**HAVUZ PROBLEMİ**

Muhammed Kurfeyiz-Erdem Özer

Bilgisayar Mühendisliği Bölümü,Kocaeli Üniversitesi

Kocaeli,Türkiye

[m.kurfeyiz98@gmail.com](mailto:m.kurfeyiz98@gmail.com) / [erdemcemozer@hotmail.com](mailto:erdemcemozer@hotmail.com)

**1.1. PROBLEM TANIMI**

Bu projenin amacı algoritma çözümleme ile belirtilen havuz probleminde çözüme ulaşmak.

**1.2. YAPILAN ARAŞTIRMALAR**

Projenin yapım aşamasında öncelikle kullanacağımız dili ve GUI tasarımı için kullanacağımız platformu seçtik. Daha sonra kullanmamız gereken iki farklı algoritma olan Max Flow ve Min Cut ile ilgili araştırmalar yaptık ve kodların nasıl uyarlandığını inceledik.

**2. TASARIM**

Uygulamamızda öncelik olarak açılış sayfasını ve kullanıcıdan istenilenleri o sayfada almayı düzenledik. Bu kısmı Intellij IDEA programı ve SceneBuilder eklentisi ile yaptık. Fxml dosyalarında aldığımız değişkenleri actionEvent ile Controller sayfamıza yolladık.

Controller’a yolladığımız matrisi graph değişkenine kaydettik. Bu sayede kullanıcının girdiği matris şeklinden hangi node’lar (musluklar) arasında yol olduğunu ve ne kadar kapasiteye sahip olduklarını almış olduk. Giriş çıkış musluklarını ve toplam musluk sayısını da değişkenlerde tuttuktan sonra GUI üzerinden ikinci sayfaya geçme tuşuna basınca bu tutulan değerleri MaxFlow MinCut algoritmamıza yollayarak önce MaxFlow değerini buluyoruz daha sonra da MinCut için kesilmesi gereken yolları gösteriyoruz.

**2.1 Controller**

Burada kullandığımız fonksiyonlar sayesinde string olarak yollanan matrisi satır ve sütunlarına göre ayırdık ve bu sayede iki boyutlu bir diziye aktardık. Bu dizi bizim hangi musluktan diğerine yol olduğunu, yolların kapasitelerini barındıran dataların olduğu matrisi içeriyor. Bu sayede kodun içerisinde graph’ı oluşturyoruz.

graphPage fonksiyonunda ise bu sayfada kullandığımız dataları diğer sayfaya aktararak kullanıcı butona bastıktan sonra yeni sayfaya geçmesini sağlıyoruz.

**2.2 GraphController**

Max Flow – Min Cut algoritmamız Controller sayfasından aldığımız dataları burada kullanıyor. setGraph fonksiyonunda değişkenleri algoritmanın bulunduğu class’lara yollayarak çalıştırıyoruz. MaxFlowMinCut class’ı içinde öncelikle residualGraph’ı oluşturuyoruz.

Daha sonra MaxFlowMinCut fonksiyonuda BreadthFirstSearch (bfs) algoritması ile arama yapıyoruz. Giriş çıkışları da yolladığımız bu fonksiyonda bfs ile dolaşılan yollardaki maximum kapasiteler bulunuyor ve kullanılan kapasitelere göre kalan yolların kapasiteleri hesaplanıyor. Daha sonra da bu kapasiteler toplanarak Maximum akış değeri bulunuyor.

MinCut için de yine bfs algoritmasına giriş node’u ve sırayla her node yollanıyor daha sonra yine residual graph üstünde tarayarak suyu tamamen kesmek için minimum kesilmesi gereken yolları buluyor.

Daha sonra bulunan bu Max Flow ve Min Cut değerlerini bir sonraki sayfada kullanıcıya aktarıyoruz.

**2.3 Graph Sayfası**

Son sayfamız olan graph sayfasında ise kullanıcıya bulunan Max Flow ve Min cut değerleri ile girdiği datanın matris hali gösteriliyor.

**2.4 Algoritma Kaba Kodu**

Verileri karşıla  
Residual Graph oluştur  
BFS ile yolları ara  
Bulunan max flow kapasitelerini topla  
Çıkışa gelene kadar 3. Ve 4. adımı tekrarla  
Min Cut için tekrar BFS ile taramaya başla  
Ulaşılan ve ulaşılamayan node’ları ArrayList ile tut  
Graph üzerinde bu nodelara bak  
Eğer bulunduysa Pair class’ı özelliklerine göre sakla  
Kullanıcıya ikinci ekranı göster  
Max flow’u ve Min cut’ı ekrana matris ile birlikte bastır

**3. YAZILIM MİMARİSİ**

Projemizi Intellij IDEA uygulaması üzerinde gerçekleştirdik. Projenin içindeki dosyalardan olan :

