CENG 215 Veri Yapıları Laboratuvarı
Ödev 2: İkili Ağaç Dolaşma İşlemleri

Şevket Umut ÇAKIR

6 Aralık 2023

Ödevde ikili arama ağaçlarındaki dolaşma işlemleri ile ilgili iki adet metodun yazılması istenmektedir.

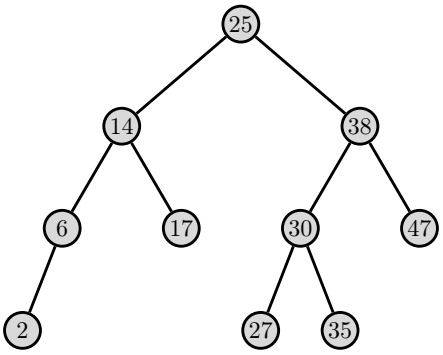
1 Algoritmalar

Bu bölümde ödevle ilgili problem ve algoritmalar anlatılmaktadır.

1.1 Seviye Sıralı Dolaşma(Level Order Traversal)

Seviye sıralı dolaşma, bir ağacın veya grafın düğümlerini kök düğümden başlayarak seviye seviye dolaşmayı ifade eder. Bu tarama yöntemi genellikle genişlik öncelikli arama (BFS) olarak da adlandırılır. Seviye sıralı dolaşmada, bir seviyedeki tüm düğümler önce ziyaret edilir ve ardından bir alt seviyeye geçilir.

Örneğin, bir ağaçtaki seviye sıralı dolaşma, kök düğümden başlayarak her seviyedeki düğümleri soldan sağa doğru ziyaret etme işlemidir. Bu, ağacın yapısını ve içeriğini düzenli bir şekilde keşfetmemize yardımcı olur. Şekil 1'de verilen örnek ağaç için seviye sıralı dolaşma, **25, 14, 38, 6, 17, 30, 47, 2, 27, 35** olacaktır.



Şekil 1: Örnek ağaç

Seviye sıralı dolaşma için kuyruk(queue) veri yapısı kullanılmaktadır ve yapısı Algoritma 1'de verilmiştir.

Algorithm 1: Level Order Traversal

```
1  Function levelOrderTraversal(root):
2      queue ← new Queue();
3      result ← [];
4      if root is null then
5          return result;
6      end
7      queue.enqueue(root);
8      while ¬queue.isEmpty() // ¬: değil işaretidir
9       do
10         node ← queue.dequeue();
11         result.append(node.val);
12         if node.left is not null then
13             queue.enqueue(node.left);
14         end
15         if node.right is not null then
16             queue.enqueue(node.right);
17         end
18     end
19     return result;
```

İmzası aşağıda verilen ve kök düğümü parametre olarak verilen `levelOrder` metodunu yazınız.

```
public static <T> String levelOrder(BTNode<T> node)
```

1.2 Preorder Dolaşma Kullanarak İkili Arama Ağacı Oluşturma

Önce değer(preorder) dolaşması verilen ikili arama ağaçları Algoritma 2 kullanılarak oluşturulabilir. Bu metotta Algoritma 2'yi kullanarak önce değer(preorder) sırasının bulunduğu diziyi parametre olarak alan ve oluşturulan ağacın kök düğümünü döndüren `bstFromPreorder` metodunu yazınız. Metodun imzası aşağıda verilmiştir.

```
public static <T extends Comparable<T>> BTNode<T> bstFromPreorder(T[] preorder)
```

Algorithm 2: Preorder Dolaşma Kullanarak İkili Arama Ağacının(BST) Oluşturulması

```
1  Function bstFromPreorder(preorder[]):
2      if preorder is empty then
3          return null;
4      end
5      root ← new BTNode(preorder[0]);
6      stack ← new Stack();
7      stack.push(root);
8      for i ← 1 to preorder.length − 1 do
9          node ← new BTNode(preorder[i]);
10         current ← stack.peek();
11         while ¬stack.isEmpty() and preorder[i] > stack.peek().val // ¬: değil işaretidir
12          do
13             current ← stack.pop();
14         end
15         if preorder[i] < current.val then
16             current.left ← node
17         end
18         else
19             current.right ← node
20         end
21         stack.push(node);
22     end
23     return root;
```

2 Ödev Sınıfları

Listing 1: BTNode.java

```java
public class BTNode<T> {
    public BTNode<T> left;
    public BTNode<T> right;
    public T value;

    public BTNode(T value, BTNode<T> left, BTNode<T> right) {
        this.left = left;
        this.right = right;
        this.value = value;
    }
}
```

Listing 2: BinaryTreeOperations.java

```java
public class BinaryTreeOperations {
    /**
     * Level order dolaşma, ikili ağacın seviye seviye dolaşılmasını
     * gerektirir. Önce kök(1.) seviye, sonra 2. seviye soldan sağa
     * ve seviyeler artacak şekilde düğümler dolaşılır.
     *
     * @param <T> Generic tür
     * @param node Dolaşılacak ağacın kökü
     * @return Aralarında virgül olacak şekilde düğümler
     */
    public static<T> String levelOrder(BTNode<T> node) {
        return null;
    }

    /**
     * Ağaçlar preorder dolaşma kullanılarak yeniden inşa edilebilir.
     * Bu metot, preoder dolaşması verilen ağacı inşa etmenizi
     * istemektedir.
     * @param <T> Generic tür
     * @param preorder Ağacın preorder dolaşma dizisi
     * @return Oluşturulan ağacın kök düğümü
     */
    public static <T extends Comparable<T>> BTNode<T> bstFromPreorder(T[] preorder) {
        return null;
    }
}
```

3 Ödev Açıklamaları

- Ödevler `https://bilmoodle.pau.edu.tr` adresindeki arayüzlerden cevaplanacaktır. Ödevinizdeki kod dosyasını kişisel bilgisayarınıza indirip, ödevi çözüp tekrar sisteme geri yükleyebilirsiniz fakat sınıf, dosya ve paket yapısının korunduğundan emin olmalısınız.
- Ödevler bireysel olarak cevaplanmalıdır, grup çalışması yapmak yasaktır
- Ödevleri yapay zeka sohbet robotlarına cevaplatmak yasaktır

- Ödevlerde kopya kontrolü yapılacaktır ve benzerlik oranı belirli bir yüzdenin üzerinde olan ödevler kopya olarak değerlendirilecektir. Değişken isimlerini değiştirmek, kodların sırasını değiştirmek anlaşılabilen yöntemlerdir, lütfen bu yola başvurmayınız.
- Ödevin son teslim tarihi **13.12.2023 saat 23:59**'dur. Belirtilen saatte sistem kapanacaktır, lütfen son ana kadar beklemeden ödevinizi gönderin.