MİNİMUM DÜĞÜM ÖRTME PROBLEMİNİN TABU ARAMA ALGORİTMASI İLE ÇÖZÜLMESİ

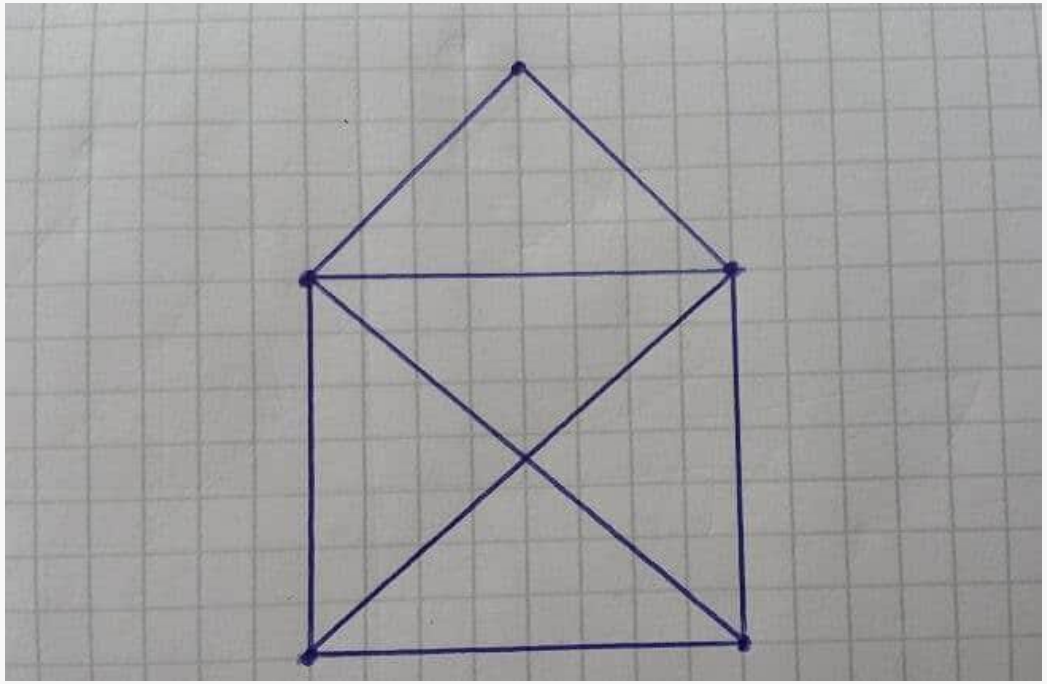
Zeki Optimizasyon Teknikleri

Gazi Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı
Kübra Nilgün Karaca
198333403

İçerik

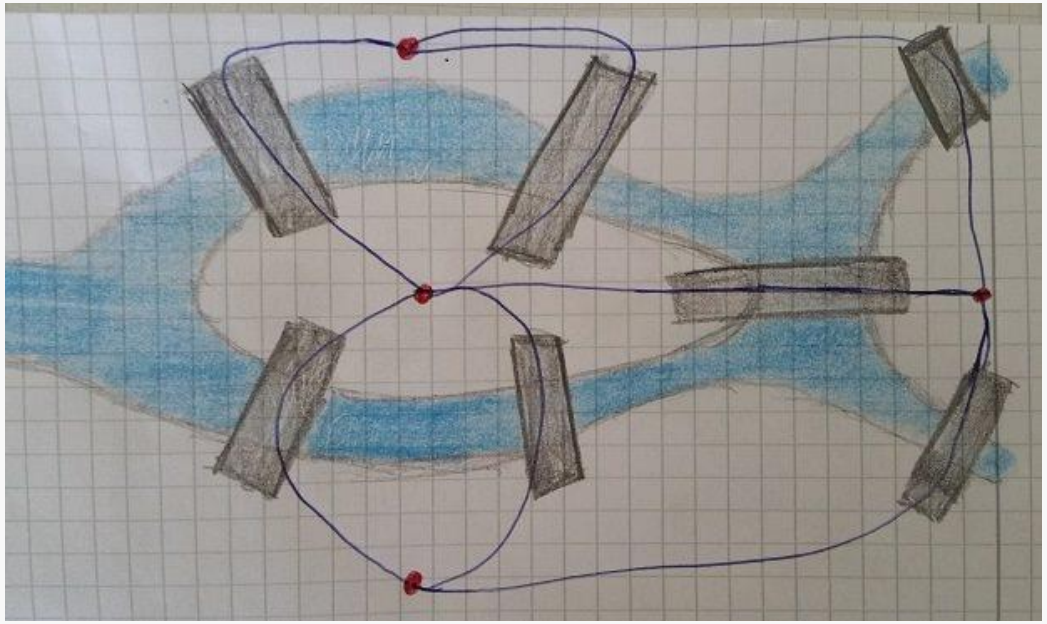
- Grafik Teorisi
- Minimum Düğüm Örtme Problemi
- Tabu Arama Algoritması
- Uygulama
- Kullanılan Veriler
- Uygulama Adımları
- Uygulama İşleyişi
- Sonuçlar
- Bulgular

