ALGORİTMA VE UYGULAMA YÖNTEMLERİ

Muhammet Sait ÇELİK 170202025
Ozan AYDOĞAN 160202039

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

KOCAELİ ÜNİVERSİTESİ

170202025@kocaeli.edu.tr 160202039@kocaeli.edu.tr

Özet

Bu projede, literatürde azami akış(maximum flow) olarak geçen ve düğümler (nodes) arasında akış kapasiteleri belirli bir şekildeki (graph) bir başlangıçtan bir hedefe en fazla akışın sağlandığı problemleri çözmeniz beklenmektedir. Devamında ise akışın sistemden geçmemesi için literatürde min-cut olarak geçen yöntemi uygulamamız beklenmektedir.

1.Giriş

Proje de istenilen şekilde kullanıcıdan girilen verilerle graphların oluşturulması ve değerlerin graphlar üzerinde gösterilmesi istenmiştir. bunların yanında verilen azami akış(maximum flow) ve literatürde min-cut olarak geçen yöntemi kullanarak yine bunları graphta göstermemiz istenmiştir. Ayrıca masaüstü uygulaması olarak tasarlanması beklenilmektedir.

2.Yöntem

Öncelikle uygulamamızı eclipse üzerinde java olarak kodladık ve javanın kütüphanelerinden olan JPanel ve Jframe üzerinde çalıştırdık. Jpanel olarak 6 panel kullanarak yaptık. Matrisleri düğüm sayılarını başlangıc bitiş noktalarını JPaneller üzerinden aldık ve graphlarımızı da aynı şekilde Jpanel üzerinde çizdirdik kodda seçim yapılarak azami akış(maximum flow) ve literatürde min-cut olarak geçen yöntemi kullanım için seçmelerini sağladık. Toplam da 9 adet java classında kodumuz çalışır hale geldi.

3.Deneysel Sonuçlar

Projemizde yapılan graph şekillendirmeleri için öncelikle graph çizimi kütüphaneleri aradık ve bunları denedik. Ama en mantıklısı ve hızlısı javanın içinde hazır gelen bir kütüphane olan Graphics2D kütüphanesi mantıklı geldi ve bunu kullanmaya karar verdik hem stabil çalışması hemde javanın içinde hazır gelmesiyle bizim kaynak bulunması kolay olacağını düşündük.

Bunun yanında internetten bulduğumuz ve kaynakçada belirtilen kodların işleyişi ve projeye nasıl dahil edeceğimize dair çalışmalar yaptık ve başarılı olduk.

4.Sonuçlar

1-) Sonuç olarak azami akış(maximum flow) ve literatürde min-cut olarak geçen yöntemi öğrenmiş olduk ve bir algoritmanın nasıl projelerimiz de kullanabiliriz diye deneyim sahibi olmamızı sağladı.

2-) Graph çizdirmek ve yapımı için standart kütüphanelerin daha mantıklı olması

3-)JPanel ve JFrame i başarıyla kullanmayı başardık.

5.Kaynakça

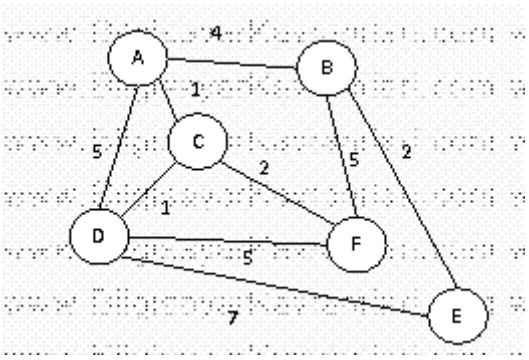
1-)
https://coderanch.com/t/340443/java/Draw-arrow-head-line
2-)
https://www.youtube.com/watch?v=M-F7z1xWS6o
3-)
https://www.geeksforgeeks.org/minimum-cut-in-a-directed-graph/

4-)
https://www.youtube.com/watch?v=0H9GLWajgTE
5-)
https://brilliant.org/wiki/max-flow-min-cut-algorithm/
6-)
https://www.youtube.com/watch?v=oHy3ddI9X3o
7-)
https://www.youtube.com/watch?v=0H9GLWajgTE
8-)
http://bilgisayarkavramlari.sadievrenseker.com/2010/05/22/ford-fulkerson-algoritmasi/
9-)
http://bilgisayarkavramlari.sadievrenseker.com/2009/03/08/graflarda-kesitler-cut-in-graphs-agaclarda-kesitler/
10-)
http://bilgisayarkavramlari.sadievrenseker.com/2010/05/22/edmonds-karp-algoritmasi/

FORD FULKERSON ALGORİTMASI

Bu algoritmanın amacı, literatürde azami akış (maximum flow) olarak geçen ve düğümler (nodes) arasında akış kapasiteleri belirli bir şekildeki (graph) bir başlangıçtan bir hedefe en fazla akışın sağlandığı problemleri çözmektir.

