Veri Yapıları Laboratuvar Föyü
Hafta 12
Çizge için dfs(derinine arama) ve bfs(enine arama) Metodlarının Yazımı

Şevket Umut ÇAKIR

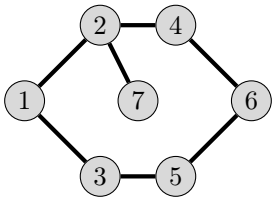
1 Giriş

Bu deneyde çizge içinde arama yapan dfs(derinine arama) ve bfs(enine arama) metodlarının yazılması amaçlanmaktadır.

2 Konu Anlatımı ve Deney Hazırlığı

2.1 Depth First Search

Çizge içinde dolaşmak için kullanılan bir yöntemdir. Dolaşma işlemi bir düğümden başlar. Gidilebilecek mümkün komşulardan bir tanesini seçer ve ilerler. Gidilebilecek diğer düğümlere bakmadan önce yeni gidilen düğümün komşuları incelenir ve bir tanesine gidilir. Gidilebilecek komşu kalmayınca geri sarılarak mümkün komşular aranır. Tek biçimlikik sağlamak için mümkün komşulardan alfabetik olarak küçük olan önce seçilir. Aşağıda DFS örneği ve algoritması verilmiştir.



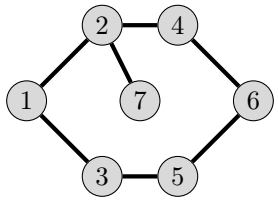
Şekil 1: DFS Örneği:1 düğümünden başlanınca sıralama 1, 2, 4, 6, 5, 3, 7 olacaktır

```
S=yeni yığıt(stack)
visited={} //boş küme
S.push(başlangıç düğümü)
while S boş olmadığı sürece do
    mevcut=S.pop()
    if mevcut, visited kümesi içindeyse then
        continue
    end
    visited.add(mevcut)
    foreach (mevcut,v) kenarına sahip v düğümleri do
        S.push(v)
    end
end
```

Algorithm 1: DFS Algoritması

2.2 Breadth First Search

Dolaşma işlemi bir düğümden başlar. Gidilebilecek mümkün komşularhepsini sırayla seçer ve ilerler. Gidilebilecek bütün komşular gezildikten sonra ilk komşunun komşuları seçilir. Tek biçimlikik sağlamak için mümkün komşulardan alfabetik olarak küçük olan önce seçilir. Aşağıda BFS örneği ve algoritması aşağıda verilmiştir.



Şekil 2: BFS Örneği:1 düğümünden başlanınca sıralama 1, 2, 3, 4, 7, 5, 6 olacaktır

```
Q=yeni kuyruk(queue)
visited={} //boş küme
Q.enqueue(başlangıç düğümü)
while Q boş olmadığı sürece do
    mevcut=Q.dequeue()
    if mevcut, visited kümesi içindeyse then
        continue
    end
    visited.add(mevcut)
    foreach (mevcut,v) kenarına sahip v düğümleri do
        Q.enqueue(v)
    end
end
```

Algorithm 2: DFS Algoritması

3 Deneyin Uygulanması

3.1 dfs Metodunun Yazımı

dfs metodu parametre olarak başlangıç düğümünün değerini alır ve geriye dolaşılan düğümlerin sırasını liste olarak döndürür. Yığıt işlemlerini gerçekleştirmek için `java.util.Stack` sınıfı, küme işlemlerini gerçekleştirmek için `java.util.HashSet` veya `java.util.ArrayList` sınıfları kullanılabilir. Yığıt işlemlerini gerçekleştiren bir sınıf yazmanıza gerek yoktur.

3.2 bfs Metodunun Yazımı

bfs metodu parametre olarak başlangıç düğümünün değerini alır ve geriye dolaşılan düğümlerin sırasını liste olarak döndürür. Kuyruk işlemlerini gerçekleştirmek için `java.util.ArrayDeque` sınıfı kullanılabilir. Kuyruk işlemlerini gerçekleştiren bir sınıf yazmanıza gerek yoktur.