CSC 648 Homework 2

Miguel Antonio Logarta

Due: Tuesday Sep 17, 2024

1 Introduction

For this homework assignment, we are comparing the performance of matrix multiplication using two different methods. Our first method will be using plain python for loops, while the second method will be using pytorch's tensors where the matrices are converted into vectorized form. Our results will be plotted on a graph using matplotlib. The main libraries used are numpy, torch, matplotlib.

2 Code

This is our plain Python matrix multiplication algorithm. I implemented the algorithm from wikipedia: Matrix Multiplication.

```
    def multiply_matrices(A: List[List[int]], B: List[List[
        int]]) -> List[List[int]]:
    # If A is an m x n matrix and B is a n x p matrix,
    # Our resulting matrix C should be of size m x p

    # Check if dimensions match (n)
    if len(A[0]) != len(B):
        print("Shapes of matrices A and B do not agree!")
        return

    m = len(A)
    p = len(B[0])
    n = len(B)

    # Initiailize array
    # C = zero_matrix(m, p)
    C = numpy.zeros((m, p))

    for i, crow in enumerate(C):
        for j, ccolumn in enumerate(crow):
            ABSum = 0
            for k in range(n):
                # print(f'k={k}, n={n}, A[{i}][{k}] B[{k}][{
                    j}]')
```

```
                ABSum += A[i][k]*B[k][j]

            C[i][j] = ABSum

    return C
```

For our homework, we were asked to create matrices W and X. W has size 90 x m while X has size m x 110. m takes on ten different values 10, 20, 30, ..., 90, 100, and our task is to time how long it takes to multiply W and X for every value of m.

```
if __name__ == "__main__":
    m_values = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
    plain_time_data = {}
    torch_time_data = {}

    for m in m_values:
        W = numpy.random.rand(90, m)
        X = numpy.random.rand(m, 110)

        W_t = torch.tensor(W)
        X_t = torch.tensor(X)

        # Plain python function with for loops
        plain_time = timeit.Timer(partial(multiply_matrices,
            W, X)).repeat(1, 1)
        plain_time_data[m] = plain_time

        # Pytorch matrices multiplication
        torch_time = timeit.Timer(partial(torch.matmul, W_t,
            X_t)).repeat(1, 1)

        torch_time_data[m] = torch_time

        matplotlib.plot('size of m', 'execution time', data=
            plain_time_data)
        matplotlib.plot('size of m', 'execution time', data=
            torch_time_data)
```