PROJECT REPORT

Group 1:

Abin Shoby

Adithya G

Jeenu Sebastian

Shaun Thayil

TOPIC: Stencils for physically realistic simulations in graphics

BRIEF: Fast fluid dynamics simulation using Navier-Stokes equations (stencils)) on the GPU

TASK 1:

- Installation of Polly from https://polly.llvm.org/get_started.html.

  mkdir polly && cd polly

  wget http://polly.llvm.org/polly.sh

  Changed cmake line to cmake ${LLVM_SRC} -DLLVM_USE_LINKER=gold -

  DCMAKE_BUILD_TYPE=Release

  chmod +x polly.sh

  ./polly.sh

- Installed Clang by using command, sudo apt install clang

TASK 2:

- Refered the link
  http://developer.download.nvidia.com/books/HTML/gpugems/gpugems_ch38.html

  (Mainly the table 38.1)

  It contains the Navier stokes equation and tips for solving the equation programatically.

TASK 3:

- Observations were made on the execution time of programs with and without the use of polly.
- command used for parallelization of loops: clang -O3 filename.c -mllvm -polly -o outfile

❖ DIFFERENTIAL AT X

> Without using polly

real 0m0.005s

user 0m0.001s

sys 0m0.004s

> Using polly

real 0m0.005s

user 0m0.005s

sys 0m0.000s

Conclusion: No change in execution time by using polly. So it is not well optimized since there is only single loop.

❖ NESTED LOOP

> Without using polly

real 0m0.015s

user 0m0.012s

sys 0m0.004s

> Using polly

real 0m0.004s

user 0m0.003s

sys 0m0.001s

Conclusion: Performance of the program which uses polly is 3.75 times than that of the normal program. It is due to the loop unrolling of nested loop.

- BUBBLESORT

Trial 1: Array Size: 100 elements

> Without using polly

real 0m0.185s

user 0m0.180s

sys 0m0.004s

>Using polly

real 0m0.050s

user 0m0.046s

sys 0m0.004s

Trial 2: Array Size: 10000 elements

> Without using polly

real 0m0.163s

user 0m0.162s

sys 0m0.001s

>Using polly

real 0m0.056s

user 0m0.053s

sys 0m0.004s

Conclusion: Performance of Bubble sort is higher by a factor of 3.3 on average when executed using polly.

❖ QUICKSORT

Trial 1: Array Size - 100 elements (Worst Case)

> Without using polly

real 0m0.002s

user 0m0.002s

sys 0m0.000s

> Using polly

real 0m0.002s

user 0m0.001s

sys 0m0.002s

Trial 2: Array Size - 1000 elements (Worst Case)

> Without using polly

real 0m0.008s

user 0m0.008s

sys 0m0.000s

> Using polly

real 0m0.004s

user 0m0.004s

sys 0m0.000s

Conclusion: When the number of elements in the array is large, we observed that the execution time of the program using polly is half of the one which doesn't use polly.
performance improvement obtained is 2x.

- MATRIX MULTIPLICATION

Order Of Matrix : 1000x1000

> Without using polly

real 0m0.003s

user 0m0.002s

sys 0m0.001s

> Using polly

real 0m0.001s

user 0m0.002s

sys 0m0.000s

Conclusion : Matrix Multiplication using is 3x Faster than one which doesn't use polly.

- FACTORIAL

With number = 20

> Without using polly

real 0m0.005s

user 0m0.000s

sys 0m0.005s

> Using polly

real 0m0.002s

user 0m0.001s

sys 0m0.000s

Conclusion: Performance of Factorial program using polly is 2.5 times the one which doesn't use polly

- PRIME NO GENERATION

With limit, n= 100000

> Without using polly

real 0m5.919s

user 0m5.877s

sys 0m0.040s

> Using polly

real 0m5.137s

user 0m5.089s

sys 0m0.048s

Conclusion: Performance of Prime no generation using polly is slightly better than execution of the program which doesn't use polly.