Grafik Teorisi



- «Şekli elinizi kaldırmadan ve aynı çizgiden sadece bir kere geçerek çizebilir misiniz?» sorusu
- «Köningsberg'in 7 köprüsü» olarak bilinen matematik problemi

Grafik Teorisi



- 1735 yılında, 7 köprü ile birbirine bağlanmış 4 bölümden oluşan Königsberg şehrinin, aynı köprüden bir kez daha geçilmeyecek şekilde gezilmesinin mümkün olup olmadığı sorusu sorulmuştur
- Euler teoreminden tanıdığımız Leonard Euler bu tür yönsüz (undirected) grafiklerde, aynı çizgiden bir daha geçmeyecek şekilde grafiğin tamamını dolaşmanın mümkün olup olmadığını formülize etmiştir

Grafik Teorisi

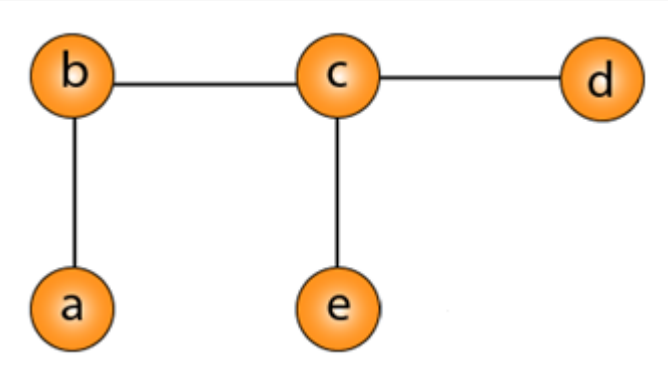
- Königsberg problemiyle birlikte grafik teorisi terimi bilim dünyasına giriş yapmıştır
- Grafik teorisi grafikleri inceleyen bir matematik dalıdır
- Grafik teorisinin ilgilendiği bazı problemler
  - *En kısa yol bulma* *(Shortest path)*
  - *Minimum ağaç kapsama* *(Minimum spaning tree)*
  - *Döngü bulma* *(Finding cycles)*
  - *Bağlantı bulma* *(Finding connectedness)*
  - *Minimum düğüm örtme* *(Minimum vertex cover)*

Minimum Düğüm Örtme Problemi

- Minimum düğüm örtme probleminde, düğüm ve kenarlarla ifade edilen bir grafikte, en az sayıda düğüm seçilerek bütün kenarların çizilmesi hedeflenir
- Bir düğüm seçildiğinde, o düğümle teması olan bütün kenarlar kapsanmış sayılır