Azami akış (maximum flow) problemini örneğin şehirler arasında bağlı boru hattına veya tedarik zincirine benzetebiliriz. Örneğin aşağıdaki şekli ele alalım:

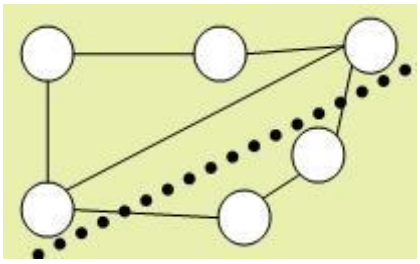


Buradaki düğümler, şehirleri ve düğümler arasındaki kenarlar (edges) ise şehirler arasındaki boru hatlarının kapasitesini belirtsin. Amacımız A düğümünden E düğüme azami miktarda akış sağlayabilmek olsun.

Ford-Fulkerson çözüm için yukarıdaki şekilde öncelikle hedef düğüme giden yolu bulur. Algoritma bu arama işlemi sırasında şayet derin öncelikli arama (depth first search ,DFS) kullanıyorsa ford fulkerson olarak isimlendirilir. Şayet aynı algoritma bu arama işlemi sırasında sığ öncelikli arama (breadth first search, BFS) kullanırsa bu durumda da edmonds karp algoritması olarak isimlendirilir. M kenar sayısı ve f maksimum akış değeri olmak üzere Big O (M*f).

MİNİMUM CUT

Bir grafı, iki parçaya bölen kesitin, en az kenarı kesmesi durumudur. Örneğin aşağıdaki grafta en az kesilebilen kenardan kesilmiştir. Bu kesilme dışındaki bütün kesitler bu kesitten daha yüksek veya aynı boyuttadır:

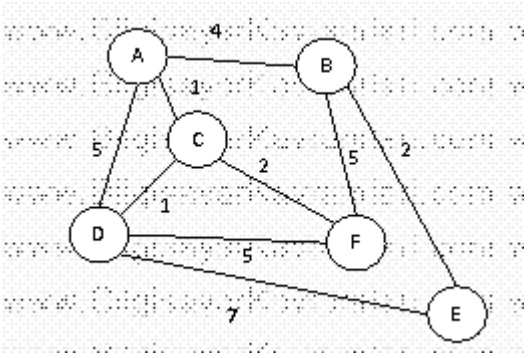


Yukarıdaki kesitin boyutu 2'dir ve yukarıdaki graf için boyutu 1 olan başka bir kesit bulunamaz.

Edmonds Karp Algoritması

Bu algoritmanın amacı, literatürde azami akış (maximum flow) olarak geçen ve düğümler (nodes) arasında akış kapasiteleri belirli bir şekildeki (graph) bir başlangıçtan bir hedefe en fazla akışın sağlandığı problemleri çözmektir.

Azami akış (maximum flow) problemini örneğin şehirler arasında bağlı boru hattına veya tedarik zincirine benzetebiliriz. Örneğin aşağıdaki şekli ele alalım:



Buradaki düğümler, şehirleri ve düğümler arasındaki kenarlar (edges) ise şehirler arasındaki boru hatlarının kapasitesini belirtsin. Amacımız A düğümünden E düğüme azami miktarda akış sağlayabilmek olsun.

