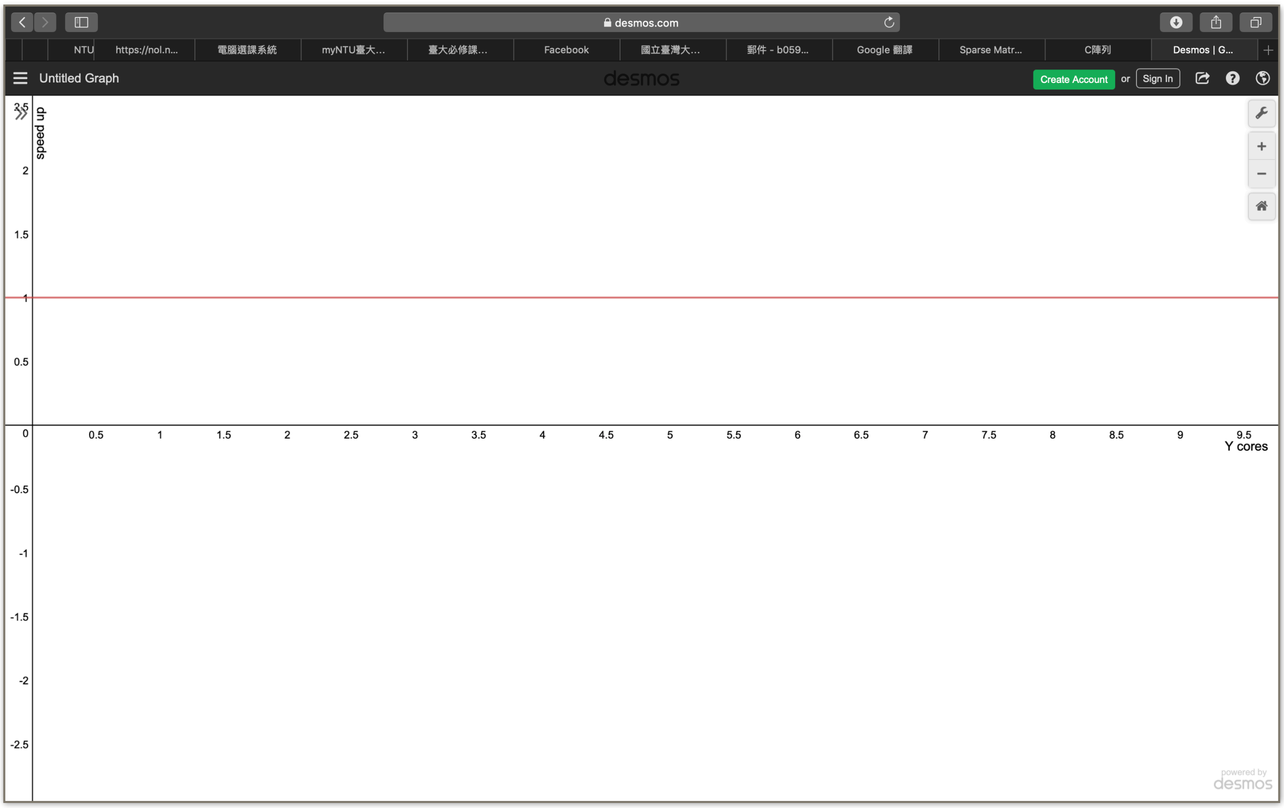
Computer Architecture - Homework #5

資工三 陳盈如 B05902118

6.3

6.3.1 [10] <§6.2>

Seeing that it is difficult to parallelize the code without modifying the code, we will not obtain any speed up by running BinarySearch on a multi-core processor. However, we can compare high and low on the first core, calculate the mid on the second core, and compare A[mid] and X on the third core without reconstructing the code.



6.3.2 [5] <§6.2>

We have already said that we cannot obtain any speed up without changing this code. However, if we create several threads to compare N elements of array A with X at the same time, we can get ideal speed up, that is, Y times speed up. Searching a number in an array can be completed in the amount of time to execute a single comparison.

6.6

6.6.1 [10] <§6.5>

The speed up would be very close to 4.

6.6.2 [10] <§6.5>

Each update would incur the cost of a cache miss, and so will reduce the speedup obtained by a factor of 3 times the cost of servicing a cache miss.

6.6.3 [10] <§6.5>

Calculate the elements in C by traversing the matrix across columns instead of rows.

6.18

6.18.1 [15] <§6.10>

```
void spare(int array[row][column]) {
        int NNZ = 0;
        for (int i = 0; i < row; i++)
                for (int j = 0; j < column; j++)
                        if (array[i][j] != 0)
                                NNZ++;
        int A[NNZ];
        int IA[row + 1];
        int JA[NNZ];
        int count = 0;
        for (int i = 0; i < row; i++) {
                int line = 0;
                for (int j = 0; j < column; j++) {
                        if (array[i][j] != 0) {
                                A[count] = array[i][j];
                                JA[count] = j;
                                count++;
                                line++;
                        }
                }
                IA[i + 1] = line;
        }
}
"a.c" 35L, 648C written
```

6.18.2 [10] <§6.10>

Suppose each floating point is 4 bytes and each index is a short unsigned integer which is 2 bytes. Therefore,

A[NNZ] requires $13 \times 4 = 52\ bytes$

IA[row + 1] requries $(6 + 1) \times 2 = 14\ bytes$

JA[NNZ] requires $13 \times 2 = 26\ bytes$

It needs 92 bytes altogether.