ETH zürich

P. Koumoutsakos, W. Petersen, P. Hadjidoukas
ETH Zentrum, CLT E 13
CH-8092 Zürich

High Performance Computing for Science and Engineering I

Fall semester 2017

Set 4 - Roofline, OpenMP

Issued: October 20, 2017
Hand in (optional): October 27, 2017 8:00am

Question 1: Roofline Model

Given the following serial code snippet:

```
1  float A[N], B[N], C[N];
2  ...
3  const int P=2;
4  for(int i=0; i<N; ++i) {
5    unsigned j = 0;
6    while(j < P) {
7      A[i] = B[i]*A[i]+0.5;
8      ++j;
9    }
10   C[i] = 0.9*A[i]+C[i];
11 }
```

a) What is the operational intensity of the code? Assume an infinite cache and state any further assumption you made. Show your calculations.

b) A compute node has a peak performance of 409.7 GFLOP/s (single precision) and a memory bandwidth of 34 GB/s. For which range of positive integer values `P` is the code of subquestion (a) memory bound? Show your calculations.

Question 2: OpenMP bug hunting

Identify and explain any bugs in the following OpenMP code. Propose a solution. Assume all headers are included correctly.

```
1  #define N 1000
2
3  extern struct data member[N];  // array of structures, defined elsewhere
4  extern int is_good(int i); // returns 1 if member[i] is "good", 0 otherwise
5
6  int good_members[N];
7  int pos = 0;
8
9  void find_good_members()
10 {
11     #pragma omp parallel for
12     for (int i=0; i<N; i++) {
13         if (is_good(i)) {
14             good_members[pos] = i;
15
16             #pragma omp atomic
17             pos++;
18         }
19     }
20 }
```

Hints:

- Identify the race condition, as we did in Quiz 2 of the last lecture (OpenMP part 2).
- In your solution you can use "omp critical" or "omp atomic capture"[1]

Question 3: OpenMP loop scheduling

Implement an equivalent version of the following code without using OpenMP worksharing constructs (for, sections, single).

```
extern double work(int i);
void test(double *a, int N)
{
    #pragma omp parallel for schedule(dynamic,1)
    for (int i = 0; i < N; i++) {
        A[i] = work(i);
    }
}
```

Hints:

- According to the specified loop scheduling policy, an idle thread gets dynamically the next available iteration (chunk size = 1).
- Study the example code for the schedule clause, available at `https://gitlab.ethz.ch/hpcse17/hs2017/blob/master/examples/openmp1/basic/for.c`.
- You will need a parallel region and some synchronization mechanism for updating the shared loop counter.

Question 4: Statistics

In `statistics.cpp`, the sequential diagnostics function `compute_max_density()` finds and prints the maximum density value and its location.

```
void compute_max_density(double *rho_, int N)
{
    // rho_ : matrix of size NxN, allocated as one dimensioal array.
    // rho[i*N+j] corresponds to rho[i, j]
    // This routine finds the value of max density (max_rho) and its
    // location (max_i, max_j) - it assumes there are no duplicate values
    double max_rho;
    int max_i,  max_j;

    max_rho = rho_[0];
    max_i = 0;
    max_j = 0;

    for (int i = 0; i < N; ++i)
        for (int j = 0; j < N; ++j)
        {
            if (rho_[i*N + j] > max_rho)
            {
                max_rho = rho_[i*N + j];
                max_i = i;
                max_j = j;
            }
        }

    printf("=============================================\n");
```

[1] omp atomic capture: OpenMP specs 3.1, section 2.8.5, especially page 74, lines 8–13.

```
26      printf("Output of compute_max_density():\n");
27      printf("Max rho: %.16f\n", max_rho);
28      printf("Matrix location: %d %d\n", max_i, max_j);
29  }
```

Provide, in the function `compute_max_density_omp()`, a parallel OpenMP implementation of the previous code .

Hints:

- Try to keep the number of memory accesses close to that of the sequential version.
- Study the hands-on example `find_max` of the last lecture (OpenMP part 2).