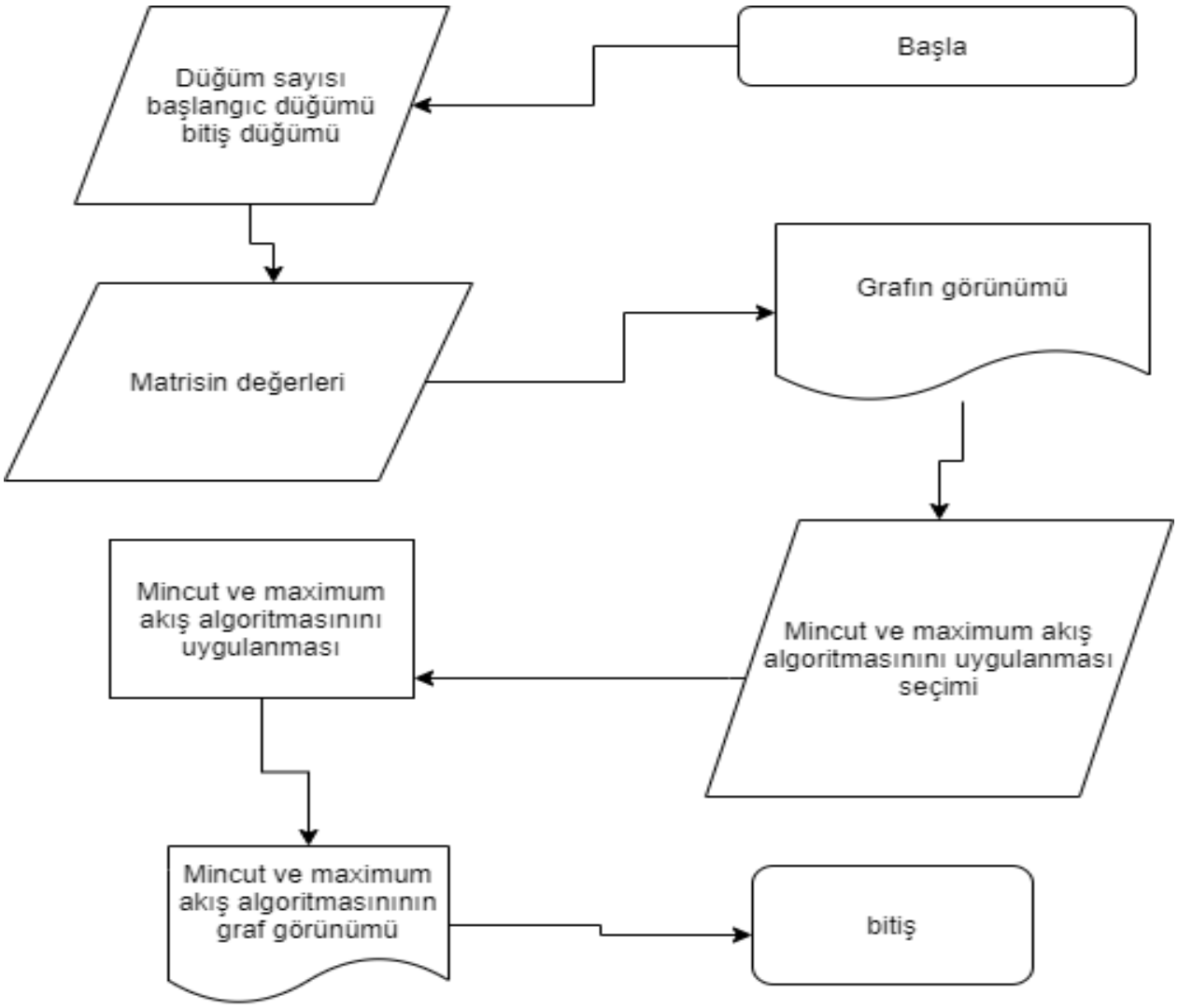
Ford-Fulkerson çözüm için yukarıdaki şekilde öncelikle hedef düğüme giden yolu

bulur. Algoritma bu arama işlemi sırasında şayet derin öncelikli arama (depth first search ,DFS) kullanıyorsa ford fulkerson olarak isimlendirilir. Şayet aynı algoritma bu arama işlemi sırasında **sığ öncelikli arama (breadth first search, BFS)** kullanırsa bu durumda da edmonds karp algoritması olarak isimlendirilir.Big O (V E)

Kaba Kod

Max-Flow ve Min-Cut algoritmaları analiz edilerek koda dahil edildi. İçerisinde BFS DFS algoritmalarını kullanarak giriş düğümünden (vana) çıkış düğümüne (musluk) olan tüm yollar tespit edilip kayıt edildi. Proje için arayüz hazırlanması yapıldı . İlk olarak kullanıcı programı çalıştırdığında karşısına JLabel JTextfield ve JButton sınıflarından oluşmuş nesneleri ,aynı zamanda JPanel sınıfından subclass olan Panel_1 class'ından türeyen nesnenin oluşturduğu bir arayüz karşılayacaktır. Kullanıcı Düğüm sayısı , Giriş vanası , Çıkış musluğu değerlerini girdikten sonra , Panel_2 sınıfından türeyen nesne ile karşılaşacaktır. Burada istenen, Düğümlerimizin birbiri ile bağlantılı olabilmesi için gerekli olan 'Edge' değerlerini her bir düğüm için yazmamız. Düğüm kenar değerlerini JTextfield sınıfından türeyen nesne tutup, sayı değerini alabilmemiz için (çünkü girilecek değeri sistem String olarak algılayacaktır.) Integer.tostring( ) metodunu kullanarak string değerini integer değere çevirip kaydederiz . Bu aldığımız değerler aslında bizim Akış kapasitesi değerleridir. JButton'dan türeyen nesneyi kullanarak bir sonraki panel olan Panel_3'den türeyen nesne ile karşılaşırız, burada karşımıza, aldığımız matris değerleriyle oluşturduğumuz Graf yapısı karşılayacaktır. Bu graf yapısında düğümler, kenarlar, kenarların yönü, kenarların taşıdığı değerler (akış kapasitesi) değerleri yazdırılacaktır. 2 tane buton karşılayacaktır, Min-Cut işleminin yapılabilmesi için ve Max-Flow işleminin yapılabilmesi için ayrı bir buton vardır. Butonlardan herhangi birine basarak yaptırmak istediğimiz işlemi onaylarız. Karşımıza yeni bir panelde yaptırdığımız işlemin sonucunu gösteren görsel çıkacaktır, aynı zamanda Back butonuyla, bir önceki panele gitme imkanı sağlayan bir butonla karşılaşacağız.

Akış Şeması


Başla
Düğüm sayısı
başlangıc düğümü
bitiş düğümü
Matrisin değerleri
Grafın görünümü
Mincut ve maximum akış algoritmasınını uygulanması seçimi
Mincut ve maximum akış algoritmasınını uygulanması
Mincut ve maximum akış algoritmasınının graf görünümü
bitiş