Multicore Computing

Weijie Zhao

08/29/2024

OpenMP

- Open Multi-Processing
- An API that supports multi-platform shared-memory multiprocessing programming in C, C++, and Fortran

OpenMP: Quick Start

```
for (int i = 0; i < N;++i){
        b[i] = a[i] + 1;
}
```

OpenMP: Quick Start

```
#pragma omp parallel for schedule(static) num_threads(8)
for (int i = 0; i < N;++i){
        b[i] = a[i] + 1;
}
```

- g++ test.cc –fopenmp -o test -O2

- brew install libomp
- clang++ test.cc -o test -O2 -Xpreprocessor -fopenmp -I/usr/local/include -L/usr/local/lib -lomp

OpenMP: Quick Start

```
int sum = 0;


for (int i = 0; i < N; ++i){
        sum += a[i];
}
```

OpenMP: Quick Start

```
int sum = 0;
#pragma omp parallel for schedule(static) default(shared) reduction(+:sum) num_threads(8)
for (int i = 0; i < N; ++i){
        sum += a[i];
}
```

OpenMP: Slow Start

- #include<omp.h>
- void omp_set_num_threads(int num_threads)
- int omp_get_num_threads()
- int omp_get_thread_num()

- #pragma omp atomic (update/read/write/capture)
- #pragma omp critical

HW 1: Logistic Regression

- Given matrix X and label Y, perform gradient descent of logistic regression
- 10 independent test cases. Each case weights 1 pt.

- The compilation is considered failed if it does not finish in 1 minute.
- A test case is considered incorrect if it does not finish in 2 minutes.
- The training accuracy must reach 60%.

- The summation of the execution time across 10 cases will be uses to rank correct solutions.

- Due: 09/13/2024 5:00 pm EDT

Grading

- Homework 40%
- Reading 10%
- Project 50%

- 90%<=A<=100%
- 80%<=B<90%
- 70%<=C<80%
- 60%<=D<70%
- 0%<=F<60%

- 5 pieces of homework.
- No late submissions.
- No 3rd party code
- Automatically tested: Please strictly follow the output format. An incorrect format is considered as a wrong answer.
- The best 4 scores among the 5 are counted in your final grade.
- The fastest correct solution in each homework gets 10% bonus score in the final grade.
- Other correct solutions that are no slower than 2X of the fastest one gets 5% bonus score in the final grade.

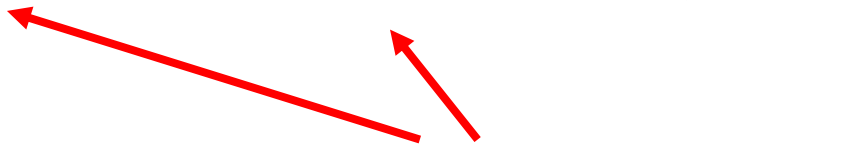
Input Data

- First line contains 8 integers: N D x0 x1 A B C M
- For i >= 2
  - X[i] = (A * X[i - 1] + B * X[i - 2]+ C) % M
- For all i
  - X[i] /= M;

- N <= 10^5
- D <= 1600

Input Data

- First line contains 8 integers: N D x0 x1 A B C M
- For i >= 2
  - X[i] = (A * X[i - 1] + B * X[i - 2]+ C) % M
- For all i
  - X[i] /= M;



Caution the potential overflow here!

- N <= 10^5
- D <= 1600

Output Format

- D lines
- Each line contains a floating number
  - The logistic regression parameters

What Do We Need to Do?

- We are required to complete two scripts
- compiler.sh
  - it is executed once before the actual testing starts
- run.sh
  - it should takes two arguments, the first argument is the input file name, the second one is the file name that you should write your sorted results into.

Testing Environment

- ssh yourusername@granger.cs.rit.edu
- Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz
- 48 threads in total (2 sockets, 12 cores per socket, 2 threads per core)
- 251 GB memory

- Testing limit:
  - 8 threads taskset -c

“Premature optimization is the root of all evil”

--- Sir Tony Hoare