Omega Ω

if $f(n) = \Omega(g(n))$

$\exists \; c > 0, \; n_0 > 0$

$0 \leq c\,g(n) \leq f(n) \quad \forall n \geq n_0$

Lower

best case.

Big O O

if $f(n) = O(g(n))$

$\exists \; c > 0, \; n_0 > 0$

$f(n) \leq c \cdot g(n) \quad \forall n \geq n_0$

upper

worst case

Omega Notation

Example 2.5 Show that $5n^2 + 10n = \Omega(n^2)$.

Solution By the definition, we can write

$$0 \le cg(n) \le h(n)$$
$$0 \le cn^2 \le 5n^2 + 10n$$

Dividing by n^2

$$0/n^2 \le cn^2/n^2 \le 5n^2/n^2 + 10n/n^2$$
$$0 \le c \le 5 + 10/n$$

Now, $\lim_{n\to\infty} 5 + 10/n = 5$.

Therefore, $0 \le c \le 5$.

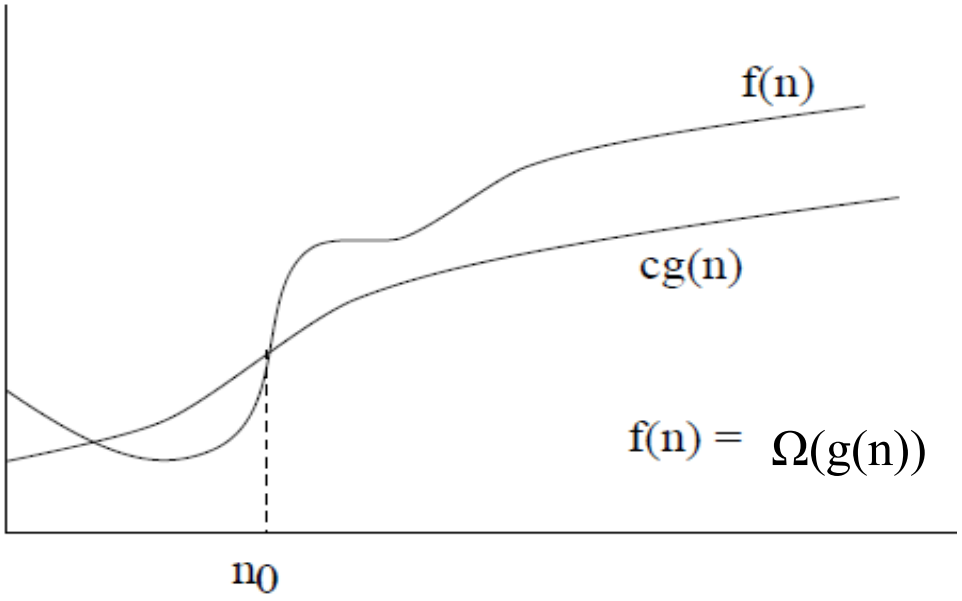
Hence, $c = 5$

Now to determine the value of n_0

$$0 \le 5 \le 5 + 10/n_0$$
$$-5 \le 5 - 5 \le 5 + 10/n_0 - 5$$
$$-5 \le 0 \le 10/n_0$$

So $n_0 = 1$ as $\lim_{n\to\infty} 1/n = 0$

Hence, $5n^2 + 10n = \Omega(n^2)$ for $c=5$ and $\forall\ n \ge n_0=1$.



Ex to prove $5n^2 + 10n = \Omega(n^2)$

$f(n)$ $g(n)$

Aim : C, n_0 satisfy $0 \le C\,g(n) \le f(n)$ ① $f(n) \ge 0$ → time.

$0 \le Cn^2 \le 5n^2 + 10n$ (2)

$\because n^2 \ge 0$ $n > 0$ (5)

$0 \le C \le 5 + \frac{10}{n}$ (3) (4) $\sqrt{}$ $n \to \infty$

$\lim_{n\to\infty} \frac{10}{n} = 0$

$\frac{10}{n}$

$\Rightarrow 0 \le C \le 5$ $n \to \infty$

only one c.

