2021-2022 Güz Yarıyılı

Algoritma Analizi

Ödev – 3: Graflar

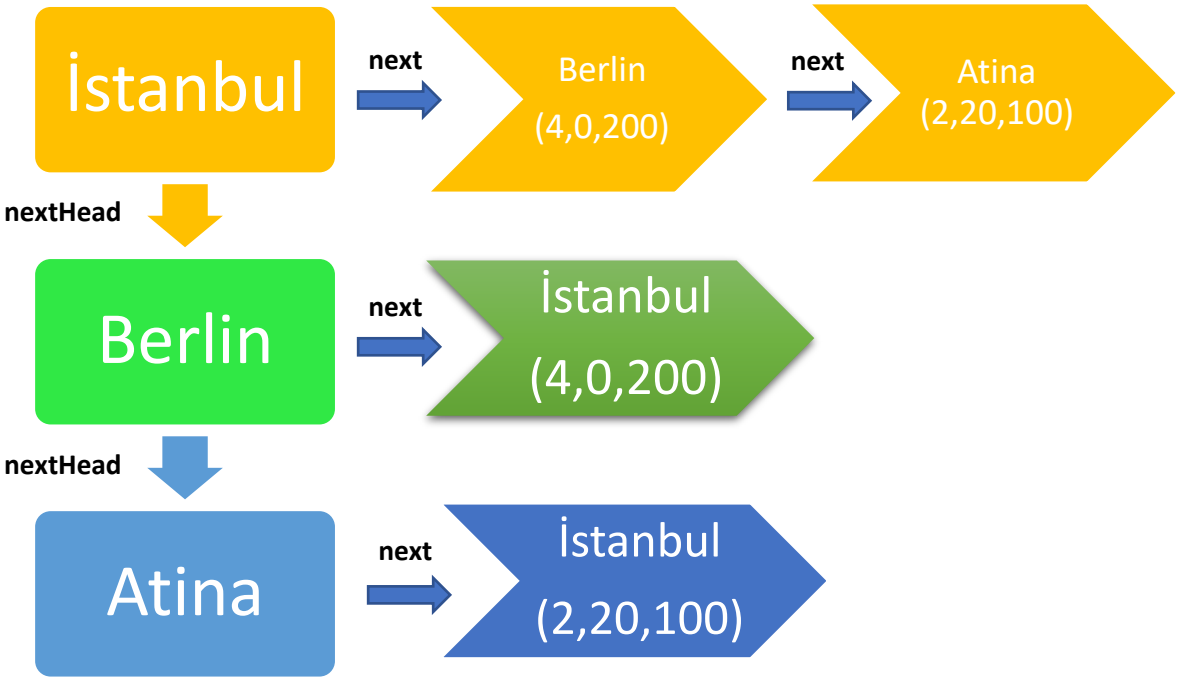
Rayene Bech

18011115

I. Yöntem

Bu problem, Depth First Search (DFS) yaklaşımı ile çözülmektedir.

- Graf için komşuluk listesi kullanıldı:



- Uçuş bilgileri tutmak için struct yapısı kullanılmaktadır. Her şehrin bir id'si var:

```
struct node {
 char city[MAX_LENGTH];
 int id;
 int hour;
 int min;
 int price;
 struct node* nextHead;
 struct node* next;
};
```

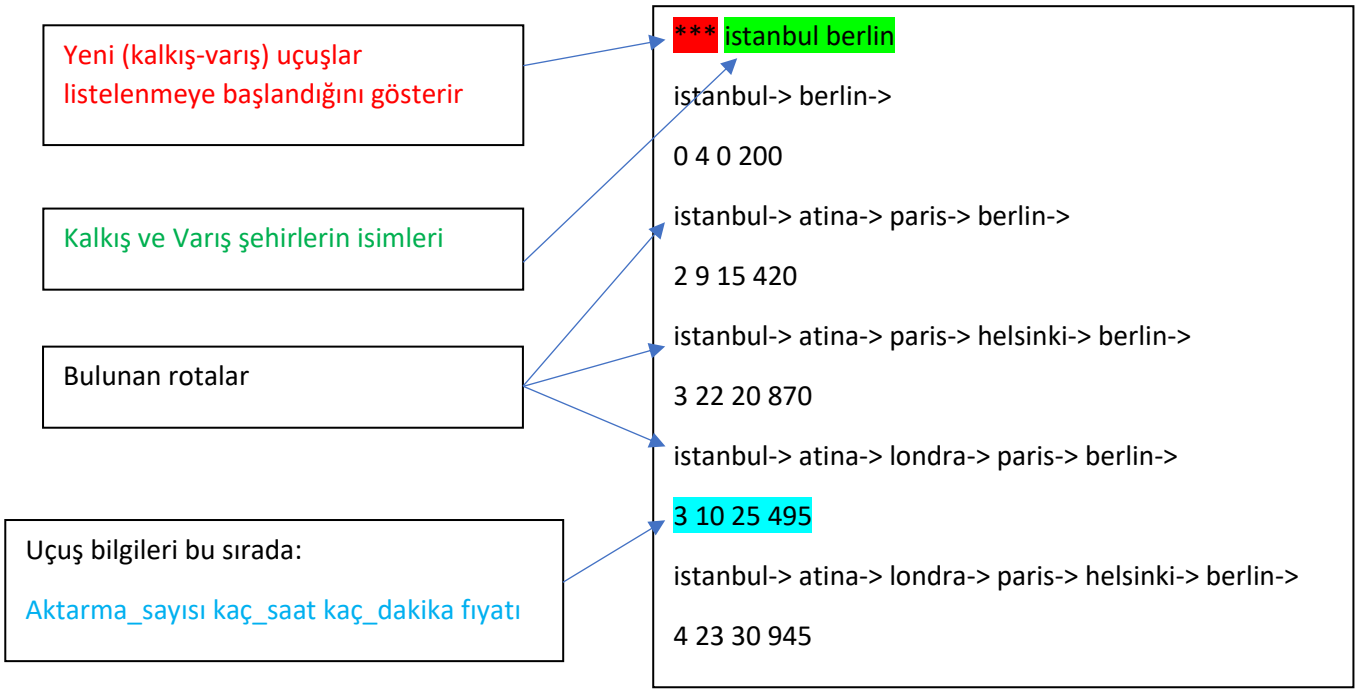
Bu problem aşağıdaki adımlar ile çözülmektedir:

