

Assignment 1: Bone-headed Timer

Due: 01/12/05, Multiplication Factor: 1.0

Your first assignment is to complete a simple timer for a dot product kernel (the kernel can be found in `ddot.c`), which works under both Solaris and Linux, with varying compilers. To do so, create a subdirectory, cd into it, and issue (on either `program` or `linprog`):

```
cp ~whaley/teach/cis5930/ASG1/* .
make archdirs
```

This will create a separate architecture-specific subdirectory for compiling different versions of the timer. Examine the Makefile, and understand what it is doing. Why is it advantageous to have separate binary directories, and why do we use the same makefile with different includes for each?

You will send a completed `dottime.c` to `whaley@cs.fsu.edu` by 10AM on the due date, and I expect it to work in the framework as given, so change only this file for correctness. I have filled in the basics of a crude timer for you, and you must complete two routines:

1. `my_time()`: This routine presently has the default timing methodology only: CPUtime using `getrusage`. Use this default timer to get things working, and then add different types of timers, which can be selected at compile time using the indicated CPP macros. You'll have at least 4 different timers, two for CPU and two for WALL times. Under Linux, only one for each will be usable, but under Solaris, there will be the option of using SUN's higher resolution interface. All timers must work like a stop watch, in that the first call returns an arbitrary number, which is subtracted from a later invocation to give elapsed time. See the file for further details.
2. `DoTime()`: This routine does the actual timing of `ddot()`. I have left in my variable and function declarations, feel free to change. Make sure your timer functions as discussed. In particular, it must repeat the operation in order to perform at least `mflop` MFLOPS, and should return the time (on average) of a single call to `ddot()`.

Useful information can be found by:

```
man [getrusage,gettimeofday,uname]      # general info
man [gethrtime,psrinfo]                  # solaris info
cat /proc/cpuinfo                         # linux
```

Run the timers in each of the created subdirs on the appropriate machines (Linux: `linprog`; Solaris: `program`). Do you understand the output? What does/does not make sense to you? Do the timings make sense, and can you make them repeatable? Do you see differences among the timers, and do such difference depend on the setting of `mflop`? Are there differences amongst platforms/compilers, and do they act like you would expect? Can you vary the results in a meaningful way by changing the compiler flags?

Part of this assignment is to familiarize you with how we'll be doing things, so be sure to fully understand the build process, and what all the files are doing.