cs327 general purpose programming using gpus

assignment # 01

name: Ahmad Saeed Zaidi
reg no: 2023073

github repository

https://github.com/AhmadSaeedZaidi/gpu_assignment_1

task 1: setup

trivial with colab. go to colab, make a notebook, set runtime, check if its working (call nvidia smi and nvcc, ! can be used to run shell commands).
i verified the setup by running !nvidia-smi to ensure the t4 gpu was attached and !nvcc --version to check the compiler. all details are pushed to assignment01/task01.md in the repo linked above.

task 2: cpu

this part generates the random data using python and saves it to a text file. simple implementation. then cpp file is made using %%writefile and run using shell commands, and we add the matrices in a simple loop.
file structure definition:
to handle the input data, i defined a simple text format:

- line 1: two integers representing rows and columns (e.g., 100 100).
- line 2: flat list of matrix A values, space separated.
- line 3: flat list of matrix B values, space separated.

output:

- line 1: two integers (rows cols).
- content: the result matrix written row-by-row (formatted as a 2d grid).

task 3: gpu

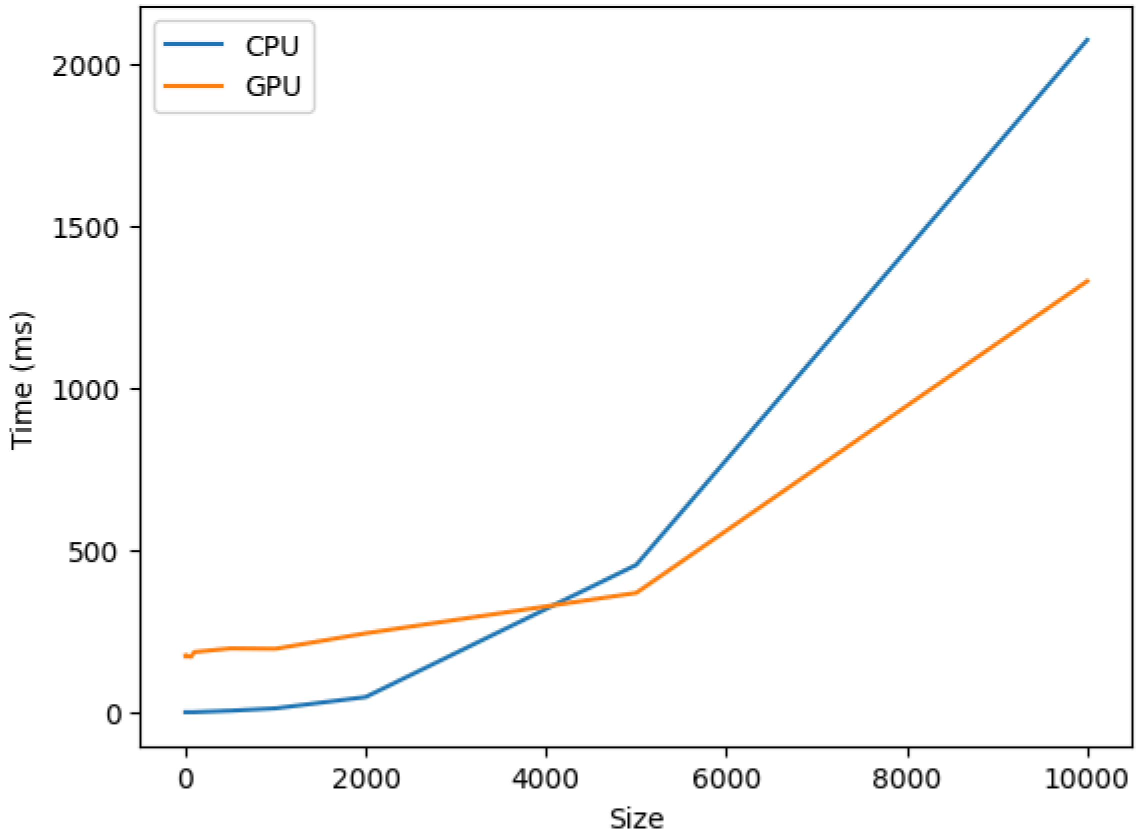
this is the cuda code. instead of a loop, it moves the data onto the graphics card, runs a kernel to add the numbers all at once (in parallel), and then moves the answers back. it uses a makefile to compile everything.
the code handles memory allocation using cudamalloc and uses cudamemcpy to transfer the

parsed matrices from host to device and back.

task 4: comparison

finally, a python script runs both the cpu and gpu programs over and over with different matrix sizes (from 2 up to 5000). it grabs the timing data from each run and saves a plot called task04_plot.png so you can visualize the difference. very cool, i didn't know about the subprocess library before this.

Visualization



findings (as listed in notebook)

gpu only performs better in this case for matrices of size roughly greater than 4000. cpu run time grows exponentially after 2000, while gpu run time seems to grow a noticeably slower. though for smaller sizes (less than 2000), the overhead for gpu compute makes it worse.