells is linear.

## TEST DESIGN

### Functional Level Tests

According to Goodchild and Rizzo (1986) tests should be performed at the level of the spatial data handling function (i.e. they should measure the performance of entire functions) because it is at this level that different GIS packages must be compared. Because they lack a formal command language, the public domain microcomputer-based GIS operate at the level of the "atomic" function (polygon overlay, reclassification, etc.). In the present study each atomic function is subdivided into component segments at the program module level, which is a more detailed level than the one suggested by Goodchild and Rizzo. Modules are measured individually, and the measurements may be summed to find the total measurement for the function.

## Performance-Influencing Factors

The performance-influencing factors were chosen to test simple but potentially strong relationships, with a minimum of interaction among the factors. Three factors were selected, one from the workload characteristics and two from the operating environment: the size of the grid, the presence or absence of a math co-processor, and the selection of the microcomputer itself.

The influence of the simple gridcell structure on performance has already been discussed. It is postulated that the volume of data in a given layer (i.e. the size of the grid) would exert an overwhelming influence on the execution speed of GIS functions that process each cell in turn.

The most influential factor in the operating environment appears to be the choice of the microcomputer. The PC/AT, with its 80286 microprocessor and its 8 MHz clock speed is reported to run almost 8 times as fast as the PC/XT with its 8088 microprocessor and its 4.77 MHz clock speed. The presence or absence of a math co-processor was chosen as a factor because the effects of a co-processor were unknown; the program makes little use of floating point arithmetic.

A factorial design was used to test the four GIS functions under a number of factor values. The microprocessor factor was tested in PC/XT and PC/AT 8 MHz configurations. The co-processor factor could be configured as either "on" or "off"; the chip could be either present or absent. Tests on the PC/AT machine were run with and without an 80287-10 math co-processor running at 10 MHz; tests on the PC/XT machine were run with and without an 8087 math co-processor running at 5 MHz.

The workload factor was tested at three levels, represented by grids of 8,000, 16,000 and 24,000 cells. Although the number of levels is too small to support statistical analysis of the results, it is sufficient to suggest a pattern in performance. The small number was used to keep the factorial design to a reasonable size. As it was, the triple of two (microprocessors) times two (co-processor configurations) times three (workload levels) resulted in twelve tests for each of the four functions.

## Workload Design

GIS functions were chosen for study based on the way they handled data and based on their ability to illuminate performance-influencing relationships. The **GRID** function, one of the data entry functions, reads rows of data into memory cell by cell from an external ASCII data file. It also transfers the appropriate data into the four binary data files on the disk. It was chosen because it handles all cells in the same manner, and because it was perceived to have a relatively long duration. This is relevant because the longer running functions cause greater user uncertainty about waiting times.

The **SCORE** function loads two data layers into memory and derives a complete cross tabulation of their cell values. Like the GRID function, it executes the same process for each cell. In addition it contains a limited number of floating point operations and it spends considerable time drawing tabular output on the screen. It would be worthwhile to test this output procedure under different conditions. The **MULTIPLY** function loads two layers into memory, performs polygon overlay by multiplying the corresponding cells of the layers, and writes the result to a new layer on the disk. It was selected for the study because it treats all cells the same and because overlay is perhaps the most important class of GIS function. The **CONTOUR** function creates and displays a contour map of a layer in vector mode. It was selected for the study because it relies heavily on floating point calculations and because it can be used to show performance variations caused by the presence or absence of a math co-processor.

## Synthetic Data Set

The objectives of the synthetic data set design were 1) to control the size of the grid and 2) to hold constant all other data characteristics as much as possible.

Three databases were constructed with grids of 100 x 80 cells, 200 x 80 cells and 300 x 80 cells, and two layers were created in each database. One contains a series of 80 vertical stripes running the length of the map layer, one cell wide (titled STRIPE). The other is a two-color test pattern (titled ZORRO) in which the proportion and distribution of the values is the same in each database. A figure "X" extends from the four corners of the layer. It intersects a figure "Z" whose top and bottom bar divide the layer into thirds.

STRIPE presents a situation in which each cell has a different value from the next, and the rows are identical in the three databases. The difference among the three grid sizes lies entirely in the number of rows. This structure insures that each row is processed in an identical manner, and that the maximum number of variations in cell values is presented. ZORRO was chosen for its simplicity, for the fact that the pattern extends throughout the layer, and because the distribution of cell values can be reproduced for databases of any size. Among the three databases in the study, the three versions of ZORRO are vertically proportional and horizontally identical. In both layers, the effects of several types of data complexity are held constant: the number of different cell values, the distribution of cell values, and the number of horizontal runs in a row.

A third layer, called ELEV, was derived from the ZORRO layer for the purpose of producing contour maps. The SPREAD function of OSU MAP-for-the-PC was used to convert ZORRO into a map of distances from the test pattern. The result consists of a set of concentric bands of cells of equal value surrounding the "Z" and the "X" pattern. The value of the cells in each band reflects its distance from the test pattern.

