**Yazılım Laboratuvarı 2**

**3. Proje Ödevi**

*Abdulcelil Kurt*

*170202001*

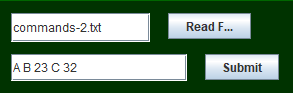
*celilkurt82@gmail.com*

***Özet***

Bu projede, bir grafta belirlenen iki düğüm arasındaki en yüksek akışın hesaplanması ve akışın tamamıyla kesilmesi için gereken en az maliyetli kesimin hesaplaması istenmektedir. Bu problemelere literatürde max flow, min cut denmektedir.

Projede Java ve Netbeans IDE kullanılmıştır.

**Kullanım**

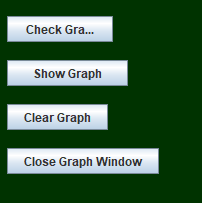
Grafın düğüm ve bağlantı bilgileri, GUI’deki form yardımıyla veya bir text dosyasından, formata bağlı kalınarak girilebilir.

Üstteki form belirtilen dosyadan bir veya birçok komutun birlikte işletilebilmesini sağlar.

Alttaki ise her seferde tek bir düğümün ilişkileri belirtilecek şekilde çalışır. Örnek satır şu anlama gelir: A’nın B ile 23, C ile 32 kapasiteli bağlantısı var.

Kaynak düğümü belirtmek için satır başına #, hedef düğümü belirtmek için satır başına @ koyulması gerekir.

Not: Eğer satır başında ‘@’ veya ‘#’ varsa bir sonraki kelimeye veya karaktere bakılır, sonrasında ne yazıldığının bir önemi yoktur.



-Graf doğru bir şekilde okutulduğunu kontrol etmek için Check Graph butonuna basmak yeterli ama bu gerekli bir işlem değil.

-Grafı çizdirmek için ‘Show Graph’ butonuna basmak gerekir. Eğer graf tutarlıysa ‘sourceNode’ dan başlayarak graf çizelecek. ‘sourceNode’ ile herhangi bağlantısı olmayan düğümler çizilmezler.

-Yeni bir graf tanımlamak için veya yanlışlıkla graf tutarsız tanımlandıysa, bellekte tutulan graf bilgilerini sıfırlamak içi ‘Clear Graph’ butonuna basmak gerekir. Bu işlemin sonucunda grafın tutulduğu değişkenler sıfırlanacaktır.

-’Close Graph Window’ butonu grafın çizildiği pencere açıksa kapatır.



-‘Find Max Flow’ butonuna basıldığında önce grafın tutarlılık kontrolü yapılır. Graf tutarlıysa ‘sourceNode’ dan başlayarak hesaplama yapılır ve sonuç GUI’nin mesaj kısmında bildirilir.

-’Find Min Cut’ butonu ile akışın kesilmesi için gerekli olan minimum maliyeti hesaplayabilirsiniz.

**Sınıflar, Methodlar ve Değişkenler**

Kullanılan sınıflar, methodlar ve belli başlı değişkenler.

**Sınıflar**

**Coordinat:** Graf çizilirken bağlantılar bütün düğümler çizildikten sonra çizildiği için düğümlerin kordinatlarının hafızada tutulması gerekiyor. Düğümlerin ‘GraphGraphic’ penceresindeki konumunu tutar.

**Node:** Düğümleri oluşturulduğu nesnedir. ‘childList’ de bağlantıları tutulur. ‘capacities’ de bağlantılarına karşılık kapasiteleri tutulur. ‘capacitiesBackup’ da kapasitelerin yedeği tutulur çünkü max flow hesaplanırken ‘capacities’ de değişiklik yapılır ve gerek graf çizdirilirken gerekse min cut hesaplanırken asıl kapasiteye ihtiyaç olur. ‘label’ değişkeninde düğümün etiketi bulunur.

**GraphUtils:** Bu sınıfta bulunan bütün değişkenler ve methodlar static’tir.

-‘sourceNode’ kaynak düğümünün tutulduğu değişkendir.

-’targetNode’ hedef düğümünün tutulduğu değişkendir.

-’isGraph’ grafın tutarlı olup olmadığının tutulduğu değişkendir.

-’totalCapacity’ azami akış miktarının tutulduğu değişkendir.

-’totalCut’ minimum kesim maliyetinin tutulduğu değişkendir.

-’nodes’ etiketine karşılık düğümün tutulduğu değişkendir.

-’getOrCreateNode(label)’ ‘nodes’ da ‘label’ etiketine sahip bir düğüm varsa döndürür yoksa oluşturur, ‘nodes’ a ekler ve döndürür.

-’createGraph(str)’ str’de aracılığı ile iletilen komuta göre düğüm oluşturur, bağlantı kurar veya atama yapar.

-’clearGraph()’ grafı tanımlayan verilerin tutulduğu değişkenleri sıfırlar.

-’checkGraph()’ ‘sourceNode’ ve ‘targetNode’ tanımlı ve kaynaktan hedefe en az bir yol varsa true döndürür.

-’checkGraphHelper()’ kaynaktan hedefe yol varsa true döndürür.

-’findMaxCapacity(tempRoot,minCapacity)’ kaynaktan hedefe azami akışı hesaplar.

