Arrays

Kuan-Yu Chen (陳冠宇)

Review

- Space and Time complexity
  - Big-Oh
  - Omega
  - Theta
  - Little-Oh
  - Little-Omega

- The master method theorem: $T(n) = aT\left(\frac{n}{b}\right) + f(n)$

  1. If $f(n) = O(n^{\log_b a-\epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
  2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log_2 n)$
  3. If $f(n) = \Omega(n^{\log_b a+\epsilon})$ for some constant $\epsilon > 0$, and if $af(\frac{n}{b}) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

Online Link

- Webex: https://ctld.webex.com/meet/kychen

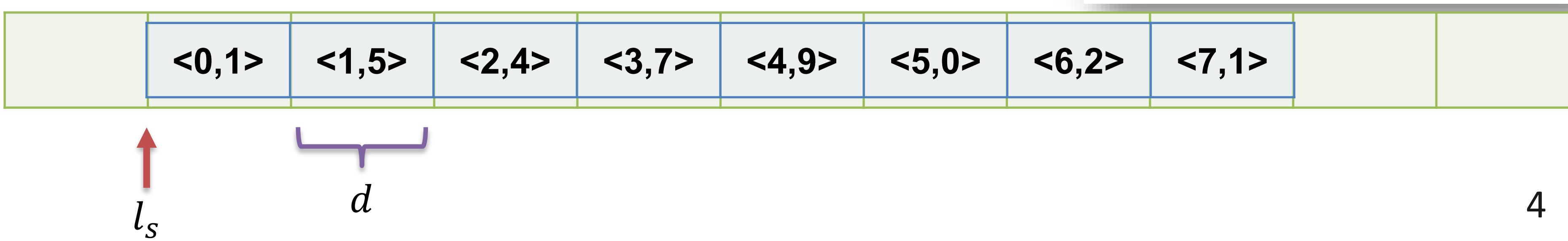
Array

- An array is a set of pairs $< index, value >$, such that each index is associated with a value
  - In C++, the index is starting at 0

1	2	3	4	5	6	7	8
<0,1>	<1,5>	<2,4>	<3,7>	<4,9>	<5,0>	<6,2>	<7,1>

  - The array A maps into continuous memory locations
    - $A_1 = A[0] = 1$
    - $A_6 = A[5] = 0$
    - Location of A_1 is l_s, i.e., $A[0]$
    - Location of A_6 is $l_s + 5d$, i.e., $A[5]$

Data Type	Size in Bytes
char	1
int	2
float	4
double	8



Example

- Given an array

  int marks[] = {99, 67, 78, 56, 88, 90, 34, 85}

  please calculate the address of marks[4] if the base address = 1000

  – An integer value requires 2 bytes

  – $l_s + 4d = 1000 + 4 \times 2 = 1008$

99	67	78	56	**88**	90	34	85
marks[0]	marks[1]	marks[2]	marks[3]	**marks[4]**	marks[5]	marks[6]	marks[7]
1000	1002	1004	1006	**1008**	1010	1012	1014

Declare an Array

- The elements of an array can be initialized at the time of declaration

```
int marks [5] = {90, 45, 67, 85, 78};
```

[0]	[1]	[2]	[3]	[4]
90	45	67	85	78

```
int marks [5] = {90, 45};
```

[0]	[1]	[2]	[3]	[4]
90	45	0	0	0

Rest of the elements are filled with 0's

```
int marks [] = {90, 45, 72, 81, 63, 54};
```

[0]	[1]	[2]	[3]	[4]	[5]
90	45	72	81	63	54

```
int marks [5] = {0};
```

[0]	[1]	[2]	[3]	[4]
0	0	0	0	0

2D Array.