Assume $C = 5$ then find n_0

$5n^2 \le 5n^2 + 10n \Rightarrow n \ge 0$ & $n > 0$ ⇒ choose $n_0 = 1$

$C = 4$

$4n^2 \le 5n^2 + 10n$ $\because n > 0$

$-n^2 \le 10n \Rightarrow -n \le 10$ always hold

$n_0 > 0$

$n_0 = 1$ or 2 or 3

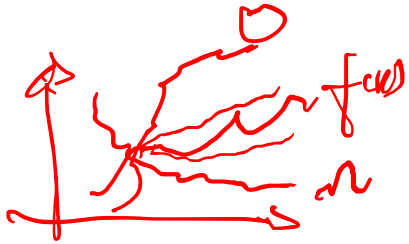
Theta Notation

Theta notation provides an asymptotically tight bound for f(n).

Θ notation is simply written as, f(n) = Θ(g(n)), where n is the problem size and Θ(g(n)) = {h(n): ∃ positive constants c_1, c_2 and n_0 such that $0 \leq c_1 g(n) \leq h(n) \leq c_2 g(n)$, ∀ $n \geq n_0$}.

Hence, we can say that Θ(g(n)) comprises a set of all the functions h(n) that are between $c_1 g(n)$ and $c_2 g(n)$ for all values of $n \geq n_0$.

Conclusion: Function Growth Rates: Mathematical Definitions.



