Data Intensive Computing - ELEN 4020
Lab1

William Becerra Gonzalez 789146
Sailen Nair 1078491
Kyle Govender 571133

February 23, 2018

Introduction

This report documents the procedure taken, as well as the reasons behind the solution proposed within the laboratory. The main problem faced during the laboratory was the ability to create an K-dimensional array dynamically, and thereafter the traversal and co-ordinate manipulation of the array.

1 Proposed Solution

The proposed solution to the problem, was to take the K-dimensional array and map it into a 1-D array. This allows for all further tasks and procedures performed on the array to be simple to visualise and code, as only a single for loop is needed traverse a 1-dimensional array. The three procedures will be explored at in greater detail below.

The K-dimensional array within the program is created dynamically, meaning that the actual memory allocation of the array is done at runtime and not at compile time. To do this the function "malloc" needed to be used. "malloc" allows for memory allocation on the heap dynamically. This was needed as the dimensions and size of the K-dimensional array were only known at runtime and not at compile time.

When the program is run, the user is prompted to enter the number of dimensions of the desired K-dimensional array, the user is then prompted to enter the bounds of each dimension of the array. The K-dimensional array is then mapped to a 1-dimensional array which the three procedures act upon. The user is then provided the opportunity to quit the program or to run the program again.

1.1 Mapping K-Dimensional array to 1-Dimensional array

The K-dimensional array was mapped to a 1-dimensional array by multiplying the length of each dimension together. The final product represents the total number of elements of the final mapped 1-dimensional array. Example:

For the 4-Dimensional array: A[20][20][20][20], the dimension sizes are multiplied together:

1-Dimension array length = 20*20*20*20.

The co-ordinates of each element in the K-dimensional array can be retrieved from the mapped 1-dimensional array by use of an algorithm that is discussed in procedure 3 below.

1.2 Procedure 1

Procedure 1 entailed setting each value in the array to 0. This procedure was performed through the use of a single for loop. The loop iterated through the entire array assigning each element the

value 0.

1.3 Procedure 2

Procedure 2 entailed uniformly setting 10% of the elements array to the value 1. This was once again done using a for loop, however the for loop did not increment by 1 each iteration, but rather a calculated amount depending on the array size. The calculation divided the total number of elements in the array by 10% of the number of elements in the array, the result produced the correct increment required for the traversal. The amount incremented allowed for the 10% of the array to be set to 1 uniformly.

1.4 Procedure 3

Procedure 3 entailed the largest amount logic and thought. This procedure entailed selecting 5% of the elements in the 1D array in a uniformly randomly fashion using the rand() function. The value is then displayed alongside its co-ordinates. Obtaining the co-ordinates proved to be the most challenging aspect. The algorithm used for obtaining co-ordinates of a K-dimensional array from a 1-dimensional array consists of one main for loop. The function created takes in three parameters, namely: the dimension of the array, the length of each dimension, as well as the index number of the 1-D array which needs to be converted to a set of co-ordinates.

The for loop will then iterate for the same number of dimensions as the K-dimensional array. Each iteration of the loop performs a modulus operation and then a division of the index number of the 1-D array. The value is then carried over to the next loop iteration where the above two calculations are performed again. After each iteration a co-ordinate value is stored in an array. This algorithm was not created by the authors of this laboratory, it has been derived from the calculations seen in [1].

Once the co-ordinate calculation is complete, it is then passed back to the procedure three function namely "printUniformRandomCoordinates". Thereafter the co-ordinates, value and index number in the 1-D array are all displayed. The results are outputted onto the console.

2 Conclusion

This report documented the steps taken to map a K-dimensional array into a 1-dimensional array. This was done dynamically as the user could input specific sizes and dimensions desired. The three procedures were discussed within the report, with for loops being the sole method of array traversal. The co-ordinates algorithm used was discussed, which details how a co-ordinate set is obtained by only knowing the index number of the 1-D array as well as the dimensions of the K-dimensional array.

References

[1] CPrograming,*https* : *//cboard.cprogramming.com/c* − *programming/*133513 − *traversing* − *n* − *dimensional* − *array.html* Last accessed February 2018.