- 2D array is also named matrix

- A matrix is a mathematical object that arises in may physical problems
  - A general matrix consists of m rows and n columns of numbers
  - If m is equal to n, we call the matrix square

$$
\begin{array}{c}
\\ \text{row 1} \\ \text{row 2} \\ \text{row 3} \\ \text{row 4} \\ \text{row 5}
\end{array}
\begin{array}{c}
\begin{array}{ccc} \text{col 1} & \text{col 2} & \text{col 3} \end{array} \\
\left[\begin{array}{rrr}
-27 & 3 & 4 \\
6 & 82 & -2 \\
109 & -64 & 11 \\
12 & 8 & 9 \\
48 & 27 & 47
\end{array}\right]
\end{array}
$$

(a)

$$
\begin{array}{c}
\\ \text{row 1} \\ \text{row 2} \\ \text{row 3} \\ \text{row 4} \\ \text{row 5} \\ \text{row 6}
\end{array}
\begin{array}{c}
\begin{array}{cccccc} \text{col 1} & \text{col 2} & \text{col 3} & \text{col 4} & \text{col 5} & \text{col 6} \end{array} \\
\left[\begin{array}{rrrrrr}
15 & 0 & 0 & 22 & 0 & -15 \\
0 & 11 & 3 & 0 & 0 & 0 \\
0 & 0 & 0 & -6 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
91 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 28 & 0 & 0 & 0
\end{array}\right]
\end{array}
$$

(b) Sparse Matrix

2D Array..

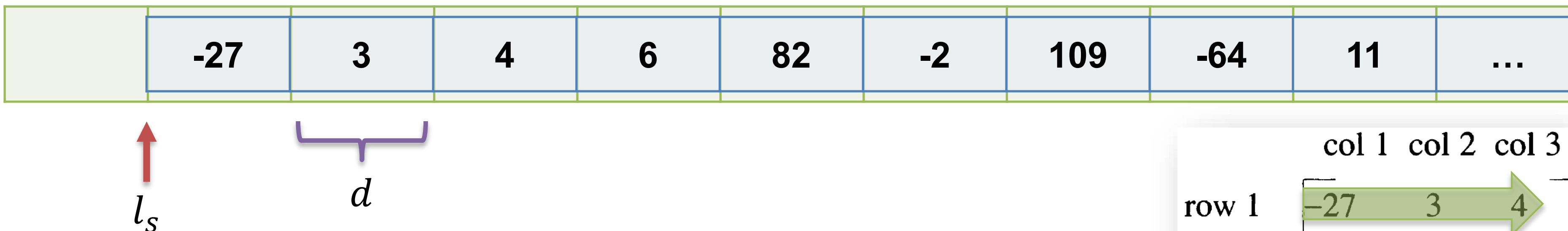
- For a matrix A, we can work with any element by writing $A_{i,j} = A[i-1][j-1]$, and the element can be found very quickly
  - It should be noted that the index is starting at 0 in C++
    - $A_{1,1} = A[0][0] = -27$
    - $A_{3,2} = A[2][1] = -64$

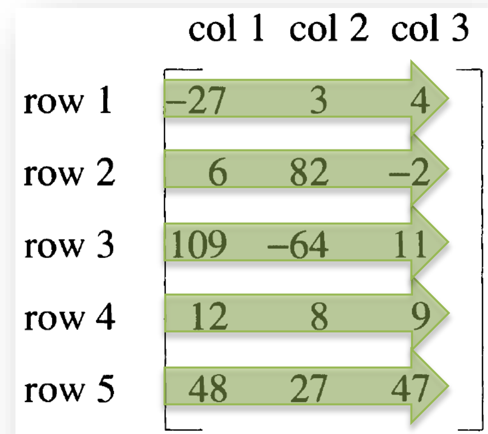
- There are two ways to store a matrix in the memory
  - Row-major
  - Column-major

$$
\begin{array}{l}
 \\
\text{row } 1 \\
\text{row } 2 \\
\text{row } 3 \\
\text{row } 4 \\
\text{row } 5
\end{array}
\begin{array}{c}
\begin{array}{ccc} \text{col } 1 & \text{col } 2 & \text{col } 3 \end{array} \\
\begin{bmatrix}
-27 & 3 & 4 \\
6 & 82 & -2 \\
109 & -64 & 11 \\
12 & 8 & 9 \\
48 & 27 & 47
\end{bmatrix}
\end{array}
$$

2D Array – Row-Major

- Given a matrix A, row-major will rearrange all of the elements, $A_{1,1}, A_{1,2}, A_{1,3}, A_{2,1}, \dots, A_{5,3}$, and then store in the memory