* Main.java
* Controller.java
* GraphController.java

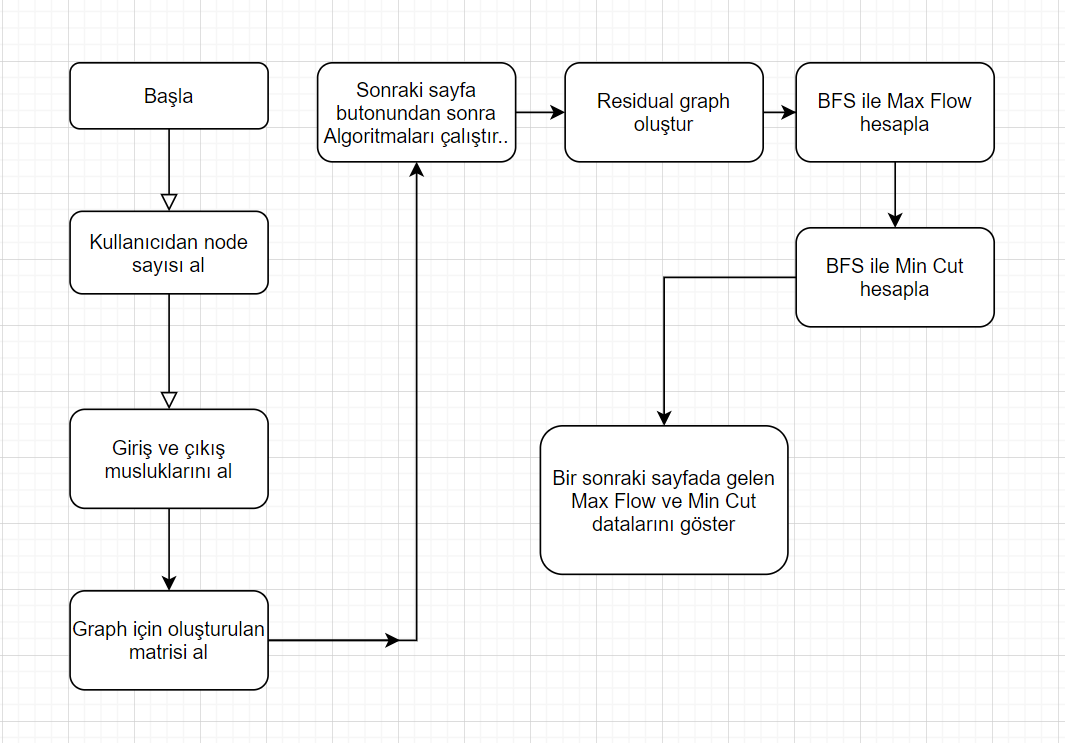
dosyaları projenin aslında çalıştığı kısımlar. Kullanıcıdan alınan verilerin tanımlandığı, tekrar başka değişkenlere aktarıldığı, algoritmalar ile üzerlerinde işlem yapıldığı kısımlar bu dosyalar.

Geriye kalan 2;

* sample.fxml
* graphPage.fxml

dosyaları ise uygulamamızın ön yüzü olan GUI kısmı. Kullanıcıdan alınan değerler için textArea’lar button’lar bu sayfalar üzerinden ayarlanıp uygulandı. GUI için Javafx platformunu ve tasarımda yardımcı olması için SceneBuilder eklentisini kullandık.

**4. VERİ AKIŞ ŞEMASI**

****

**6. GENEL YAPI**

Projemizde GUI üzerinden kullanıcıdan node sayısı, giriş ve çıkış musluklarını ve kullanacağı graph için matris alma kısmı; daha sonra da alınan bu dataları arka kısımda işleyerek Max Flow ve Min Cut değerlerini algoritmalarla bulması sorunsuz çalışmaktadır. Kullanıcı graph için isterlerini tek tek girmek yerine bir seferde o graph için matris oluşturarak daha hızlı bir şekilde uygulamayı kullanabilir.

**7. KAZANIMLAR**

Bu projenin yapımı sırasında algoritma çözümleme kullanarak istediğimiz en fazla akışı nasıl sağlayabileceğimi ve akış durdurmak için kesilmesi gereken en az kapasiteli boruları bulduk. GUI için kullandıklarımız tekrardan bir hatırlatma oldu. Bu sayede bir masaüstü programı oluşturmanın temel taşlarını hatırladık.

**8. REFERANSLAR**

* <https://stackoverflow.com/>
* <https://www.youtube.com/watch?v=oHy3ddI9X3o>
* <https://www.youtube.com/watch?v=Tl90tNtKvxs&t=190s>
* <https://www.youtube.com/watch?v=I76yiYlnKxQ&t=13s>
* <https://www.youtube.com/watch?v=zSeqO7Eno30>
* <https://en.wikipedia.org/wiki/Max-flow_min-cut_theorem>
* <https://www.cs.princeton.edu/courses/archive/spr04/cos226/lectures/maxflow.4up.pdf>
* <https://www.geeksforgeeks.org/minimum-cut-in-a-directed-graph/>
* <https://algs4.cs.princeton.edu/64maxflow/FordFulkerson.java.html>

**9. FOTOĞRAFLAR**