## Test Scripts

During each run the program was monitored while it executed a short script of operations. Different scripts were used to test different functions, and in most cases they were comprised of a single command followed by the command to exit the program. In the script for the GRID function, GRID is asked to initialize the STRIPE layer by reading cell values from a raw data file. The script for the MULTIPLY test requires the multiplication of the STRIPE layer by the ZORRO layer. The SCORE script calls for a cross tabulation report regarding STRIPE and ZORRO. The CONTOUR function is required to produce a contour map of the ELEV layer, specifying ten contour levels in all.

### Performance Monitor

The Pfinish* performance monitoring software was used to measure performance times and to count the number of times different sections of the code were entered. Pfinish allows the user to define and measure blocks of code as large as the entire program and as small as a single executable statement. The user may also request a number of output reports which aggregate performance test results. The most prominent report in the profile, which record the number of times each block of code is visited along with the combined duration of the visits. Timing in Pfinish relies on the hardware clock, which has a resolution of 18.2 ticks per second (Phoenix Technologies, 1986).

The user lists the blocks to be measured and requests output reports in a batch file. When a test is run, both Pfinish and the program being tested are loaded into memory. Then the program being tested runs while Pfinish records the performance information that was requested in the batch file, and the appropriate output reports are generated.

The fact that Pfinish is resident introduces certain artifacts. It slows down the apparent processing speed of the program (however this does not affect the internal execution time of the program). Pfinish also occupies space on the disk, and this reduces the available space for the program. During the tests on OSU MAP-for-the-PC, the layer size in the GIS had to be reduced to 25,000 cells in order to accommodate the performance monitor.

## PERFORMANCE MEASUREMENTS AND RESULTS

### Performance Indices

Blocks were defined at the level of the program module. This decision was based on the fact that each function is associated with its own module (sometimes two or three modules) which does nothing but implement that function. A different batch file was used to test each of the four functions, GRID, SCORE, MULTIPLY, and CONTOUR, and the same batch file was used to manage all twelve tests of a single function. Each batch file was constructed to measure the important modules that implemented the function.

Three modules or sets of modules played a key role in each batch file. One of these was always the main module or modules that implemented the function. The second was the module which transfers blocks of data to and from the disk. The final required module is the one that waits for user input from the keyboard. It had to be included because a method was needed to subtract out the time spent waiting for user input.

## Preliminary Results

The entire battery of tests consists of twelve different tests for each of four functions: 48 tests in all. Ordinarily each of these would be run several times and results would be expressed as the means and variances of a number of runs. This procedure is necessary to reduce the **noise**, or the variation, produced by the clock resolution. Because the current project is limited in both size and scope, multiple runs were not attempted. Instead, the modules that primarily implement each function were run five times together at the outset to give an indication of the amount of variation that could be expected. In all cases there was less than 5% variation.

## Results

The analysis of results was limited to scrutiny of a small number of primary modules. It is in these modules that performance-influencing factors can be measured. A list of the modules that implement each function is shown below.

| Function | Modules |
|---|---|
| GRID | INPUTR |
| SCORE | SCORE, PSCORE, OUTSTR |
| MULTIPLY | OVRLAY |
| CONTOUR | THREAD, COTOUR |

The graphs in Figures 1, 2, and 3 show the timing results for the GRID, SCORE, MULTIPLY and CONTOUR functions. In each graph a line connects performance times of modules processing the 8,000-, 16,000-, and 24,000-cell workloads. Each line represents a different combination of microcomputer and co-processor factors in the operating environment. All results are measured in clock ticks.

In all the modules, performance in the PC/XT environment appears to be four to five times the duration in comparable PC/AT configurations. The primary interaction among the factors is tied to the machine speed. In all cases the variation associated with the grid size factor and with the co-processor factor is more pronounced in the slower machine environment. The trend is depicted in the form of steeper curves showing the variation due to data volume. It is depicted in the form of greater distance between curves showing the differences due to the presence or absence of a co-processor.

The math co-processor appeared to have almost no effect in either the GRID or the MULTIPLY function, and this is because neither function relies on floating point processes at all. However the CONTOUR function makes extensive use of floating point operations, and the results of the CONTOUR tests show a substantial effect of the co-processor. The THREAD module performed ten times as fast on a PC/XT with a co-processor than on a PC/XT with no processor installed.

Because each line on the graphs represents only three observations, the lines cannot be analyzed to predict performance for a full range of workloads. However, the graphs indicate that strong linear relationships exist between the size of the grid and the performance time.

## SUMMARY

These tests represent a demonstration of testing techniques for microcomputer-based gridcell GIS. The results show promising patterns, but conclusive results would have to based on a more extensive testing regime. All tests should be run a number of times, and results should be expressed as the means of the output of the runs. In addition, a larger number of observations is needed to properly test the influence of the data size factor.

