Sorted and Doubly Linked Lists

University of Massachusetts Amherst
ECE 242 – Data Structures and Algorithms
Lecture 13

Variants of linked lists

• Today:
  – Sorted linked list
  – Doubly linked list

• Variants require changes
  – Sorted linked list: how objects are arranged in list
  – Doubly linked list: list objects themselves
Sorted linked list

• What do we need to change for a sorted linked list?

Insertion in sorted linked list

• Revised insertion operation

```java
public void insert(int item) {
    ListObject o = new ListObject(item);
    ListObject previous = first;
    if (first == null) { // list is empty
        first = o;
        return;
    }
    if (first.data > item) { // item needs to go at first spot of list
        o.next = first;
        first = o;
        return;
    }
    while (previous.next != null && previous.next.data < item) {
        previous = previous.next;
    }
    if (previous.next == null) { // check if at end of list
        previous.next = o;
        return;
    } else { // implement insertion
        o.next = previous.next;
        previous.next = o;
        return;
    }
}
```
Sorted linked list

• Code:
  – List object: ListObject.java
  – List functionality: SortedLinkedList.java
  – main() method: SortedLinkedListExample.java

Observations

• Insertion in singly linked list is complex
  – Doubly linked list improves problem
• Complexity of operations
  – Unsorted linked list
    • Insertion
    • Removal of specific element
  – Sorted linked list
    • Insertion
    • Removal of specific element
  – Can we use binary search on sorted linked list?
Observations

• Insertion in singly linked list is complex
  – Doubly linked list improves problem
• Complexity of operations
  – Unsorted linked list
    • Insertion: O(1)
    • Removal of specific element: O(n)
  – Sorted linked list
    • Insertion: O(n)
    • Removal of specific element: O(n)
  – Can we use binary search on sorted linked list? Yes but not efficient

Doubly linked list

• Main idea:
  – Have pointer to next and previous element in list

• What do we need to change from linked list?
Doubly linked list

- DoublyListObject has additional pointer

```java
public class DoublyListObject {
    public int data;
    public DoublyListObject prev;
    public DoublyListObject next;

    public DoublyListObject(int item) {
        data = item;
        prev = null;
        next = null;
    }
}
```

Doubly linked list

- Code:
  - List object: DoublyListObject.java
  - List functionality: DoublyLinkedList.java
  - main() method: DoublyLinkedListExample.java
Insertion in doubly linked list

```java
public class DoublyLinkedList {
    private DoublyListObject first;

    public DoublyLinkedList() {
        first = null;
    }

    public void insertFirst(int item) {
        DoublyListObject o = new DoublyListObject(item);
        o.next = first;
        if (first != null) { // list has elements
            first.prev = o;
        }
        first = o;
    }
}
```

public void findAndDelete(int item) {

  DoublylistObject current = first;
  while (current!=null && current.data!=item) { // traversal
    current = current.next;
  }
  if (current==null) { // end of list and no match
    return;
  } else { // first element
    if (current.prev==null) { // not first element
      current.prev.next = current.next;
    } else { // first element
      first = current.next;
    }
  }
  if (current.next==null) { //not last element
    current.next.prev = current.prev;
  }
  return;
}
}
Doubly linked list tradeoff

- **Benefits**
  - Easier removal of objects

- **Drawback**
  - Need to manage multiple pointers

- **Other variants**
  - More complex lists simplify some specific operations
    - E.g., double-ended, doubly linked list

Double-ended doubly linked list

- **Code:**
  - List object: DoublyListObject.java
  - List functionality:
    DoublyLinkedDoubleEndedList.java
  - `main()` method:
    DoublyLinkedDoubleEndedListExample.java
Next Steps

• Discussions Thursday
• Lecture on Friday