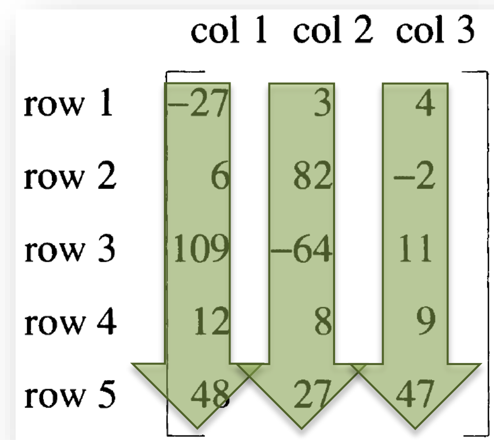
  - $A_{1,1} = A[0][0] = -27$
  - $A_{4,3} = A[3][2] = 9$
  - Location for $A_{4,3} = l_s + [(4-1) \times 3 + (3-1)] \times d$
  - Location for $A_{i,j} = l_s + [(i-1) \times 3 + (j-1)] \times d$
  - Generally, given an $m \times n$ matrix A, location for $A_{i,j}$ is $l_s + [(i-1) \times n + (j-1)] \times d$

2D Array – Column-Major

- Given a matrix A, row-major will rearrange all of the elements, $A_{1,1}, A_{2,1}, A_{3,1}, A_{4,1}, \ldots, A_{5,3}$, and then store in the memory

	-27	6	109	12	48	3	82	-64	8	…

l_s d



  - $A_{1,1} = A[0][0] = -27$
  - $A_{4,3} = A[3][2] = 9$
  - Location for $A_{4,3} = l_s + [(3-1) \times 5 + (4-1)] \times d$
  - Location for $A_{i,j} = l_s + [(j-1) \times 5 + (i-1)] \times d$
  - More generally, given an $m \times n$ matrix A, location for $A_{i,j}$ is $l_s + [(j-1) \times m + (i-1)] \times d$

Examples – 1

- Given a 2D array A, the location for $A_{3,2}$ is 1110, and the location for $A_{2,3}$ is 1115. If the size for each element is 1, please indicate the location for $A_{5,4}$.
  - Since the location for $A_{3,2}$ is 1110 and the location for $A_{2,3}$ is 1115, so the storage method is column-major!
  - Assume the size of row is m

		$A_{2,3}$ 1115	
	$A_{3,2}$ 1110		

$$Location(A_{i,j}) = l_s + \left((j-1) \times m + (i-1)\right) \times d$$

$$Location(A_{2,3}) = l_s + \left((3-1) \times m + (2-1)\right) \times 1 = 1115$$

$$Location(A_{3,2}) = l_s + \left((2-1) \times m + (3-1)\right) \times 1 = 1110$$

$$l_s + 2 \times m + 1 = 1115$$

$$l_s + m + 2 = 1110$$

$$l_s = 1102 \qquad m = 6$$

$$Location(A_{5,4}) = 1102 + \left((4-1) \times 6 + (5-1)\right) \times 1 = 1124$$

Examples – 2

- Given a matrix A, if $Location(A_{2,3}) = 18$, $Location(A_{3,2}) = 28$, and $Location(A_{1,1}) = 2$, please calculate $Location(A_{4,5}) =?$
  - It is row-major
  - $l_s = Location(A_{1,1}) = 2$

$A_{1,1}$ 2			
		$A_{2,3}$ 18	
	$A_{3,2}$ 28		

$$Location(A_{i,j}) = l_s + [(i-1) \times n + (j-1)] \times d$$

$$Location(A_{2,3}) = 18 = 2 + [(2-1) \times n + (3-1)] \times d = 2 + n \times d + 2 \times d$$

$$Location(A_{3,2}) = 28 = 2 + [(3-1) \times n + (2-1)] \times d = 2 + 2 \times n \times d + d$$

$$10 = n \times d - d$$

$$d = 2 \qquad n = 6$$

$$Location(A_{4,5}) = 2 + [(4-1) \times 6 + (5-1)] \times 2 = 46$$

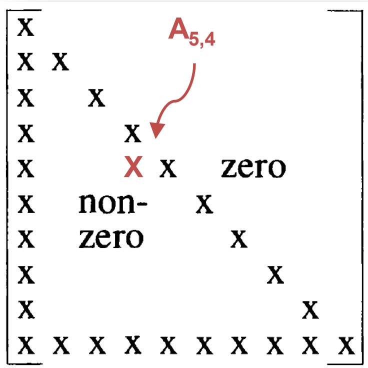
Lower-Triangular Matrix.

