

Computer Architecture Laboratory

Benchmarking Experiments

Objectives

- To gain some experience in benchmarking
- To compare the performance of machines in common use at PSC
- To understand the influence of compilers in determining machine performance

Method

Using only one program - the synthetic benchmark named Dhrystone - we will compare two machines: a server-quality PC and a desktop PC. We will also examine the effects of compilers and the possible influence of the operating system on machine performance. For this lab, you'll use Turing and a PC in the systems lab. For each machine, find the following information:

Processors (type and number) Clock Memory Cache

and put the information in Section 1 of the answer sheet for this lab.

On "oz", in directory ~wjt/Arch, there is a file "dhry.c". This file must be copied onto both machines (using ftp if necessary). The file is the standard dhrystone benchmark which accompanied the textbook "Computer Architecture, A Quantitative Approach" by Hennessy and Patterson slightly modified by your instructor. H&P got it from the "usenet". It is a translation into 'C' of the Ada version of Dhrystone initially published by Reinhold C. Weicker (see "author" in program documentation for a reference.)

This program is written to use the timing functions on Unix machines with compilers such as gcc and c89, and the timing functions of MS VisualC++ on a wintel system. To use these implementation-dependent include files for timing, there is a definition in the program that must be selected. The definition is immediately below the opening documentation; use your favorite editor to remove the comments around the "define" for the type of compiler. There are three defines: GCC, POSIX and VC. For example

```
#define GCC /* use this for the gcc compiler on a Unix machine */
```

In addition, to get reasonable timing results the program should execute for a few seconds. This is done by choosing the number of loops of dhrystone are executed. These may also be selected in a "define" statement. On Turing, removing the comment markings, select:

```
#define LOOPS 5000000 /* use this for fast Pentium */
```

Adjust the number in LOOPS until you get a run that takes about 10 seconds of machine time. You must select the proper value; choose fewer LOOPS for slower machines

To compile a program in Unix you must invoke one of the "C" compilers. We currently have two Unix compilers gcc and c89. The former comes with FreeBSD and is from the Free Software Foundation. The latter is FreeBSD's attempt to comply with the Posix standard for operating systems. You invoke the compiler with the command:

gcc [option] file or c89 [option] file

There is a Unix manual page for these compilers which you can obtain with the command: `man gcc` or `man posix` which will give you more information about these two c compilers.

The [option] field in gcc allows for various types of optimization. They are:

- O0 All off (that is oh zero)
- O1 Fast optimizations
- O2 Fast + global microcode optimization
- O3 All on

An example of a compilation statement would be:

```
gcc -O1 dhry.c
```

The c89 compiler has only one level of optimization and is invoked as follows

```
c89 filename.c            no optimization  
c89 -O filename.c        optimization on
```

The file produced by either process is `a.out` and it is already linked. It may be executed by simply typing its name.

Built into this benchmark is an output of the total time needed, the number of loops, and the number of dhrystones/sec (LOOPS/TIME). Each loop is called "one dhrystone" and so the number of dhrystones does not matter as much as the performance which is in dhrystones/sec. On time-shared machines, swapping (page faults) and other factors are not reproducible on each run and successive runs of the benchmark may yield slightly different results. Therefore, several runs may be necessary to form a meaningful average.

Experiment 1. This experiment compares the relative performance of compiler optimizations. Compile and run the benchmark on Turing using the gcc compiler and each of the compiler options. You may need to average several runs to get meaningful data. (Also, check to determine if the benchmark is sensitive to the number of loops that you defined in the `#define LOOPS =` statement.)

Repeat for c89.

Put the results in Section 2 of the Answer Sheet and complete the questions asked.

Experiment 2. This experiment compares the relative performance of gcc and c89. Compile and run (several times to get an average) the benchmark on Turing using the gcc compiler with the -O3 option. On the answer sheet enter the results for the gcc compiler along with the results you got for the -O3 option of the cc compiler from part 2. Complete the questions in part 3 of the answer sheet.

For experiments 3-5 you will need a PC which has both a MS Windows OS and MS VC++ and Linux (or FreeBSD) and gcc. The machines in the Programming Lab have all this software. You'll be given the login name and password in class. This is the first year we've done this lab in this manner. Work together and communicate with classmates to solve any "systems" problems that arise.

Experiment 3. This experiment compares the performance of Turing and a PC. Port Dhrystone to a machine in M312 and modify dhry.c by changing the "define" for the loops and machine name in the "defines" section of the program. Compile the program using gcc with the -O3 option and make the same measurements that you made on "Turing". In part 4 of the lab sheet, compare the two machines: Turing and a PC.

Experiment 4. This experiment changes gears a bit and examines the Microsoft Visual C++ compiler's optimizations. In Dhrystone, change the defines to VC (which is MS VC++), compile the code to generate an .exe file and finally execute the file. The Microsoft compiler has several compiler options; examine each one. Put your results in part 5 of the answer sheet.

Experiment 5. This experiment compares Unix with Microsoft. One run of Dhrystone will use either Linux with the gcc compiler (with all options on) and the other will use Microsoft Windows XP (or other Microsoft OS) and Visual C++ with its best compiler options. Run Dhrystone in each "world" and measure the relative performance of the two worlds. Complete part 6 of the answer sheet.

Answer Sheet

Name _____

Report all performance results in Dhrystones/sec

1.

	Turing	PC
Number of processors		
Type of processors		
Clock speed		
Main Memory		
Cache Memory		

2. gcc -O0 _____
gcc -O1 _____
gcc -O2 _____
gcc -O3 _____

Using gcc -O0 as the "base machine" how much faster is the machine with compiler option? Gcc fully optimized is _____ % faster than gcc with no optimization.

- c89 _____
c89 -O _____
Optimized c89 is _____ % faster than unoptimized c89.

3. gcc -O3 _____
c89 -O _____

Which compiler is faster (with this option)? _____ By what percentage? _____

4. turing: gcc -O3 _____
PC: gcc -O3 _____

Rank the machines 1, 2 with 1 being the fastest.

1 _____

2 _____

1: 2 _____ is _____ percent faster than _____

Is the increase in speed directly related (linearly related?) to clock speed? Explain.

5. Comparing optimization types inside the VC++ compiler:

Optimization	Dhrystones/sec
_____	_____
_____	_____
_____	_____
_____	_____

_____ is faster than _____ by _____ %.
(fastest) (slowest)

6. Comparing Unix/gcc and WinXX/VC++:

Linux with gcc yielded _____ dhrystones/sec.

Windows XP with VC++ yielded _____ dhrystones/sec.

_____ is faster than _____ by _____ %.