Explanation

Hwang Chan Il

c.hwang@digipen.edu

5415279

1. Locality

Principle : Programs tend to use data and instruction with addresses near or equal to those they have used recently.

Temporal locality : Recently referenced items are likely to be referenced again in the near future.

Spatial locality : Items with nearby addresses tend to be referenced close together in time.

Ex ) unsigned char value = 0;

   for (unsigned column = 0; column < Columns; ++column)

   {

    for (unsigned row = 0; row < Rows; ++row)

    {

    Data[row][column] = value++;

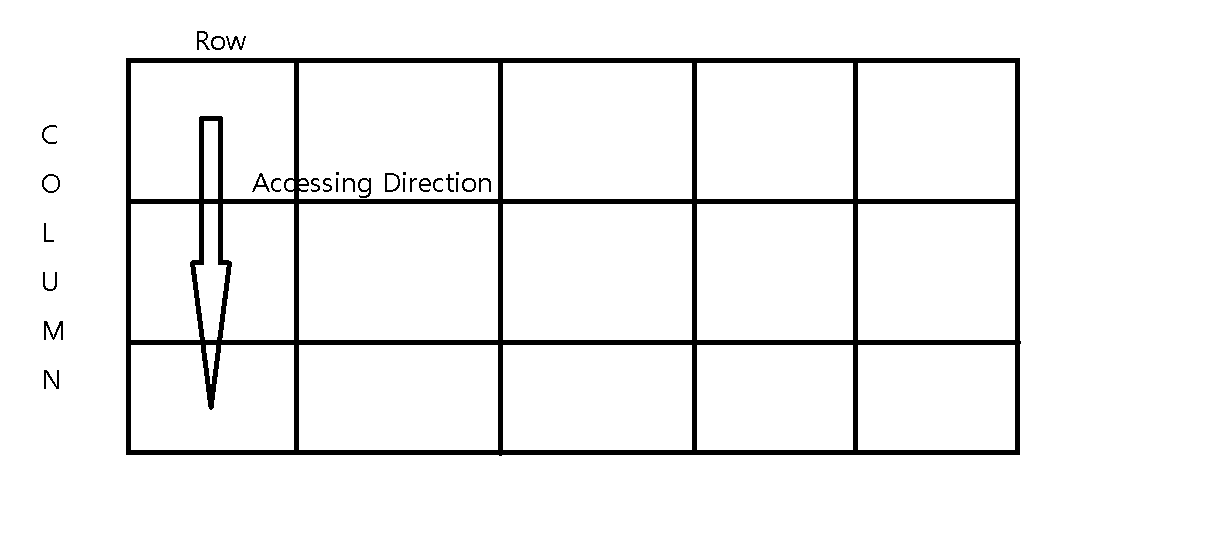
   }

   }

   WriteValiesToDisk("values.original.bin");

temporal locality = value / spatial locality = Data

Qualitative Estimates of Locality

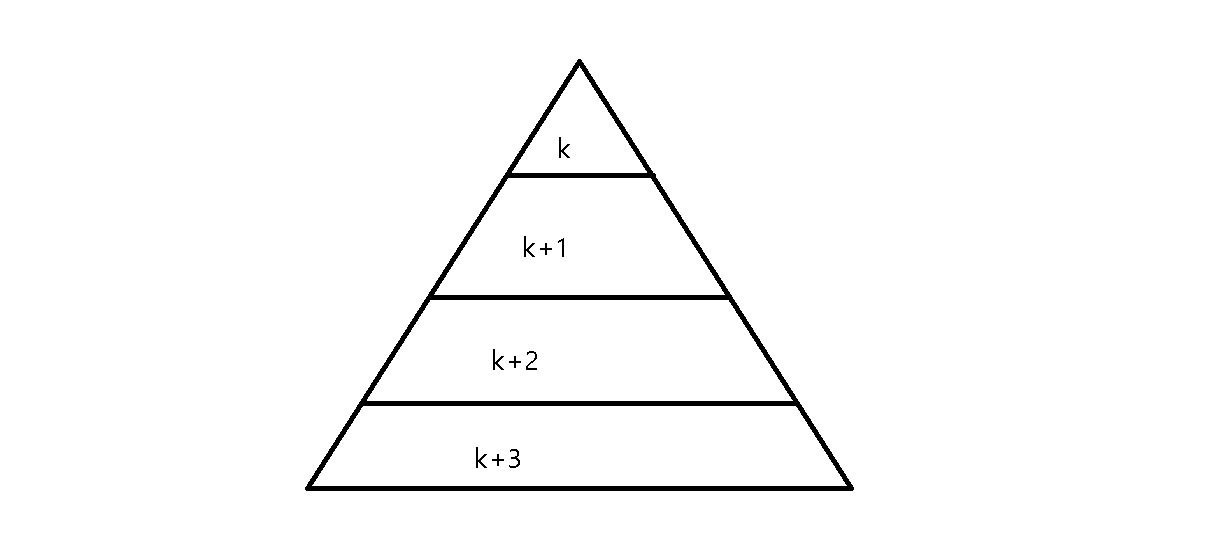
Above code has bad locality because it didn’t access nearby address. 