A further variation in the workloads is needed to represent a more complete assortment of predictive factors: the variety of distinct cell values, the number of horizontal runs, the probable distribution of cell values, user-defined search radii, etc. The results of such a study can be analyzed using simple regression techniques. Outcomes for individual modules might then be summed to arrive at performance predictions for entire functions.

## BIBLIOGRAPHY

Brickner, R. G., 1986. "An Execution Profiler for the PC," PC Tech Journal, Vol 4 (11): 120-130.

________, 1987. "Execution Profilers for the PC, Part 2," PC Tech Journal, Vol 5 (2): 166-171.

DeWitt, D. J., 1985. "Benchmarking Data base Systems: Past Efforts and Future Directions," Data base Engineering, Vol 8 (1): 2-9.

Ferrari, D., G. Serazzi and A. Zeigner, 1983. Measurement and Tuning of Computer Systems. Prentice-Hall, Englewood Cliffs, N.J.

Goodchild, M. F. and B. R. Rizzo, 1986. "Performance Evaluation and Workload Estimation for Geographic Information Systems," Proceedings, Second International Symposium on Spatial Data Handling.

Greenlee, D. D., J. W. Van Roessel and M. E. Wehde, 1986. An Evaluation of Vector Based Geographic Information Systems at the EROS Data Center, U.S.G.S. EROS Data Center, draft report.

Heidelberger, P. and S. S. Lavenberg, "1984. Computer Performance Evaluation Methodology," IEEE Transactions on Computers, Vol c-33 (12).

Marble, D.F. and L. Sen, 1986."The Development of Standardized Benchmarks for Spatial Data base Systems," Proceedings, Second International Symposium on Spatial Data Handling.

Phoenix Technologies, Ltd., 1986. PFinish User's Manual. Norwood, MA: Phoenix Computer Products Corporation.

Stonebraker, M., 1985. "Tips on Benchmarking Data Base Systems," Database Engineering, Vol 8 (1): 10-18.

Tomlinson, R. F. and A. R. Boyle, 1981. "The State of Development of Systems for Handling Natural Resources Inventory Data," Cartographica, Vol 18 (4): 65-95.

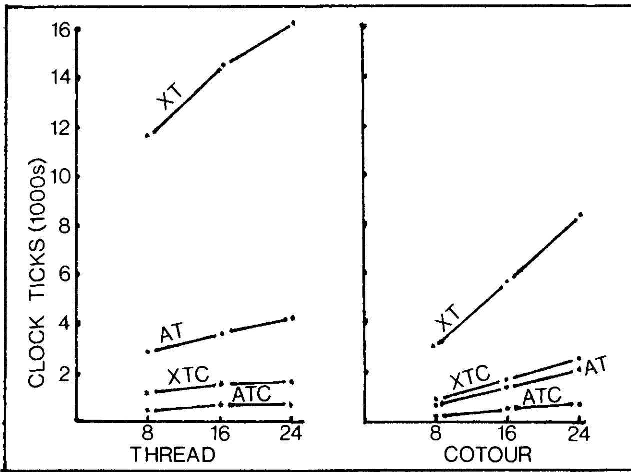

FIG 1
TIMING
RESULTS
FOR
CONTOUR

ATC, XTC are co-processor configurations

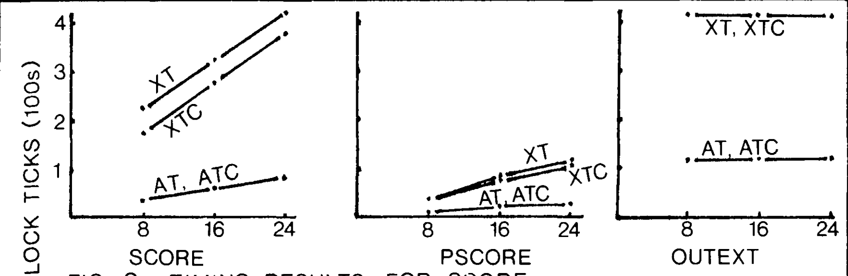

FIG. 2 TIMING RESULTS FOR SCORE

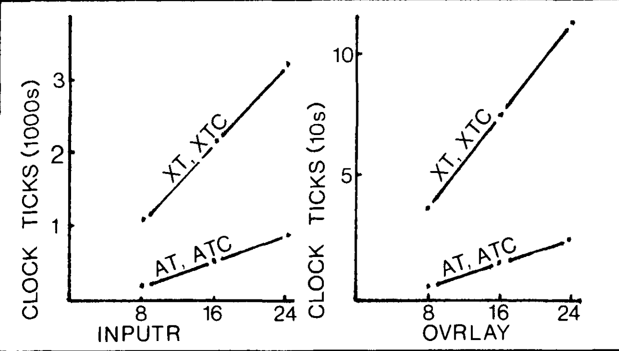

FIG 3
TIMING RESULTS
FOR
GRID AND
MULTIPLY

DATABASE SIZE (1000s of cells)