Loop Optimizations - Exercises*

Charles Severance
Kevin Dowd

This work is produced by The Connexions Project and licensed under the
Creative Commons Attribution License †

Exercise 1
Why is an unrolling amount of three or four iterations generally sufficient for simple vector loops on a RISC processor? What relationship does the unrolling amount have to floating-point pipeline depths?

Exercise 2
On a processor that can execute one floating-point multiply, one floating-point addition/subtraction, and one memory reference per cycle, what's the best performance you could expect from the following loop?

```
DO I = 1,10000
  A(I) = B(I) * C(I) - D(I) * E(I)
ENDDO
```

Exercise 3
Try unrolling, interchanging, or blocking the loop in subroutine `BAZFAZ` to increase the performance. What method or combination of methods works best? Look at the assembly language created by the compiler to see what its approach is at the highest level of optimization.

NOTE: Compile the main routine and `BAZFAZ` separately; adjust `NTIMES` so that the untuned run takes about one minute; and use the compiler's default optimization level.

```
      PROGRAM MAIN
      IMPLICIT NONE
      INTEGER M,N,I,J
      PARAMETER (N = 512, M = 640, NTIMES = 500)
      DOUBLE PRECISION Q(N,M), R(M,N)
C
```

*Version 1.3: Aug 25, 2010 10:50 am -0500
†http://creativecommons.org/licenses/by/3.0/

```
      DO I=1,M
        DO J=1,N
          Q(J,I) = 1.0D0
          R(I,J) = 1.0D0
        ENDDO
      ENDDO
C
      DO I=1,NTIMES
        CALL BAZFAZ (Q,R,N,M)
      ENDDO
      END

      SUBROUTINE BAZFAZ (Q,R,N,M)
      IMPLICIT NONE
      INTEGER M,N,I,J
      DOUBLE PRECISION Q(N,M), R(N,M)
C
      DO I=1,N
        DO J=1,M
          R(I,J) = Q(I,J) * R(J,I)
        ENDDO
      ENDDO
C
      END
```

Exercise 4

Code the matrix multiplication algorithm in the "straightforward" manner and compile it with various optimization levels. See if the compiler performs any type of loop interchange.

Try the same experiment with the following code:

```
DO I=1,N
  DO J=1,N
    A(I,J) = A(I,J) + 1.3
  ENDDO
ENDDO
```

Do you see a difference in the compiler's ability to optimize these two loops? If you see a difference, explain it.

Exercise 5

Code the matrix multiplication algorithm both the ways shown in this chapter. Execute the program for a range of values for N. Graph the execution time divided by N3 for values of N ranging from 50×50 to 500×500. Explain the performance you see.