ETH login ID: __________________________
(Please print in capital letters)

Full name: ______________________________

263-2300: How to Write Fast Numerical Code
ETH Computer Science, Spring 2017
Midterm Exam
Wednesday, April 26, 2017

Instructions

• Make sure that your exam is not missing any sheets, then write your full name and
login ID on the front.

• No extra sheets are allowed.

• The exam has a maximum score of 100 points.

• No books, notes, calculators, laptops, cell phones, or other electronic devices are
allowed.

Problem 1 (12 = 2 + 5 + 5) __________________________
Problem 2 (13 = 3 + 10) __________________________
Problem 3 (20 = 4 + 8 + 4 + 4) __________________________
Problem 4 (18 = 6 + 6 + 6) __________________________
Problem 5 (20 = 4 + 3 + 3 + 4 + 3 + 3) __________________________
Problem 6 (17 = 2 + 2 + 3 + 3 + 2 + 3 + 2) __________________________

Total (100) __________________________
Problem 1: Operational Intensity \((12 = 2 + 5 + 5)\)

Consider a function that multiplies \(n \times n\) matrices of doubles \((\text{sizeof(double)} = 8)\) in the form \(C = AB + C\) implemented as straightforward triple loop. The function is run on a computer with a last level cache (write-back/write-allocate) of size 256 KB. Assume an initially empty cache for the below questions.

1. What is the flop count \(W(n)\)?

Solution: \(W(n) = 2n^3\)

2. Determine a lower bound for the data movement \(Q(n)\) (in bytes) incurred by the function considering compulsory reads and writes. Then use the result to obtain an upper bound for \(I(n) = W(n)/Q(n)\). Show enough detail so we can see your reasoning.

Solution: Concerning only compulsory data movements we observe that at least the 3 matrices have to be loaded into memory, and the output should be stored in the \(C\) matrix.

\[
Q(n) \geq (3n^2_{\text{reads}} + n^2_{\text{writes}}) \cdot 8 = 4n^2 \cdot 8
\]

Consequently:

\[
I(n) \leq \frac{2 \cdot n^3}{4n^2 \cdot 8} = \frac{n}{16}
\]

3. For which sizes \(n\) would you expect the previous upper bound to be (roughly) the accurate value for \(I(n)\)? Show your derivation and reasoning.

Solution: Assuming any replacement policy and any level of associatively, the safest option is having the 3 matrices fit inside the cache:

\[
3n^2 \cdot 8 \leq 256 \cdot 2^{10} \implies n \leq \left\lfloor \sqrt[218]{\frac{24}{218}} \right\rfloor = \left\lfloor \sqrt{2/3} \cdot 128 \right\rfloor = 104
\]

The last step (104) was not required.
Problem 2: Flop Count \((13 = 3 + 10)\)

Consider the following code for computing the determinant of an invertible matrix using Bareiss algorithm. Assume that \(N > 1\).

```c
void bareiss(float **A, int N) {
    int i, j, k;
    for (i = 0; i < N-1; i += 1) {
        for (j = i + 1; j < N; j += 1) {
            for (k = i + 1; k < N; k += 1) {
                if (i == 2) A[j][k] /= A[i-1][i-1];
            }
        }
    }
}
```

1. Define a detailed floating point cost measure \(C(N)\) for the function `bareiss`. Ignore integer operations.

**Solution:** \(C(N) = (add(N), \text{mul}(N), \text{div}(N))\)

2. Compute the cost \(C(N)\) as just defined. Show your derivation.

**Solution:**

\[
\text{add}(N) = \sum_{i=0}^{N-1} i^2 = \frac{N^3}{3} + \mathcal{O}(N^2)
\]

\[
\text{mul}(N) = \sum_{i=0}^{N-1} i^2 \cdot 2 = \frac{2N^3}{3} + \mathcal{O}(N^2)
\]

\[
\text{div}(N) = (N - 3)^2 = N^2 + \mathcal{O}(N)
\]

**Note:** Lower-order terms (and only those) may be expressed using big-O notation. This means: as the final result something like \(3n + \mathcal{O}(\log(n))\) would be ok but \(O(n)\) is not.