- FIBONACCI

With number =1000000

> Without using polly

real 0m8.100s

user 0m0.503s

sys 0m1.899s

> Using polly

real 0m7.995s

user 0m0.455s

sys 0m1.928s

Conclusion: Performance of Fibonacci generation using polly is slightly better than execution of the program without using polly.

- FREQUENCY OF NUMBERS

No of elements n= 100000

> Without using polly

real 0m15.422s

user 0m15.413s

sys 0m0.000s

> Using polly

real 0m4.583s

user 0m4.578s

sys 0m0.004s

Conclusion: Performance of the program that finds frequency of each numbers in an array using polly is better than execution of the program without using polly by a factor of 3.36.

- PERMUTATION

Number of elements, n=400

> Without using polly

real 1m8.685s

user 0m7.622s

sys 0m1.672s

>Using polly

real 1m7.653s

user 0m7.350s

sys 0m1.812s

Conclusion: Performance of program that finds permutation of set of numbers using polly is better than execution of the program without using polly.

- CHAIN MATRIX MULTIPLICATION

Trial 1 : Number Of Matrices - 10

> Without using polly

real 0m2.126s

user 0m0.003s

sys 0m0.000s

> Using polly

real 0m2.724s

user 0m0.002s

sys 0m0.001s

Trial 2 : Number Of Matrices - 20

>Without using polly

real 0m8.446s

user 0m6.395s

sys 0m0.000s

>Using polly

real 0m6.329s

user 0m4.189s

sys 0m0.000s

Conclusion : As the number of matrices to be multiplied increases, program which uses polly performs better than the one which is not using polly.

- BINARY SEARCH

Trial 1 : Array Size - 100 (0-99)

Element to be found - 99

>Without using polly

real 0m2.477s

user 0m0.002s

sys 0m0.001s

>Using polly

real 0m2.992s

user 0m0.001s

sys 0m0.000s

Trial 2 : Array Size - 500 (0-499)

Element to be found - 499

>Without using polly

real 0m4.552s

user 0m0.002s

sys 0m0.000s

>Using polly

real 0m7.560s

user 0m0.003s

sys 0m0.000s

Conclusion: The Execution Time of Binary Search increases with increase in array size. No optimization found using polly

- DFS

No of vertices = 1000

>Without using polly

real 0m0.940s

user 0m0.575s

sys 0m0.363s

>Using polly

real 0m0.515s

user 0m0.234s

ssys 0m0.282s

Conclusion: Performance of DFS using polly is better than execution of the program without the use of polly.

- DIJKSTRAS ALGORITHM

no of vertices=10

>Without using polly

real 0m0.004s

user 0m0.001s

sys 0m0.003s

>Using polly

real 0m0.003s

user 0m0.001s

sys 0m0.003s

Conclusion: Performance of Dijkstras algorithm is slightly improved by using polly.

- LINEAR SEARCH

No of elements in the array = 10000

>Without using polly

no of elements =10000

real 0m0.005s

user 0m0.005s

sys 0m0.000s

>Using polly

real 0m0.001s

user 0m0.000s

sys 0m0.001s

Conclusion: Performance of Linear search is improved by a factor of 5 by using polly.

- FINDING ALL SPANNING TREES OF A GRAPH

Complete graph of 1000 vertices was used.

>Without using polly

real 0m1.624s

user 0m1.608s

sys 0m0.008s

>Using polly

real 0m0.844s

user 0m0.832s

sys 0m0.009s

Conclusion: Performance of algorithm to find all spanning trees is improved by a factor of 2

by using polly

- LONGEST INCREASING SUBSEQUENCE

  No of Elements in the array = 20

  >Without using polly

  real 0m0.022s

  user 0m0.021s

  sys 0m0.001s

  >Using polly

  real 0m0.013s

  user 0m0.009s

  sys 0m0.004s

  Conclusion: Performance of longest increasing subsequence program is improved by a factor of 2 by using polly.

Final Conclusion

On Average, performance of Polly with clang is 3x times than that of GCC. Some times the usage of polly improve performance by a factor of 5. But in some cases there is only a slight improvement or no improvement in the performance.