Memory Hierarchies

Approach for organizing memory and storage systems known as a memory hierarchy.

1. Caches :

Cache : smaller faster storage device that acts as a staging area for subset of the data in a larger, slower device.



Why memory hierarchy? : because of locality, program tends to access the data at level k more often than they access the data at level k+1(because level k+1 is slower than level k but larger and cheaper than level k)

Ex) unsigned char value = 0;

   for (unsigned column = 0; column < Columns; ++column)

   {

    for (unsigned row = 0; row < Rows; ++row)

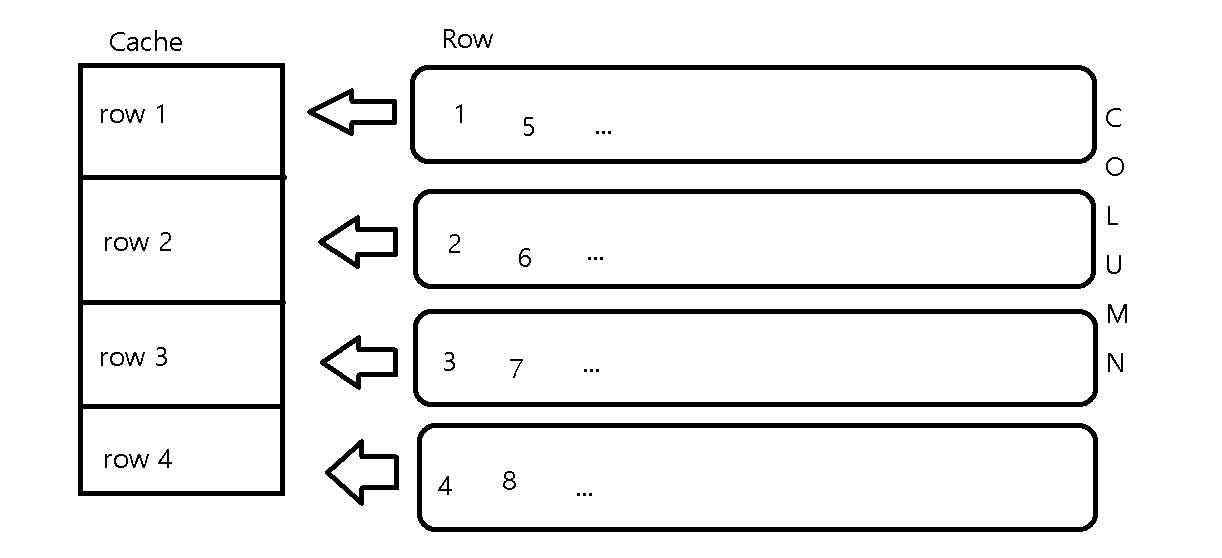
    {

    Data[row][column] = value++;