The following formulas may be helpful:

- \(\sum_{i=0}^{n-1} i = \frac{n(n-1)}{2} = \frac{n^2}{2} + \mathcal{O}(n)\)
- \(\sum_{i=0}^{n-1} i^2 = \frac{(n-1)n(2n-1)}{6} = \frac{n^3}{3} + \mathcal{O}(n^2)\)
Problem 3: Bounds \((20 = 4 + 8 + 4 + 4)\)

Consider the AVX code below that performs point-wise multiplication of two arrays of interleaved complex numbers:

```c
#include <immintrin.h>

// Assume 'neg' is a global variable
__m256d neg;

// Assume that 'init_f' will be called once before 'f'

void init_f () {
    const double global_neg [] = {1.0, -1.0, 1.0, -1.0};
    neg = _mm256_loadu_pd(global_neg);
}

// Assume that N is divisible by 2

void f (const double * lhs, const double * rhs, double * res, size_t N) {
    __m256d va, vb, v1, v2, v3, v4, v5;
    size_t i;

    for (i = 0; i < 2 * N; i += 4) {
        va = _mm256_loadu_pd (lhs + i);
        vb = _mm256_loadu_pd (rhs + i);

        v1 = _mm256_mul_pd (va, vb);
        v2 = _mm256_permute_pd (vb, 0x5);
        v3 = _mm256_mul_pd (v2, neg);
        v4 = _mm256_mul_pd (va, v3);
        v5 = _mm256_hsub_pd (v1, v4);

        _mm256_storeu_pd (res + i, v5);
    }
}
```

Assume that the code is executed on Haswell computer, such that the whole working set of function \(f\) fits L1 cache and is already loaded into L1 (warm cache scenario). Also assume that \(N\) is always divisible by 2, and assume that the \(init_f\) function has been executed once in the main function to initialize the global variable \(neg\).

Figure[1] shows the port structure of the Haswell microarchitecture, and information on SIMD intrinsics used in the code above. Integer operations can be ignored in this question. Show enough detail with each answer so we understand your reasoning.
1. Determine the (mathematical) cost of \( f \) measured in flops.

**Solution:** The algorithm consists of multiplication instructions on lines 29, 31 and 32 (\_mm256\_mul\_pd) and an instruction for horizontal subtraction of adjacent pairs of double precision numbers (\_mm256\_hsub\_pd) on line 33. The loop runs for \( \frac{N \cdot 2^4}{4} \) iterations and each instruction performs 4 flops at a time. Therefore:

\[
W(N) = \frac{N \cdot 2^4}{4} \cdot 4 \cdot (3\_mm256\_mul\_pd + 1\_mm256\_hsub\_pd) = 8N \text{ flops}
\]

2. Determine a lower bound (as tight as possible) for the runtime and an associated upper bound for the performance of \( f \) based on the instruction mix, ignoring dependencies between instructions (i.e., don’t consider latencies and assume full throughput).

**Solution:** Assuming no dependencies, the two load, the store, and the permute can be executed in parallel on ports 2, 3, 4 and 5, and each will be executed in one cycle. The mul and hsub instruction will also be executed in parallel on ports 0 and 1. To achieve the tightest bound possible on runtime we assume that the out of order execution engine of the CPU will schedule two mul instructions on Port 0 and the hsub and the remaining mul on Port 1. The latter is best done interleaved hsub - mul - hsub - mul - etc. which obeys the gap of the hsub. In summary, every cycle 3 muls and 1 hsub will be issued. Thus:

\[
\frac{8N \text{ flops}}{\frac{N}{2} \cdot 2 \text{ cycles}} = \frac{8N}{N} = 8 \text{ flops/cycle}
\]
3. Take now into consideration the dependencies: draw a DAG for one loop iteration (i.e., for the lines 26–35). The nodes are the instructions.

