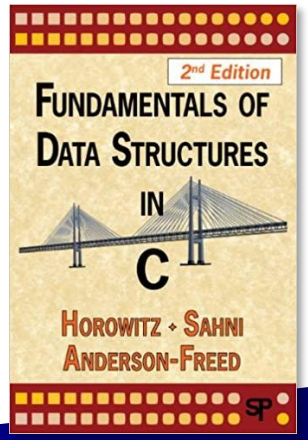
Data Structure

Linked List

Shin Hong

Apr 21, 2020

Chapter 4. Linked List



Array vs. Linked List

- An **array** stores successive data objects are located fixed distance apart in a continuous memory region
  - Pros: random access to a certain index is possible
  - Cons: changing the index of an object (by insertion and by deletion) possibly results in moves of many objects

- A **linked list** stores each element with a pointer (memory address) of the successor to form a chain of elements
  - Pros: insertion or deletion of a specific element can be done without moving other elements
  - Cons: an element at an index can be accessed only through its precessor

Singly Linked List

- array-based implementation
  - index of first data element
  - the link of a data element indicates the index of next data
    - link is zero for last element

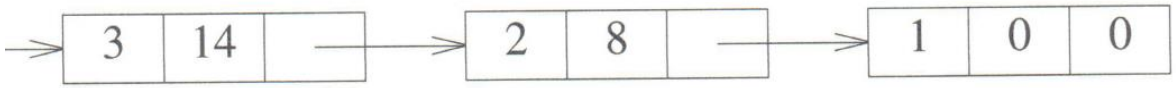
- pointer-based implementation
  - self-referential structure
  - header node
    https://github.com/hongshin/DataStructure/tree/linkedlist/ver3
  - circular list
    https://github.com/hongshin/DataStructure/tree/linkedlist/ver4

	data	*link*
1	HAT	15
2		
3	CAT	4
4	EAT	9
5		
6		
7	WAT	0
8	BAT	3
9	FAT	1
10		
11	VAT	7
	.	.
	.	.
	.	.

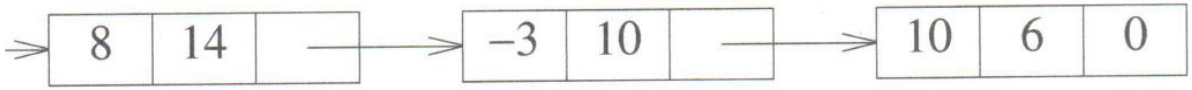
Polynomials (1/2)

- Use a linked list to represent a polynomial in one variable
  - Polynomial: $A(x) = a_{m-1}x^{e_{m-1}} + a_{m-2}x^{e_{m-2}} + \ ... + a_0x^{e_0}$ where a_i are non-zero coefficients and e_i are positive exponents
  - A term is a pair of a coefficient and a exponent
  - Represent $A(x)$ as a list of terms in the ascending order of exponents
  - Examples:

$a = 3x^{14} + 2x^8 + 1$



$b = 8x^{14} - 3x^{10} + 10x^6$



Polynomials (2/2)

- Define arithmetic operation as linked list manipulation
  - add
    - merge two ordered lists
    - add coefficients for two terms of the same exponent
  - multiplication
    - multipy two terms by creating a new term with the multiplication of the two coefficients and the addition of the two exponents
    - add the multiplications of all pairs of terms of two polynomials

Doubly Linked List

- Each node has two links, one in the forward direction and the oether in the backward directions
  - we can go back and forth easily along a linked list
  - take a constant amount of time for removing a node or inserting a new node after a node

- A header node can be used for ensuring link invariants

```
node == node->left->right == node->right->left
```