COL-216 Assignment-1

Mohit Thakur 2019CS10373
Sidharth Agarwal 2019CS50661

February 22, 2021

Aim

Write a MIPS Assembly Program for obtaining the area under a curve formed by joining successive points by a straight line.

Approach

1. First we calculated area under the curve in case of straight line joining $(x_1, y_1)\&(x_2, y_2)$. For calculating the area, we have subdivided the problem in two cases
   1) Both $y_1\&y_2$ are of same sign $\implies y_1 \cdot y_2 \geq 0$. Then, $Area = 0.5 \cdot (x_2 - x_1) \cdot |(y_2 + y_1)|$
   2) $y_1\&y_2$ are of opposite sign $\implies y_1 \cdot y_2 < 0$. Then, $Area = 0.5 \cdot (x_2 - x_1)|(y_2^2 + y_1^2)/(y_2 - y_1)|$
2. Then we reduced our problem to calculate such successive areas and add them to a common variable area.
3. For that, we have first stored the current x_i and y_i in registers, taken input x_{i+1} and y_{i+1} in different registers and then caluclated the area under the curve formed by this straight line, added it the net area till this moment and after which we have moved x_{i+1} value to register having x_i value and y_{i+1} value to register having x_i value. and repeated the process.

Design Decisions

1. First, we ignored the values of n that are less than 1 and simply returned 0 as area for those cases.
2. If somewhere in between we have a sequence of inputs like ....(p, q), (x, y_1), (x, y_2)....(x, y_m), (t, s).... (equal x coordinates,unequal y coordinates)such that $p < x < t$ then our program computes the area equivalent to that formed by the lines from (p, q)...(x, y_1)first occurence of x and (x, y_m)...(t, s)last occurence of x
3. We used single precision floating type values to store the areas and variables at all times(except counter and n).

Testing

We followed 2 methods for testing:

Smaller inputs in Qtspim interface

For this we wrote smaller cases to cover all the corner situations and generated some small random test cases to check the correctness of the code.
These case include:

1. $N \leq 1 \implies$ Area is 0
2. Area is integer
3. x coordinates are equal for some inputs.
4. large values of x and y

We realized this cannot be used for stress testing as input needs to be typed in manually.

Stress test with almost similar C++ code

Wrote the C++ code with line-to-line similarity to the MISP code.
Wrote another C++ code to generate random test cases and area.
These cases include larger sizes of n, x and y.

Conclusion

After running stress tests and corner cases on the input, we have realized that output is correct for first 23 bits(almost 7 digits in decimal). That bottleneck is due to the usage of single precision floating point values. Result can be made more accurate by using double precision floating points. But we feel the error with single precision is quite acceptable for the purposes.