An Overview of the WPI Benchmark Suite *

David Finkel Robert E. Kinicki Jonas A. Lehmann

Department of Computer Science
Worcester Polytechnic Institute
Worcester, MA 01609
mach@cs.wpi.edu

The November 1990 issue of Performance Evaluation Review included a number of articles and opinions on the merits of commercial benchmark suites. In the spirit of continuing this discussion, we present here a brief introduction to the WPI Benchmark Suite.

The Worcester Polytechnic Institute Mach Research Group has developed a set of benchmarks designed to evaluate the performance of Unix-like operating systems. This series of C programs is designed to compare BSD Unix, System V.4 Unix, Mach 2.5, Mach 3.0 and OSF/1 running on the same hardware platform. This concept of comparing operating system performance differs from the intent of most other available benchmarking programs and commercial suites. Many benchmarks ([SPE90], [CUR76], [DON87], [SMI90], and [WEI84]) emphasize CPU intensive applications because their objective is to use these measurement tools to quantify hardware computational speed. Because our focus was assessing the performance implications of the Mach operating system design, we surveyed published Mach-specific benchmarks ([BLA89], [FOR89], [GOL90], and [TEV87]) and found that these performance studies were predominantly low-level tests which repeatedly exercised a single system call or system service.

The design philosophy of our benchmark development was to have a two-tiered set of programs which is identical to what Ponder [PON91] calls synthetic and diagnostic benchmarks. The major programs are high-level synthetic benchmarks designed to reflect the usage of operating system services found in user application programs, and the low-level benchmarks consist of individual system functions which can be used to isolate and identify specific weaknesses in operating system designs. This article discusses only the application level programs in the WPI Benchmark Suite.

Synthetic programs have a number of advantages for comparing operating system behavior on a variety of platforms. In synthetic benchmarks, the mix of system calls can be precisely controlled. Furthermore, the programs can be parameterized, allowing longer runs and larger data sets when the benchmarks are run on large-scale systems. With a suite of synthetic programs, it is possible to collectively cover the range of standard system services. Most importantly to our objectives, synthetic benchmarks simplify the task of porting the suite to various versions of Unix. In this first version of the WPI Benchmarks, specific Mach system calls are avoided and the timing mechanisms were designed to adjust to variations in Unix systems.

The following is a brief explanation of the six programs in the WPI Benchmark Suite. The five programs with an S prefix are truly synthetic programs, while Jigsaw is a test program designed to utilize specific system services. A more detailed discussion of the methodology used to create the synthetic programs can be found in [FIN90].

*This research was supported by a grant from the Research Institute of the Open Software Foundation.

1 Scomp

This program creates a mix of Unix system calls which are designed to mimic system resource usage of gcc compiling gcc. Data was collected by using gprof to monitor the procedure calls used when gcc compiles itself. From the procedure call information, Scomp was synthesized to recreate the structure of gcc to some extent and to issue Unix system calls in a pattern similar to gcc.

2 Sdbase

This client-server database benchmark uses TCP/IP sockets to communicate between a single server and multiple clients. The system is composed of a concurrent database server, a number of client processes, a database generation program, a large database file and programs to analyze server and clients.

The requested services include reading a random record from the database, modifying a record and appending new records. The client activity is based on a job mix discussed in Byte Magazine [SMI90].

3 Sdump

Modelled after the Unix dump program, this benchmark reads a set of one Mbyte files from a directory representing a file system and transfers the data to a process emulating a tape device. The transport of the data from the reading process to the writing process is done via Unix pipes. The writing process can either dump the merged file to a null device or to disk. The number of files dumped is a run-time parameter.

4 Sftp

By emulating an FTP transfer, Sftp is designed to show transmission rate performance for various buffer sizes. The host machine participating in the TCP/IP transfer runs a server background task which responds to remote client requests for file transfers.

