Check if Linked List is Palindrome

Shikha Nair

February 8, 2018

Question

Complete the method boolean isPalindrome(Node head)
Given the head node of a singly-linked list, return true if the list is a palindrome, and false if it is not.

Explanation and Algorithm

Using a Stack
We will focus on the method which uses a stack to determine if the linked list is a palindrome. First, traverse the linked list and add each node to a stack. Next, traverse the list once more and for each node, pop a node off the stack and check if the two match.

Reversing the List
Another way to solve this in O(n) time is to create a second list which is the reverse of the given, and check if the two lists are the same. Another interesting way to use a reverse list, which also uses no extra space, is to go to the middle of the list, reverse the second half of the list, and then check if the first and second half are identical.

Hints

1. What main feature of a palindrome would be helpful to know to figure out how to solve this problem?
2. Do you need some other data structure to solve this?
3. Would a second linked list or a stack be helpful?
4. If you use a stack, how do you compare the data added to the stack with the data in the linked list?

Code

Stack Method

```
boolean isPalindrome(Node head)
    {
        Stack<Node> stack = new Stack<Node>();

        Node tmp = head;

        while(tmp!=null){

          stack.push(tmp);

          tmp = tmp.next;

        }

        Node x;

        tmp = head;

        while(tmp!=null){

          x = stack.pop();

          if(tmp.data!=x.data){
              return false;
          }

        tmp = tmp.next;

       }

        return true;

    }
```

Big O analysis

The stack method use O(n) time as it traverses the linked list twice and the stack once, however it requires O(n) extra space to create the stack.

Sources

Question, answer and other material taken from GeekforGeeks.org.