Array

Hsuan-Tien Lin

Dept. of CSIE, NTU

March 3, 2020

Array

What is Array?

wikipedia: *a collection of elements, each identified by one array index*

array: numbered lockers

Memory is (Generally Viewed as) Array

pointer: stores index to memory array

Array as Memory Block in C/C++

access

- `data getByIndex(index):`

  `arr[index]`, which means
  `memory[arr + index * sizeof(data)]`

maintenance

- `construct(length):`
  - `malloc(sizeof(data)*length)` in C
  - `new data[length]` in C++
- `updateByIndex(index, data):`

  `arr[index] = data`

desired property: fast computation of address from `index`
⟹ fast random access

Array as Abstract Data Structure

access

- `data getByIndex(index)`
- `insertByIndex(index, data)`

maintenance

- `construct(length)`
- `updateByIndex(index, data)`
- `removeByIndex(index)`

implicit assumption:
`index` to address done by fast math formula

C++ STL Vector: a Growing Array

access

two more features supported with automatic growing

- `insertByIndex(index, data)`
- `insertAtBack(data)`

maintenance

one more features supported

- `removeByIndex(index)`

STL vector: a more “structured” way of using arrays

Two Dimensional Array

One Block Implementation of 2-D Array

access

`index = (row, col)`

- `data getByIndex(index)`

  address `= arr + sizeof(data) * (row*nCol+col)`

maintenance

`length = (nRow, nCol)`

- `construct(length)`

  `arr = new data[nRow * nCol]`

fast math formula: arithmetic

Array of Array Implementation of 2-D Array

access

`index = (row, col)`

- `data getByIndex(index)`

```
address = arr[row] + sizeof(data) * col
```

maintenance

`length = (nRow, nCol)`

- `construct(length)`

```
   arr = new data*[nRow]
arr[c] = new data[nCol] for all c
```

fast math formula: dereference & arithmetic

Comparison of Two Implementations

	one block	array of array
space	elements	elements & `nRow` pointers
construct	“fixed”	prop. to `nRow`
get	one deref	two deref

tradeoff: one block usually faster;
array of array often easier for programmers

Two Implementations for Triangular 2-D Array

tradeoff: one block faster & succinct;
array of array again easier for programmers

A Tale between Two Programs

row sum

```
int rowsum(){
  int i, j;
  int res = 0;
  for(i=0;i<MAXROW;i++)
    for(j=0;j<MAXCOL;j++)
      res += array[i][j];
}
```

column sum

```
int colsum(){
  int i, j;
  int res = 0;
  for(j=0;j<MAXCOL;j++)
    for(i=0;i<MAXROW;i++)
      res += array[i][j];
}
```

knowing architecture helps

Ordered Array

Definition of Ordered Array

an array of consecutive elements with ordered values

`insert` of Ordered Array

“cut in” from the back

construct of Ordered Array

insertion sort: construct with multiple insert

`update` and `remove` of Ordered Array

maintenance

- `updateByIndex(index, data)`: rotate up or down
- `removeByIndex(index)`: fill in from the back

ordered array: more maintenance efforts

Binary Search within Ordered Array

Application: Book Search within (Digital) Library

comparable elements: book IDs

Sequential Search Algorithm

similar to `getMinIndex`

Ordered Array: Sequential Search Algorithm with Cut

ordered: possibly easier to declare not found

Ordered Array: Binary Search Algorithm

"cut" multiple times by fast random access to the middle