   }

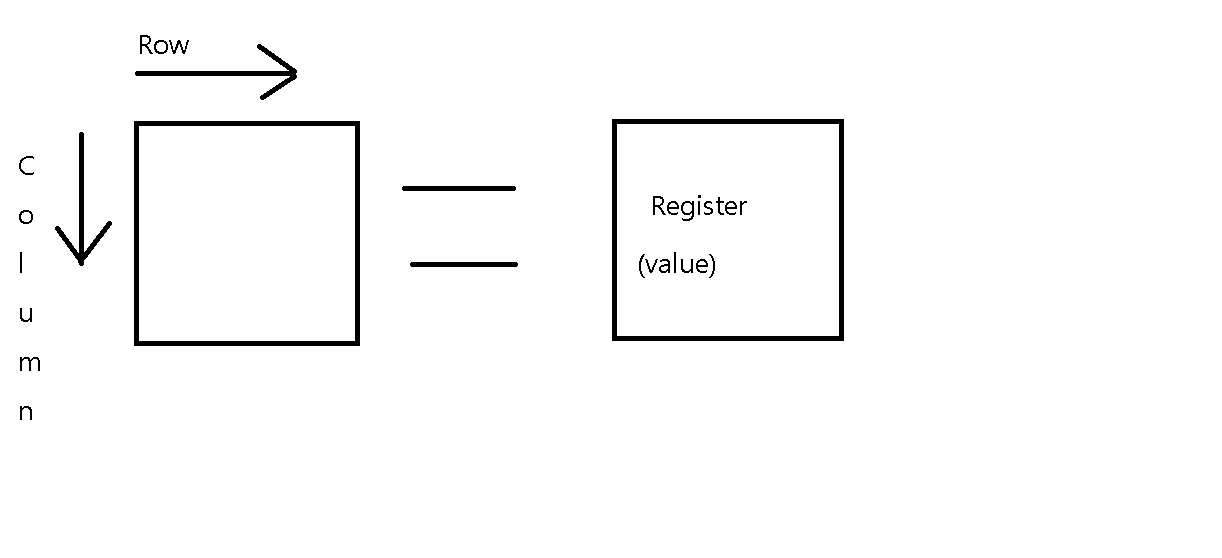
   }

   WriteValiesToDisk("values.original.bin");



In Above code, it always cache miss so it can be slow.

Rearrange loops to improve spatial locality



Stepping through columns in one row :

Sum += a[0][i];

( access successive elements, so it is fast )

Stepping through rows in one column :

Sum += a[i][0];

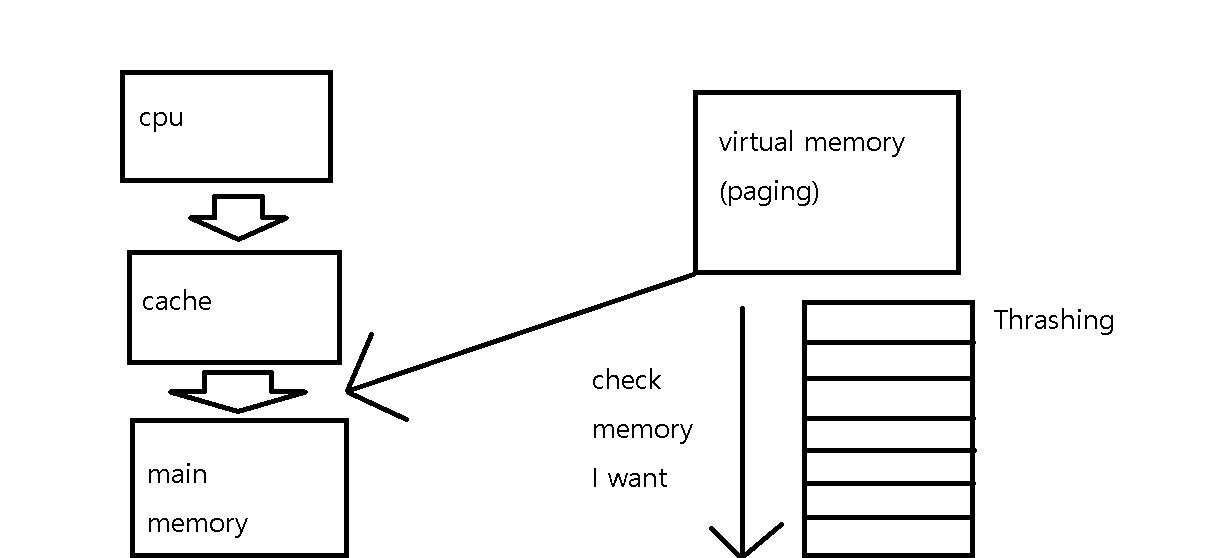
( access distant elements, so it is slow )

1. Virtual Memory for caching

Why VM? : 1 . use main memory efficiently

2. simple memory management

3. isolate address spaces



1. My Code

