**Problem Statement:** Addition of two large vectors

**Google Colab Platfrom Code:**

**Command to install nvidia gpu on google colab:** !nvidia-smi

import cupy as cp

import numpy as np

import time

# Size of large vectors

N = 10000000

# Create large vectors on CPU

a\_cpu = np.random.rand(N).astype(np.float32)

b\_cpu = np.random.rand(N).astype(np.float32)

# ---------- CPU ----------

start\_cpu = time.time()

c\_cpu = a\_cpu + b\_cpu

end\_cpu = time.time()

print(f"CPU Vector Addition Time: {end\_cpu - start\_cpu:.4f} seconds")

# ---------- GPU ----------

a\_gpu = cp.asarray(a\_cpu)

b\_gpu = cp.asarray(b\_cpu)

start\_gpu = time.time()

c\_gpu = a\_gpu + b\_gpu

cp.cuda.Device(0).synchronize()  # Ensure GPU is done

end\_gpu = time.time()

print(f"GPU Vector Addition Time: {end\_gpu - start\_gpu:.4f} seconds")

**Command to run the shells in google colab:**

Shift enter

**Output:**  
CPU Vector Addition Time: 0.0169 seconds

GPU Vector Addition Time: 0.0039 seconds

**LeetGPU Platform Code:**

**// vector\_add.cu**

#include <stdio.h>

\_\_global\_\_ void vecAdd(float \*A, float \*B, float \*C, int N) {

int i = blockIdx.x \* blockDim.x + threadIdx.x;

if (i < N)

C[i] = A[i] + B[i];

}

int main() {

int N = 5;

size\_t size = N \* sizeof(float);

float A[] = {1, 2, 3, 4, 5};

float B[] = {10, 20, 30, 40, 50};

float C[5];

float \*d\_A, \*d\_B, \*d\_C;

cudaMalloc(&d\_A, size);

cudaMalloc(&d\_B, size);

cudaMalloc(&d\_C, size);

cudaMemcpy(d\_A, A, size, cudaMemcpyHostToDevice);

cudaMemcpy(d\_B, B, size, cudaMemcpyHostToDevice);

vecAdd<<<1, N>>>(d\_A, d\_B, d\_C, N);

cudaMemcpy(C, d\_C, size, cudaMemcpyDeviceToHost);

printf("Result Vector C:\n");

for (int i = 0; i < N; i++)

printf("%f ", C[i]);

printf("\n");

cudaFree(d\_A);

cudaFree(d\_B);

cudaFree(d\_C);

return 0;

}

Output:

Running NVIDIA GTX TITAN X in FUNCTIONAL mode...

Compiling...

Executing...

Result Vector C:

11.000000 22.000000 33.000000 44.000000 55.000000

Exit status: 0