5 SXipc

SXipc emulates network traffic between an X server and a set of X clients. Utilizing eight different X client types measured by Droms [DRO90], SXipc is a script-driven program which allows for a large number of local and remote clients in combination to issue requests to the X server. This program currently characterizes the communication behavior of X. Efforts to include the I/O activity associated with X windows or to incorporate the CPU activity of servicing window requests are not included in this benchmark.

6 Jigsaw

Jigsaw solves a mathematical model of a jigsaw puzzle [GRE86] where the four sides of a puzzle tile have a recognizable relation with the sides of neighbor tiles in the solved puzzle. The benchmark builds a puzzle, scrambles tiles, and records the time required to solve the jumbled puzzle. Puzzle size is variable. With tile sizes of 1 or 4 kbytes, this benchmark is targeted at studying memory allocation and paging behavior.

The WPI Benchmarks are in the public domain, and are available by sending e-mail to mach@cs.wpi.edu. We welcome your comments on the benchmarks.

References

[BLA89] D.L. Black, R.F. Rashid, D.B. Golub, C.R. Hill, and R.V. Baron, "Translation Look-aside Buffer Consistency: A Software Approach", *Digest of Papers, COMPCON Spring '89, Thirty-Fourth IEEE Computer Society International Conference: Intellectual Leverage* , (1989), 184 - 190. [Also Carnegie Mellon University School of Computer Science Technical Report, CMU-CS-88-201].

[CUR76] H.J. Curnow and B.A. Wichmann, "A Synthetic Benchmark", *The Computer Journal*, 19 (1976), 43 - 49.

[DON87] J. Dongarra, J.L. Martin and J. Worlton, "Computer Benchmarking: Paths and Pitfalls", *IEEE Spectrum*, July 1987, 38 - 43 , .

[DRO90] R.E. Droms and W.R. Dyksen, "Performance Measurements of the X Window System Communication Protocol", Bucknell University Computer Science Department Technical Report #90-9, March 1990.

[FIN90] D. Finkel, R.E. Kinicki, A. John, B. Nichols, and S. Rao, "Developing Benchmarks to Measure the Peformance of the Mach Operating System", *Proceedings of the Usenix Mach Workshop*, Oct. 90, Burlington, Vt., 83-100.

[FOR89] A. Forin, J. Barrera, M. Young, and R.F. Rashid, "Design, Implementation and Performance Evaluation of a Distributed Shared Memory Server for Mach", Carnegie Mellon University School of Computer Science Technical Report, CMU-CS-88-165. [Also published as "The Shared Memory Server", USENIX Winter Conference, San Diego, 1989.]

[GOL90] D. Golub, R. Dean, A. Forin, and R. Rashid, "Unix as an Application Program", *Proceedings of the USENIX Summer Conference*, June, 1990, 87 - 95.

[GRE86] P.E. Green and R.J. Juels, "The Jigsaw Puzzle - A Distributed Performance Test", *Proceedings of the 6th International Conf. on Distributed Computing Systems*, May 19-23, 1986 288 - 295.

[PON91] C. Ponder, "Benchmark Semantics", *Performance Evaluation Review*, Vol. 18, No. 4, April 1991, 20 - 23.

[SMI90] B. Smith, "The Byte Unix Benchmarks", *Byte*, March 1990, 273 - 277.

[SPE90] "Benchmark Results ", *SPEC Newsletter*, Vol. 2, No. 2, Spring 1990.

[TEV87] A. Tevanian, "Architecture-Independent Virtual Memory Management for Parallel and Distributed Environments: The Mach Approach", Ph.D. Thesis, Carnegie Mellon University School of Computer Science, Dec., 1987. [Also Carnegie Mellon University School of Computer Science Technical Report, CMU-CS-88-106].

[WEI84] R.P. Weicker, "Dhrystone: A Synthetic Systems Programming Benchmark", *Comm. of the ACM*, 27 (1984), 1013 - 1030.