- Given a **square matrix** A with m rows
  - The maximum number of nonzero terms in row i is i
  - $A_{i,j} = 0, if\ i < j$
  - Such a matrix is lower-triangular matrix
  - The total number of non-zero terms is $1 + 2 + \cdots + m = \frac{m(m+1)}{2}$
  - For large m, it would be worthwhile to save the memory space by only storing non-zero part
    - Row-major

$$\forall i \geq j,\ Location(A_{i,j}) = l_s + \left[\left(\frac{(1+(i-1))}{2} \times (i-1)\right) + (j-1)\right] \times d$$
$$= l_s + \left[\frac{i \times (i-1)}{2} + j - 1\right] \times d$$

- Column-major

$$\forall i \geq j,\ Location(A_{i,j}) = l_s + \left[\frac{(1+m)}{2} \times m - \frac{(1+(m-j+1))}{2} \times (m-j+1) + (i-j)\right] \times d$$
$$= l_s + \left[i + m \times (j-1) - \frac{j \times (j-1)}{2} - 1\right] \times d$$

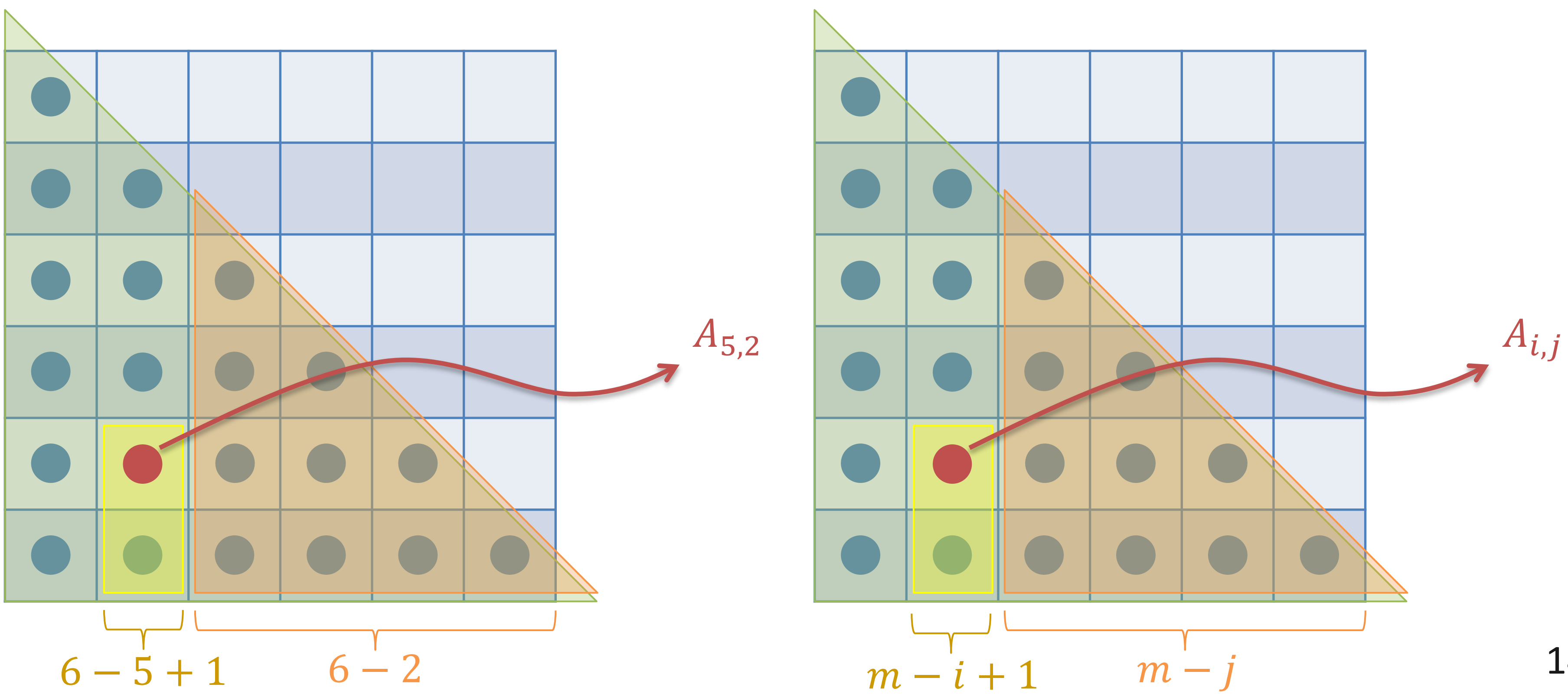


lower triangular

Lower-Triangular Matrix..

