ELEN4020 Data Intesive Computing for Data Science Lab 1 Report

Leantha Naicker (788753), Fiona Rose Oloo (790305),
Boitumelo Mantji (823869), Justine Wright (869211)

The University of Witwatersrand

February 22, 2018

Abstract

This report contains a short description of the methods used to solve the problems presented in this lab. The lab contained three procedures that were to be carried out on K-Dimensional arrays and the pseudo code to each procedure can be found within the report.

1 Introduction

The objective of this lab was to write a C language based program, which can dynamically generate multi-dimensional integer arrays and perform three procedures on the arrays. Additionally, a method to traverse a K-dimensional array, with K varying, was implemented. The sections shows more detail about the implemented solution.

2 Procedure 1

This procedure moves through the array using a pointer and the size. As it traverses the array, it checks each element to see if it set to zero, if not then it sets it to zero. The routine for this code can be seen below.

Algorithm for Procedure1: Initialize all elements in K-dimension integer array with bounds $N_0, N_1, ...N_{K-1}$ to zero.

```
Procedure1 (ptr_A, size)
    in: a pointer to the beginning of the array and the size of the array counter
    while $counter < size - 1$ do
        call setZero(ptr_A*)
        ptr_A ++
        counter ++
    end while
```

3 Procedure 2

This procedure takes in a pointer to the first element of the array and the array size. The procedure determines what integer value would represent ten percent of the array and changes the first ten percent of the array to 1's. The routine of this code can be seen below.

Algorithm for Procedure2: Set ten percent of the elements in a K-dimensional array to 1's.

```
Procedure2 (ptr_A, size)
    in: a pointer to the beginning of the array and the size of the array
    percent ← 10
    ones ←$size * (percent/100)$
    for loop (0 to ones)
        ptr_A ← 1
        ptr_A ++
    close for loop
```

4 Procedure 3

This procedure randomly chooses five percent of the array, traverses through the chosen five percent elements and prints their respective indices and values. The indices are calculated by using a simple maths function. The routine for procedure 3 is described below.

Algorithm for Procedure3: Randomly selects five percent of the array elements in a random uniform fashion, prints the respective indices of the elements and the value at the chosen cell.

```
Procedure3 (ptr_A, size, dimensions, bounds)
    in: a pointer to the beginning of the array, the size of the array, the dimensions and the bounds
    percent ← 5
    numberofElements ← call numberFromPercent(percent)
    range ← Range 0 and $size - numberFromPercent$
    linearIndex ← call randomElements (range)
    shift ptr_A by linearIndex
    for (j ← 0 to 5% of size)
        index ← call calculateIndex(bounds, dimensions, linearIndex, size)
        call printSeries(index, dimensions, true)
        call printSeries(ptr_A, 1, false)
        ptr_A ++
        linearIndex++;
        deleteArray(index)
    end for
```

5 Array Generation

A dynamic array was generated on the heap memory by calculating the size needed and allocating a pointer to the memory. Since arrays are sequential containers (meaning that the memory is next to each other), the arrays were flattened. The size calculation involved multiplying the bounds for each dimension of the array.

6 Traversing the Array

As mentioned in the above section, the arrays generated were flattened, so a pointer was used to move through the array. In order to do so, the start address of the array and the size of the array needed to be kept track of to prevent errors.

7 Conclusion

The is use of C-programming to write three procedures which takes a K-dimensional integer array as input has been presented. Each array element is set to zero in the first procedure while the in the second procedure, 10% of the elements of the array are set to 1's. The third procedure,randomly selects 5 % of the elements of the array the coordinate indices are then printed and the contained values. A main program was created that implemented the three procedures to dynamically create a 2D,3D, 4D and 5D array.