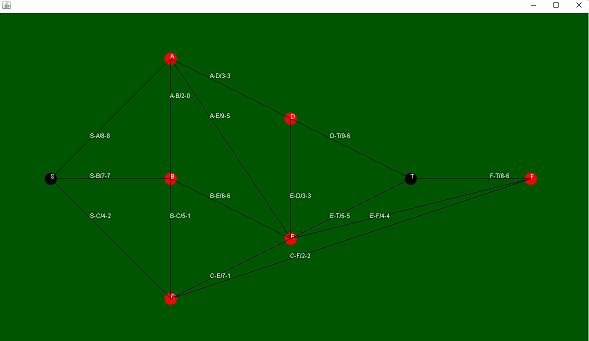
-’calMinCut()’ kaynaktan hedefe akışı kesebilmek için gereken minimum maliyetli işlemi veya işlemleri döndürür ve bağlantıların kapasitesini günceller.

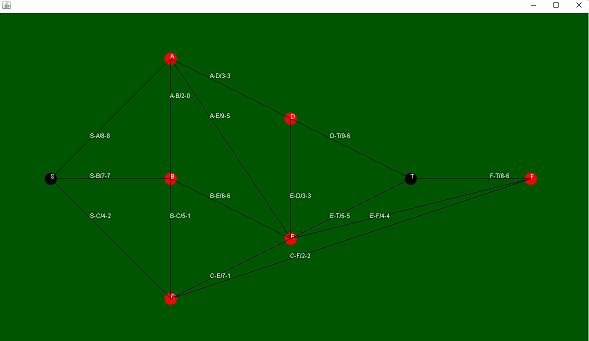
-’resetCapacities()’ herbir düğümün ‘capacities’ ini ‘capacitiesBackup’ a göre günceller.

**GUI:** Java Form sınıfıdır. Butonlar ve düğümü tanımlanmasında kullanılan formları barındırır.

**GraphGraphic:** ‘GraphUtils’ sınıfında tutulan static ‘nodes’ değişkeni ile grafı çizer. Düğümler arasındaki mesafe düğüm sayısı ile orantılı olacak biçimde artar çünkü düğümle orantılı olarak bağlantı sayısı artıyor ve bağlantı bilgilerinin birbirine karışmaması için aradaki mesafe düğüm sayısı ile doğru orantılı olmalıdır.

Graf çizilmeye kaynak düğümden başlar. Çizilen düğüm, düğüme karşılık çizildiği kordinat gelecek şekilde coorMap’a kaydedilir. Mevcut düğümün herbir bağlantısı eğer daha önce çizilmediyse çizilir ve coorMap’a kaydedilir . Dikey olarak pencereyi ortalayacak şekilde başlanır.





**Max Flow Algoritması**

Azami akış miktarının hesaplanması için probleme özgü bir BFS algoritması kullanıldı.

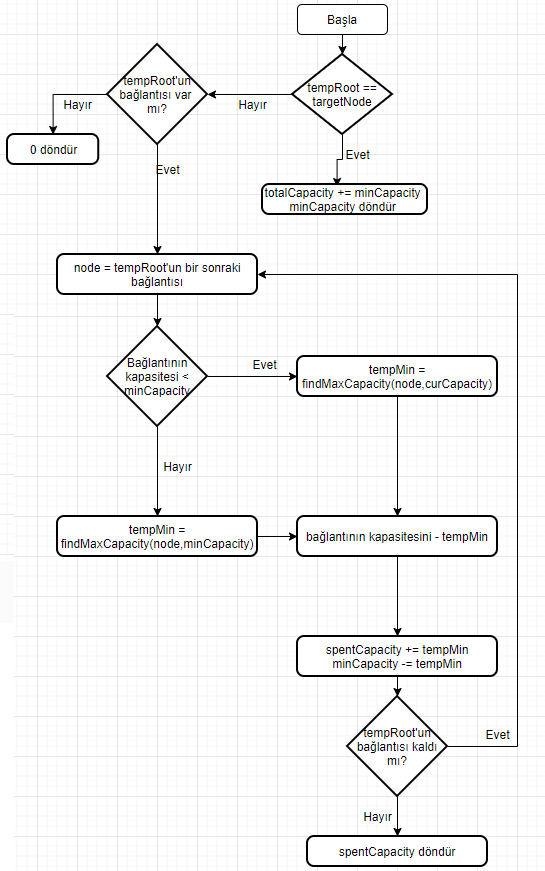
Örnek olarak A ‘sourceNode’ ve D ‘targetNode’ olsun. ABCD yolu için en düşük kapasiteli bağlantı B-C arasında ise A-B, B-C ve C-D bağlantılarının kapasiteleri B-C’nin kapasitesi kadar azaltılır ve totalCapacity değişkeni B-C’nin kapasitesi kadar arttırılır. Sonra diğer yol kombinasyonları için işlem tekrar edilir. İşlemler bittiğinde toplam akış miktarı hesaplanmış olur.

**Zaman karmaşıklık analizi:**

V: Düğüm sayısı

E: Bağlantı sayısı

Algoritma karmaşıklığı: O(E+V) dir.



**Kaynaklar**

Edmonds Karp algoritması:

<http://bilgisayarkavramlari.sadievrenseker.com/2010/05/22/edmonds-karp-algoritmasi/> ,

<https://www.youtube.com/watch?v=zSeqO7Eno30>

Max Flow hakkında genel bilgi:

<https://www.cs.princeton.edu/courses/archive/spr04/cos226/lectures/maxflow.4up.pdf>

Min Cut hakkında genel bilgi:

<http://www.mathcs.emory.edu/~cheung/Courses/323/Syllabus/NetFlow/max-flow-min-cut.html>

Max Flow-Min Cut genel bilgi:

<https://www.youtube.com/watch?v=VYZGlgzr_As>