Solution:

4. Based on 3. above estimate the runtime (latency) of one loop iteration using latency information.

Solution:
The DAG above shows the following instructions on the critical path:

1 cycle - vb - _mm256_loadu_pd
1 cycle - v2 - _mm256_permute_pd
5 cycles - v3 - _mm256_mul_pd
5 cycles - v4 - _mm256_mul_pd
5 cycles - v5 - _mm256_mul_pd
1 cycle - res - _mm256_storeu_pd

Therefore 18 cycles in total.
Problem 4: Cache Mechanics \((18 = 6 + 6 + 6)\)

We consider a 128 byte data cache that is fully associative and can hold 4 doubles in every cache line. The cache uses least recently used replacement policy. A double is assumed to require 8 bytes.

For the C code below we assume a cold cache. The code accesses elements of array \(A\) in the order determined by function \(f\). Assume that \(A\) is randomly initialized, cache aligned (that is, \(A[0]\) is loaded into the first slot of a cache line) and that all scalar variables are held in registers.

Note: It helps to draw the cache.

```c
int i, j;
double A[55];
for (i = 0; i < 55; i++)
    A[f(i)] = A[f(i)] + i;
```

Consider the reads and the writes and answer the following:

1. For \(f(i) = i\):
   (a) Determine the miss rate.

   **Solution:**
   \[
   \text{Cache miss rate} = \frac{14 \times 100}{110} = \frac{140}{11} \approx 12.7\% 
   \]

   (b) What kind of misses occur?

   **Solution:** Compulsory

   (c) What kind of locality does the code have with respect to accesses of \(A\) and this cache.

   **Solution:** Spatial and Temporal

2. For \(f(i) = (2 \times i) \% 55\):
   (a) Determine the miss rate.

   **Solution:**
   \[
   \text{Cache miss rate} = \frac{28 \times 100}{110} = \frac{280}{11} \approx 25.4\% 
   \]

   (b) What kind of misses occur?

   **Solution:** Compulsory and Capacity
(c) What kind of locality does the code have with respect to accesses of A and this cache.

**Solution:** Spatial and Temporal

3. For \( f(i) = (21 \cdot i) \mod 55 \):

(a) Determine the miss rate.

**Solution:**

\[
\text{Cache miss rate} = \frac{55 \times 100}{110} = 50\%
\]

(b) What kind of misses occur?

**Solution:** Compulsory and Capacity

(c) What kind of locality does the code have with respect to accesses of A and this cache?

**Solution:** Temporal
Problem 5: Roofline \((20 = 4 + 3 + 3 + 4 + 3 + 3)\)

Given is a computer that supports single precision floating point operations \((\text{sizeof(float)} = 4)\). It does not support any SIMD operations. The relevant port information is as follows:

Port 0: fma, mul
Port 1: fma, mul, add

Each of these operations has a throughput of 1 per port. The memory bandwidth is 4 floats per cycle. The cache block size is 4 floats and the cache is write-back/write-allocate. We assume a cold cache scenario.

1. Draw the roofline plot for this computer into the above graph. Give a very brief explanation.

The peak \(\pi\) is 2 fmas/cycle = 4 flops/cycle. The bandwidth \(\beta\) is 16 bytes/cycle.
Now we consider the following function, where \( k \) is a global variable. \( x \) and \( y \) are not aliased.

```c
// assume \( k \) is defined globally
void f(float * x, float * y, int n) {
    int i; float a, b;
    for (i = 0; i < k * n; i += k) {
        a = x[i];
        b = y[i];
        y[i] = a * (2 * a + b);
    }
}
```

2. Based (only) on the instruction mix (i.e. ignoring dependencies), which performance is maximally achievable for this function and why? Draw an associated tighter horizontal roof into the plot above.

**Solution:**

\[ \pi_{\text{max}} = 3 \text{ flops/cycle} \]

There are \( n \) fmas and \( n \) mults, i.e., \( 3n \) flops. The fastest way to execute these is in parallel in \( n \) cycles, which gives a peak of 3 flops/cycle.

3. At what operational intensity \( I \) does this new horizontal roof intersect with the memory roof?

**Solution:** One has to solve \( \beta I = 3 \), which yields \( I = \frac{3}{16} \) flops/byte.
4. Assume $k = 1$. Determine the operational intensity (reads and writes) $I$ of $f$ and mark it in the plot. Based on this $I$, which peak performance is achievable?