- İlk başta, dosyadan şehirler okunur. "**Cities**" dizisinde unique değerleri kaydedilir. Bunun için bu iki fonksiyon çağırılır:

void mappingFile (char * filename, char cities[MAX_CITY][MAX_LENGTH] , int* size);

int addIfUnique (int n, char cities[n][MAX_LENGTH], char city[MAX_LENGTH]);

- Ondan sonra, yukarıda gösterilen şekil gibi bir graf oluşturulur. Yine dosyadan okuyarak, uçuş bilgileri eklenmektedir.
- Sonra, kullanıcıdan kalkış ve varış şehirlerin isimlerin girmesini istenmektedir. Grafta HeadListesinde bu şehirler bulunmuyorsa, demek ki uçuş yok. Bu durum kontrol edilir ve uyarı gösterir.
- Ondan sonra kullanıcıdan maksimum kaç aktarma ve sıralama neye göre yapılacağı ile ilgili bilgiler okunmaktadır.
- Bu şehirler arasında daha önce arama yapılmışsa, sonuçlar **"cache.txt"** dosyadan çekilir. **printresult()** fonksiyonu çağırarak, kullanıcının girdiği kriterlere göre uygun uçuşlar sırayla gösterilmektedir. **"cache.txt"** dosyası belli bir format üzerinde oluşturuldu. Mesala: İstanbul- Berlin arasındaki uçuşlar :



- Eğer dosyada uçuş bilgisi bulunmuyorsa, DFS fonksiyonu çağırarak yeni bir arama yapılır.
- **DFS** fonksiyonu, bütün rotalar bulmak istediğimiz için biraz farklı bir şekilde çalışıyor. Kalkış şehirden yola çıkarak, gezilen komşular **path[ ]** dizisinde adresleri saklanır. Ve Yine her gezilen şehir "**visited[]"** dizisinde 1 olur. Eğer varış şehre ulaştıysak, **visited[]** dizisi bu şehir için yine sıfırlanır ve **path[]** dizisinden rota bilgileri dosyaya saklanır aynı zamanda **flights[]** dizisinde saklanır.
- DFS bittiğinde Kalkış ve Varış şehirler arasında bütün rotalar bulmuş oluruz. **printresult()** fonksiyonu çağırarak, kullanıcının girdiği kriterlere göre uygun uçuşlar sırayla gösterilmektedir.

Zaman Karmaşıklığı:

Graf işlemleri : Grafta n düğüm ve E kenar varsa:

- Graf oluşturmak için: komşuluk listesi kullandığımızdan, Headlistesi oluşturması (**createHead** fonksiyonu) **O(n)** sürecektir. Ondan sonra her kenar eklemek için **InsertNode O(1)** ve **searchNode O(n)** çağırılmaktadır. Toplam olarak O(n) sürecektir.
- DFS için: Her düğüm gezildiği için O(n) sürecek. Ayrıca her düğüm için bütün kenarlara bakıldığından ayrıca O(e) sürecektir. Demek ki toplam olarak **O(n+e)** sürecektir.