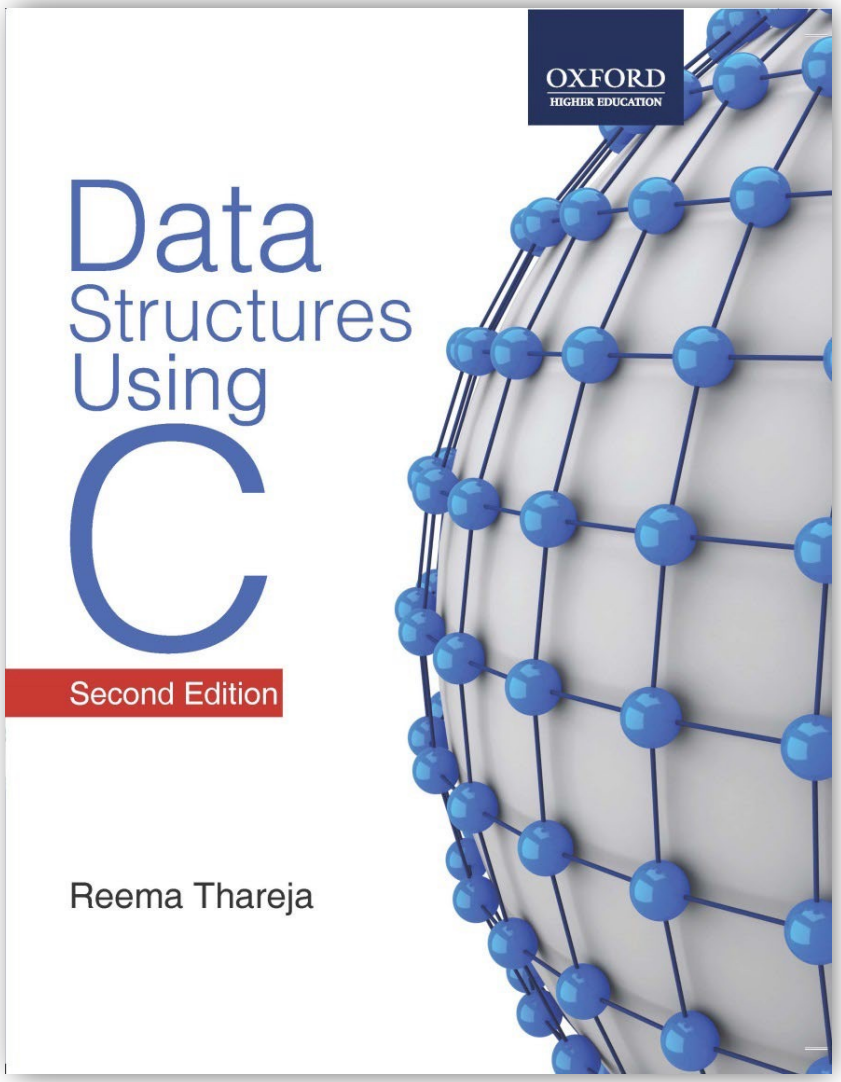
worst

- Big-O expression, f(n) = O(g(n)) says that f(n) grows NOT FASTER than g(n)

- Big-Omega, $f(n) = \Omega(g(n))$, says that f(n) grows AT LEAST AS SLOW (and maybe faster) as g(n)

- Theta says the two functions grow at the same rate

- Thank you!



Data Structures Using C, 2e

Reema Thareja

Chapter 6

Linked Lists

Introduction

- We have studied that an array is a linear collection of data elements in which the elements are stored in consecutive memory locations.
- While declaring arrays, we have to specify the size of the array, which will restrict the number of elements that the array can store.
- But what if we are not sure of the number of elements in advance?
- Moreover, to make efficient use of memory, the elements must be stored randomly at any location rather than in consecutive locations. So, there must be a data structure that removes the restrictions on the maximum number of elements and the storage condition to write efficient programs.
- Linked list is a data structure used to implement other data structures, such as stacks, queues and their variations.

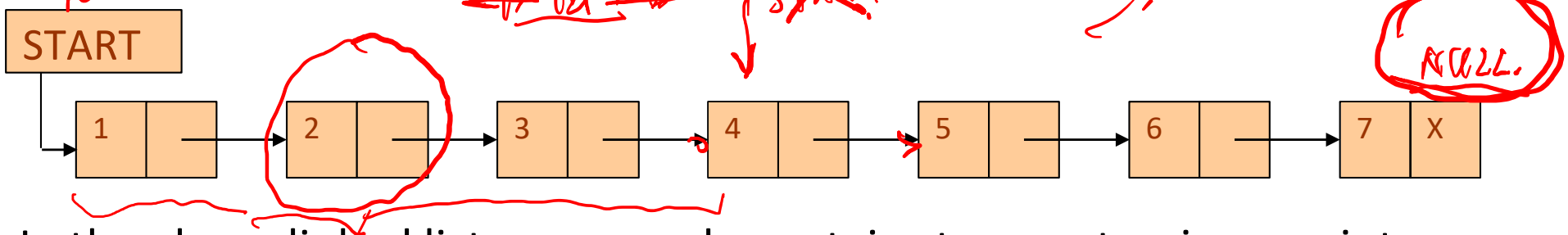
Linked list

- store randomly at any location → efficiency.
- Dynamic length.

Array

- in consecutive memory location
- fixed size.

Simple Linked List



- In the above linked list, every node contains two parts – i.e. one integer and the other a pointer to the next node.
- The left part of the node which contains data may include a simple data type, an array or a structure.
- The right part of the node contains a pointer to the next node (or address of the next node in sequence).
- The last node will have no next node connected to it, so it will store a special value called NULL.