**Solution:** The arrays $x$ and $y$ have to be read and $y$ has to be written. Thus:

$$Q(n) = 3n \cdot 4 = 12n \text{ bytes}$$
$$W(n) = 3n \text{ flops}$$
$$I(n) = \frac{3n}{12n} = \frac{1}{4} \text{ flops/byte}$$

Since this is larger than $3/16$, the peak of $3$ flops/cycle is achievable.

5. Assume now $k = 2$. Determine the operational intensity (reads and writes) $I$ of $f$ and mark it in the plot. Based on this $I$, which peak performance is achievable?

**Solution:** Since the data is now accessed with a step size of 2 and the cache block size is 4 floats, twice the data is loaded, i.e.,

$$Q(n) = 3n \cdot 4 \cdot 2 = 24n \text{ bytes}$$
$$W(n) = 3n \text{ flops}$$
$$I(n) = \frac{3n}{24n} = \frac{1}{8} \text{ flops/byte}$$

Since this is smaller than $3/16$, the memory bound is tighter: performance

$$P \leq \beta \cdot I = 16 \cdot 1/8 = 2 \text{ flops/cycle}$$

6. Give a general formula for $I$, dependent on $k$.

**Solution:** $W(n) = 3n$, independent of $k$. $Q(n)$ increases with $k$ until $k = 4$, when for each float an entire block of 4 is loaded. Thus:

$$I(n) = \frac{3n}{12nk} = \frac{1}{4k} \text{ flops/byte if } k \leq 4$$
$$I(n) = \frac{3n}{48n} = \frac{1}{16} \text{ flops/byte if } k > 4.$$
Problem 6: Sampler \(17 = 2 + 2 + 3 + 3 + 2 + 3 + 2\)

1. Provide \(\text{row}_\text{start}(M)\) of the below matrix when expressed in Compressed Sparse Row (CSR) format.

\[
M = \begin{pmatrix}
0 & 1 & 3 & 7 \\
0 & 0 & 5 & 0 \\
0 & 2 & 6 & 0
\end{pmatrix}
\]

**Solution:** \(\text{row}_\text{start}(M) = \{0, 3, 4, 6\}\)

2. Name two processor mechanisms that can dynamically increase the ILP of an executable.

**Solution:** Out-of order-execution, register renaming, branch prediction.

3. A function computing \(y = 2 \times x + y\), where \(x, y\) are vectors of doubles of length \(n\) runs on a processor that can execute per cycle one floating point add and one floating point mult. Running with cold cache once, the function achieves 25% of peak. Estimate the memory read bandwidth.

**Solution:** The function needs to read 2 doubles = 16 bytes to run at peak. Since it runs at 25% of the peak, the memory read bandwidth should be 4 bytes/cycle.

4. Assume two functions \(f_1, f_2\) that both implement matrix multiplication. \(f_1\) is implemented (and run) with optimal square blocking on a computer with a last level cache of size \(\gamma_1\). \(f_2\) is implemented (and run) with optimal square blocking on a computer with a last level cache of size \(\gamma_2\). Estimate the ratio of the operational intensities \(I_1, I_2\) of \(f_1, f_2\) for large \(n\).

**Solution:** \(I_1 = \Theta(\sqrt{\gamma_1})\), \(I_2 = \Theta(\sqrt{\gamma_2})\). Thus \(I_2/I_1\) can be estimated as \(\sqrt{\gamma_2}/\sqrt{\gamma_1}\) or \(\Theta(\sqrt{\gamma_2}/\sqrt{\gamma_1})\)
5. Why does a numerical function typically need more ILP (independent operations) than available execution units to achieve high performance on modern processors (e.g., on Intel Haswell)?

**Solution:** To compensate for the deep pipelines (long latencies) of the execution units.

6. Given is a computer with one cache of size 512 bytes and a cache block size of 16 bytes. How many entries should a TLB have at least so that cache hits imply TLB hits? (Ignore conflicts due to associativity.)

**Solution:** The cache has 32 cache blocks, and each can map to a different page in the TLB. To make sure that a cache hit also implies TLB hit, we need at least 32 entries.

7. Name two types of dependencies that are avoided if C code is written in SSA style?

**Solution:** WAW (write-after-write) and WAR (write-after-read) dependencies.