263-2300-00: How To Write Fast Numerical Code
Assignment 3: 100 points
Due Date: Thu March 21 17:00
http://www.inf.ethz.ch/personal/markusp/teaching/263-2300-ETH-spring13/course.html
Questions: fastcode@lists.inf.ethz.ch

Submission instructions (read carefully):

- (Submission)
  We set up a SVN Directory for everybody in the course. The Url of your SVN Directory is https://svn.inf.ethz.ch/svn/pueschel/students/trunk/s13-fastcode/YOUR.NETZH.LOGIN/ You should see subdirectory for each homework.
- (Late policy)
  You have 3 late days, but can use at most 2 on one homework. Late submissions have to be emailed to fastcode@lists.inf.ethz.ch.
- (Formats)
  If you use programs (such as MS-Word or Latex) to create your assignment, convert them to PDF and submit to svn in the top level of the respective homework directory. Call it homework.pdf.
- (Plots)
  For plots/benchmarks, be concise, but provide necessary information (e.g., compiler and flags) and always briefly discuss the plot and draw conclusions. Follow (at least to a reasonable extent) the small guide to making plots (lecture 5).
- (Neatness)
  5% of the points in a homework are given for neatness.

Exercises:

1. *Cache mechanics (30 pts)* We consider a direct-mapped cache with parameters $(S, E, B) = (4, 1, 16)$ and the following code:

   ```
   double x[5], y[5];
   double sum = 0; int i;

   for (i = 0; i < 10; i++)
     sum += x[2*i%5]*y[2*i%5];
   ```

   We assume that `x[0]` goes into the first slot of the first block, `y[0]` goes into the first slot of the third block, and that `sum` and `i` are held in registers (meaning you do not need to consider them in the cache analysis).

   (a) Determine the miss/hit sequences for `x` and `y` (something like `x`: MHHHMHHM..).

   (b) Determine the miss rate for both the arrays.

   (c) What is the operational intensity of the entire code?

   It helps to draw the cache. Provide enough detail so we see how you did it.

2. *Cache mechanics (30 pts)* Consider the following piece of code:

   ```
   double x[128], sum;
   int i,j;

   for (int i = 0; i < 64; i++) {
   j = i + 64;
   sum += x[i] * x[j];
   }
   ```

Assume the following:

- Array `x` begins at memory address 0.
- The cache is cold.
- The only memory accesses are to the entries of `x`.

Case 1

(a) Assume your cache is a 512-byte direct-mapped data cache with 16-byte cache blocks. What is the cache miss rate?

(b) If the cache were twice as big, what would be the miss rate?

Case 2

(a) Assume your cache is a 512-byte 2-way set associative using an LRU replacement policy with 16-byte cache blocks. What is the cache miss rate?

(b) Will larger cache size help to reduce the miss rate?

(c) Will larger cache line help to reduce the miss rate?

3. *MMM analysis (20 pts)* Consider the following code for a triple loop MMM ($C = AB + C$):

```
// A, B, C, are n x n matrices (data type double)
for (i = 0; i < n; i++)
  for (j = 0; j < n; j++)
    for (k = 0; k < n; k++)
      C[i][j] = C[i][j] + A[i][k]*B[k][j];
```

We assume a cold data cache with block of size B bytes. Ignoring conflict misses:

(a) For fixed n, which is the smallest cache size (in bytes) such that the triple loop only incurs compulsory misses? Explain.

(b) For a last-level cache of size 8 MB with $B = 64$ bytes, what is the largest n such that you have only compulsory misses?

(c) What is the operational intensity using n from 3b?

4. *Proofreading (15 pts)* Find a major mistake in the solution of Q4 from last year's exercise and fix it.