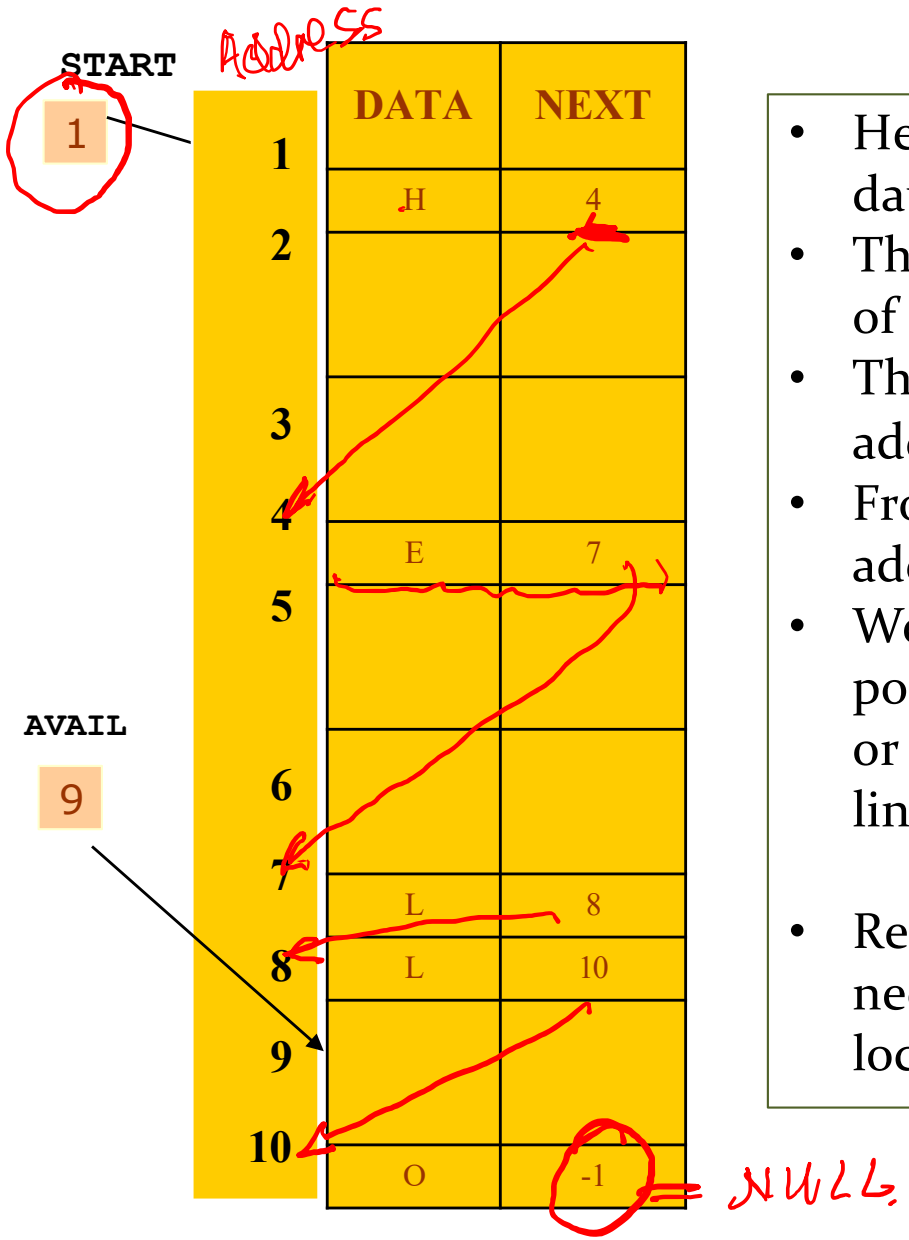
Simple Linked List

- We can traverse the entire linked list using a single pointer variable called START.
- The START node contains the address of the first node; the next part of the first node in turn stores the address of its succeeding node.
- Using this technique, the individual nodes of the list will form a chain of nodes.
- If START = NULL, this means that the linked list is empty and contains no nodes.
- In C, we can implement a linked list using the following code:

```
struct node
{
    int data;
    struct node *next;
};
```

Note: Linked lists provide an efficient way of storing related data and perform basic operations such as insertion, deletion, and updation of information at the cost of extra space required for storing the address of the next node.

One Linked List in Memory



- Here, in this example, START = 1, so the first data is stored at address 1, which is H.
- The corresponding NEXT stores the address of the next node, which is 4.
- The second data element obtained from address 4 is E.
- From the entry in the NEXT, we get the next address, that is 7, and fetch L as the data.
- We repeat this procedure until we reach a position where the NEXT entry contains –1 or NULL, as this would denote the end of the linked list.

- Remember that the nodes of a linked list need not be in consecutive memory locations.