Sıralamada Merge Sort kullanıldığı için zaman karmaşıklığı **O(n* Log n).**

Yer Karmaşıklığı:

Graf işlemleri : Grafta n düğüm ve E kenar varsa:
n tane düğüm, her düğüm için iki kenar (yönsüz bir graf) toplam O(n+2e) = O (n+e) yer kapsamaktadır.
Visited, ve path dizileri O(n) yer kapsamaktadır.
Sıralamada Yer karmaşıklığı **O(n),** n burada uçuş sayısı (rotalar)

II. Uygulama

- İlk başta, verilen “Sample.txt” dosyasından veriler “**cities[]”** dizisine okunup bir graf oluşmaktadır. Oluşan graf :

```
Printing the graph:
istanbul -> atina id= 2 (2,20,120) -> berlin id= 1 (4,0,200)
|
berlin -> paris id= 4 (1,15,100) -> helsinki id= 3 (6,10,250) -> istanbul id= 0 (4,0,200)
|
atina -> londra id= 5 (3,10,175) -> paris id= 4 (3,40,200) -> istanbul id= 0 (2,20,120)
|
helsinki -> paris id= 4 (7,10,300) -> berlin id= 1 (6,10,250)
|
paris -> londra id= 5 (0,40,100) -> atina id= 2 (3,40,200) -> helsinki id= 3 (7,10,300) -> berlin id= 1 (1,15,100)
|
londra -> atina id= 2 (3,10,175) -> paris id= 4 (0,40,100)
|
```

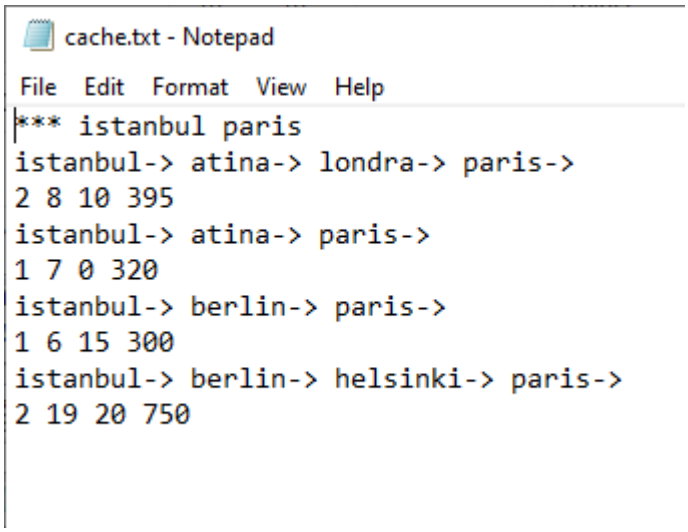
- Örnek 1: İstanbul- Paris arasında 4 aktarmaya kadar uçuşlar artan fıyata göre göstermek istendiğinde:

```
********** Welcome to the Flight Management System ****************

Please enter the source: istanbul

Please Enter the destination: paris

Please Enter maximum number of stops: 4

To sort results by price enter 0, to sort them by duration enter 1: 0
```

Daha önce bu arama yapılmadığı için DFS çağırılır ve sonuçlar bu şekilde listelenir:

```
 ***************** DFS start ***************

Flight no: 1
istanbul->berlin->paris->
number of stops: 1 . Total duration: 6 hour 15 min
Total cost: 300

Flight no: 2
istanbul->atina->paris->
number of stops: 1 . Total duration: 7 hour 0 min
Total cost: 320

Flight no: 3
istanbul->atina->londra->paris->
number of stops: 2 . Total duration: 8 hour 10 min
Total cost: 395

Flight no: 4
istanbul->berlin->helsinki->paris->
number of stops: 2 . Total duration: 19 hour 20 min
Total cost: 750
```

“cache.txt” dosyası oluşturuldu ve sonuçlar oraya da eklendi:



- Örnek 2: İstanbul Paris arasında sadece 1 aktarmalı olan uçuşlar artan sürelere göre göstermek istendiğinde:

```
********** Welcome to the Flight Management System ***************

Please enter the source: istanbul

Please Enter the destination: paris

Please Enter maximum number of stops: 1

To sort results by price enter 0, to sort them by duration enter 1: 1
```

Bu arama daha önce yapıldığı için sonuçlar tekrar DFS çağırmadan direkt "cache.txt" dosyasından çekilir ve filtrelenir:

```
Flights retrieved from the cache:

Flight no: 1
istanbul-> berlin-> paris->

number of stops: 1 . Total duration: 6 hour 15 min
Total cost: 300

Flight no: 2
istanbul-> atina-> paris->

number of stops: 1 . Total duration: 7 hour 0 min
Total cost: 320
```

- Örnek 3: İstanbul Paris arasında aktarmasız olan uçuşlar artan sürelere göre göstermek istendiğinde:

```
********** Welcome to the Flight Management System ***************

Please enter the source: istanbul

Please Enter the destination: paris

Please Enter maximum number of stops: 0

To sort results by price enter 0, to sort them by duration enter 1: 0
```

Daha önce bu arama yapıldığı için "cache.txt" dosyasından sonuçlar çekilir. Ama belirtilen filtrelere göre uçuş bulunmadığı için bu mesaj verilmektedir:

```
Flights retrieved from the cache:

Unfortunately, there are no direct flights. Try searching for indirect flights
```

Örnek 4: İstanbul – Cezayir arasındaki uçuşlar artan sürelere göre göstermek istendiğinde, Cezayir'e giden uçuşlar bulunmadığı için böyle bir mesaj verilmektedir:

```
********** Welcome to the Flight Management System ***************

Please enter the source: istanbul

Please Enter the destination: cezayir

Unfortunately we do not have any flights to the city cezayir
```