Kyle Salitrik

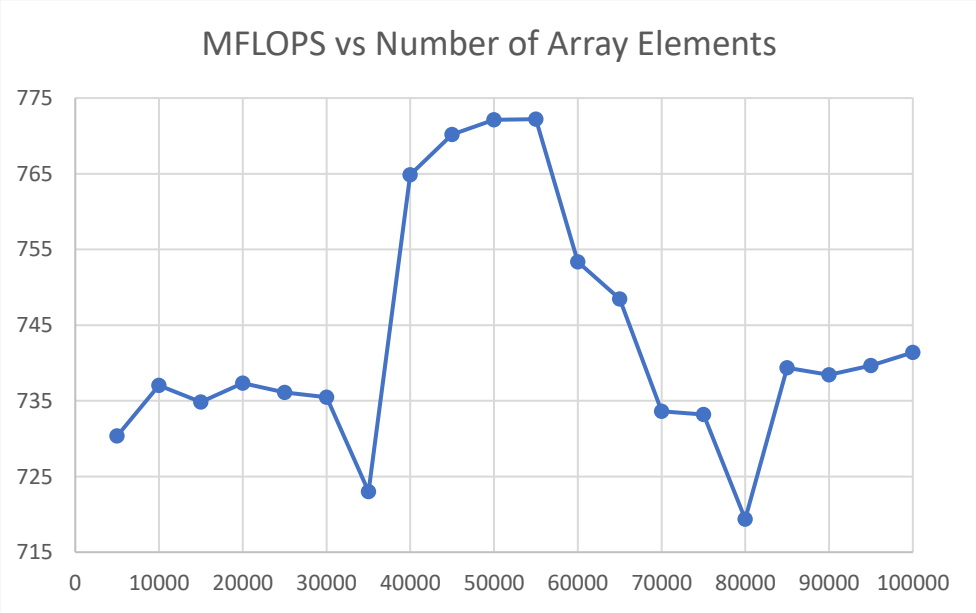
Assignment 1: Memory Read Benchmark

Within my program I used a for loop to generate numbers for N starting with 5,000 and adding 5,000 every iteration. From here I increased R from 1,000 to 100,000 by factors of 10 until the benchmark began to take a somewhat significant time to run. Long integers were used as rollover occurred because of the number of iterations it took to get the program running long enough.

Below is a table that shows the values of R, N, and MFLOPS for the combinations of values. It seems peculiar that the number of MFLOPS decreases significantly and rises again at 60,000 array items and 85,000, respectively. Using the '-O2' flag with GCC increased MFLOPS significantly, from around 600 for R = 100,000, N = 5000 to around 730 MFLOPS as shown in the table. The equipment and compilation command used are as follows:

- CPU: Intel Core i7-6700HQ, 2.6 GHz
- GPU: GeForce GTX 1060
- Memory: 16GB
- Operating System: Ubuntu 16.04 LTS 64-bit
- Compiler: GCC
- Compile Command: gcc -Wall kps168_assignment_1.c -o benchmark -O2

R	N	MFLOPS
100000	5000	730.33862
100000	10000	737.02807
100000	15000	734.83269
100000	20000	737.3446
100000	25000	736.10003
100000	30000	735.49143
100000	35000	723.00889
100000	40000	764.86537
100000	45000	770.16927
100000	50000	772.14574
100000	55000	772.20604
100000	60000	753.37285
100000	65000	748.44516
100000	70000	733.59445
100000	75000	733.17481
100000	80000	719.3676
100000	85000	739.35388
100000	90000	738.43546
100000	95000	739.65418
100000	100000	741.39544



Appendix 1: Code

```
#include <stdio.h>
#include <sys/time.h>

/*
MEMORY READ BENCHMARK
System Info:
    CPU: Intel Core i7-6700HQ, 2.6 GHz
    GPU: GeForce GTX 1060
    Memory: 16GB
    Operating System: Ubuntu 16.04 LTS 64-bit
    Compiler: GCC
compiled using: gcc -Wall kps168_assignment_1.c -o benchmark -O2; ./benchmark
*/

void get_walltime(double *wcTime){
    struct timeval tp;
    gettimeofday(&tp, NULL);
    *wcTime = (double) (tp.tv_sec + tp.tv_usec / 1000000.0);
}

void dummy(double A, double B, double C, double D){}

double benchmark(long int R, long int N){
    double A[N], B[N], C[N], D[N], S, E, MFLOPS;
    for (int i = 0; i < N; i++){
        A[i] = 0.0;
        B[i] = 1.0;
        C[i] = 2.0;
        D[i] = 3.0;
    }
    get_walltime(&S);
    for (int j = 1; j < R; j++){
        for (int i = 1; i < N; i++){
            A[i] = B[i] + C[i] * D[i];
        }
        if (A[2] < 0) {
            dummy(0.0, 0.0, 0.0, 0.0);
        }
    }
    get_walltime(&E);
    MFLOPS = (R * N * 2.0) / ((E - S) * 1000000.0);
    return (MFLOPS);
}

int main(){
    double mflops;
    long int m, n, r;
    m = 20;
    r = 100000;
    printf("R,N,MFLOPS\n");
    for (int i = 0; i < m; i++){
        n = 5000 + 5000 * i;
        mflops = benchmark(r, n);
        printf("%ld,%ld,%f\n", r, n, mflops);
    }
    return 0;
}
```