

ELEN4020: Data Intensive Computing

Laboratory Exercise 1

Kavilan Nair	1076342
Christopher Maree	1101946
Iordan Tchaporov	1068874
Laura West	1084327

23/02/2018

K-Dimensional Integer Array

A commonly used approach for accessing elements in a multidimensional array involves iterating with a number of nested loops. The number of loops required is equal to the dimensions of the array. This approach is computationally expensive but allows accessing the elements of a static array. Adapting this method for dynamic arrays raises the computation time and space complexity and is difficult to implement.

Arrays, regardless of dimension, are stored contiguously in computer memory. In the C programming language, this is stored using a row major order. By transforming the K co-ordinates of a K-dimensional array into a 1-dimensional co-ordinate system, any element can be accessed using a pointer, eliminating the need for multiple loops.

The program prompts the user to enter the number of dimensions and the size of each dimension. The array is then initialized with these parameters. Thereafter, procedures one, two and three are called, taking in the array and its bounds.

Procedure One

Procedure one initializes all the elements in the array to zero. This is achieved by iterating through the one dimensional co-ordinate system and setting each element to zero.

Procedure Two

Ten percent of the elements in the array are set to 1. The total number of elements is calculated by multiplying the bounds for each dimension. Every tenth element in the array is set to one, starting at the first element in the array until the number of elements in the array are exceeded. As a result, if ten percent of the total number of elements in the array is not an integer, this number is rounded up. For example a 7x7 array with 49 elements will have 5 elements uniformly set to one.

Procedure Three

This function prints the value and co-ordinate of 5% of all elements. The element indices are chosen in a uniform and random fashion. The function also ensures that the same element is not printed twice. The total number of elements in the array is calculated and a variable is assigned to 5% of this number. If the variable is not an integer, it is rounded up. An array, `printedValues`, is created that will later store the indices that are generated, its elements are initialized to -1. Random numbers within the bounds of the K-dimensional array are assigned using a random number generator that is seeded by time. These numbers are converted to the K-dimensional co-ordinates using the `getLocation()` function (explained below). Each randomly generated number is compared to all the numbers stored within the `printedValues` array, if any number has been generated previously, a new random number is generated.

`getLocation()`

The `getLocation()` function takes in three parameters, namely the number of dimensions, an array storing the size of each dimension and the linear location. The index of the Kth dimension is calculated by modding the linear position by the size of the Kth dimension. The linear position is then divided by the size of the Kth dimension. This process is then repeated, decrementing K with each iteration until K co-ordinates have been obtained.