- The inference for column-major

$$\forall i \geq j,\ Location(A_{i,j}) = l_s + \left[\frac{(1+m)}{2} \times m - \frac{(1+(m-j))}{2} \times (m-j) - (m-i+1)\right] \times d$$
$$= l_s + \left[i + m \times (j-1) - \frac{j \times (j-1)}{2} - 1\right] \times d$$



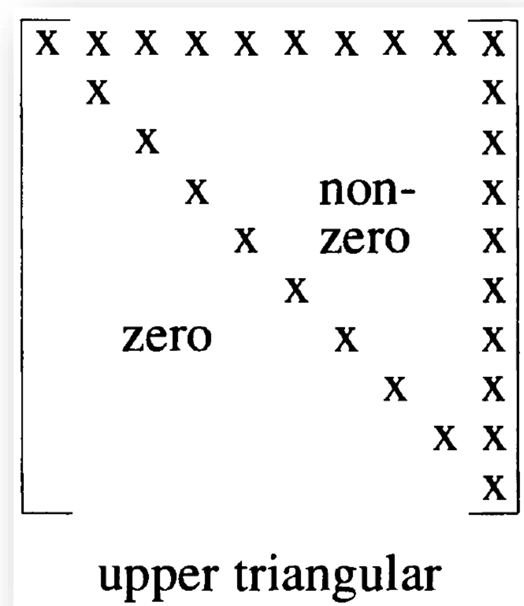
Upper-Triangular Matrix

- Given a **square matrix** A with n columns
  - The maximum number of nonzero terms in column j is j
  - $A_{i,j} = 0, if\ i > j$
  - Such a matrix is upper-triangular matrix
  - The total number of non-zero terms is $1 + 2 + \cdots + n = \frac{n(n+1)}{2}$
  - For large n, it would be worthwhile to save the memory space by only storing non-zero part
    - Row-major

$$\forall i \le j,\ Location\left(A_{i,j}\right) = l_s + \left[j + n \times (i-1) - \frac{i \times (i-1)}{2} - 1\right] \times d$$

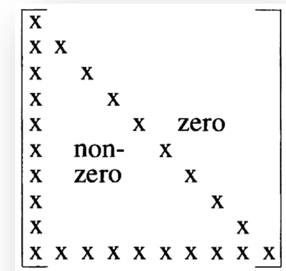
    - Column-major

$$\forall i \le j,\ Location\left(A_{i,j}\right) = l_s + \left[\frac{j \times (j-1)}{2} + i - 1\right] \times d$$



Example

- Given a lower-triangular matrix A, the size of columns and rows are both 100. If we leverage a memory B to store the matrix by using row-major



lower triangular

1. How many memory elements do we need?

$$1 + 2 + 3 + \cdots + 100 = \frac{1+100}{2} \times 100 = 5050$$

2. Which memory block will store $A_{70,50}$? (the starting number in the memory is 1)

$$Location(A_{i,j}) = l_s + \left[\frac{i \times (i-1)}{2} + j - 1\right] \times d$$

$$1 + [(1 + 2 + 3 + \cdots + 69) + 49] \times 1 = 1 + \left[\frac{1+69}{2} \times 69 + 49\right] \times 1 = 2465$$

3. Which element in A will be stored in B with the starting address 152?

$$1 + [(1 + 2 + 3 + \cdots + (i-1)) + (j-1)] \times 1 = 1 + \left[\frac{1+(i-1)}{2} \times (i-1) + (j-1)\right] \times 1 = 152$$

$$\because i \geq j$$

$$\therefore i = 17 \quad j = 16$$

Questions?

kychen@mail.ntust.edu.tw