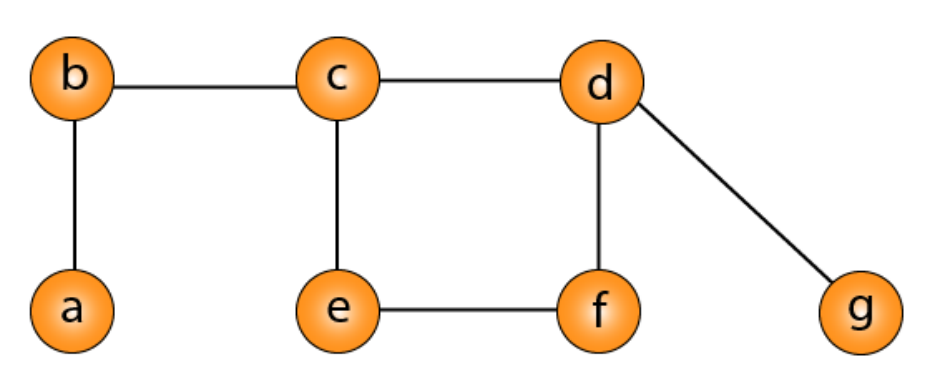
düğümler = {a, b, c, d, e}

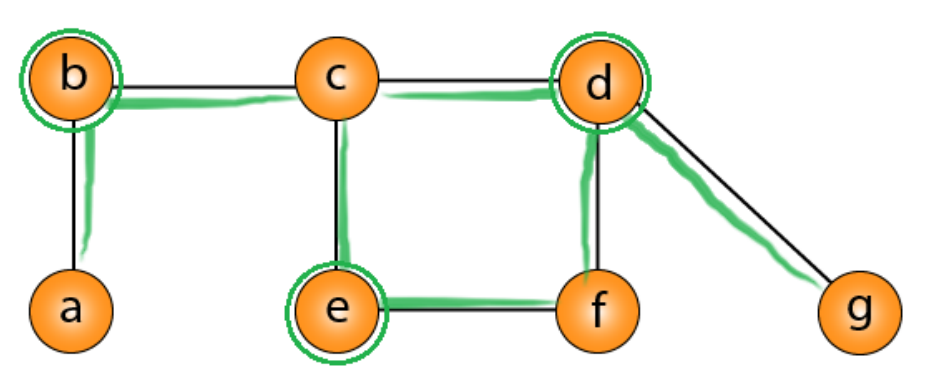
kenarlar = {{a, b}, {b, c}, {c, e}, {c, d}}



Minimum Düğüm Örtme Problemi

- Grafiğin kapsanması
- *b*, *e*, *d* düğümleri





Minimum Düğüm Örtme Problemi

- Minimum düğüm örtme probleminin çeşitli alanlarda gerçek dünya uygulamaları bulunmaktadır
- Uygulama alanlarından bazıları
  - *kablosuz iletişim*
  - *devre tasarımı ve ağ akışları*
  - *endüstriyel makine ataması*
  - *veri toplama*
  - *ağ güvenliği*

Tabu Arama Algoritması

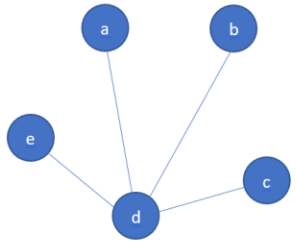
Tabu Search Pseudo Code

```
S_best Tabu_Search(TabuList)
begin
  S_current, S_candidate, S_best ← ConstructInitialSolutions();
  LongTermMemory (LTM) ← InitializeMemoryLTM();
  ShortTermMemory (STM) ← InitializeMemorySTM();

  repeat
    S_candidate = SelectBestNeighbor();

    if ((not move(STM, S_candidate)) OR (Aspiration(S_candidate))
    then
      S_current = S_candidate;
      S_best = UpdateBest(S_best, S_candidate)
      STM = UpdateShortTermMemory(STM + S_current)
      LTM = UpdateLongTermMemory(LTM + S_best)
    end if
  until StopCondition()

  return S_best
end
```

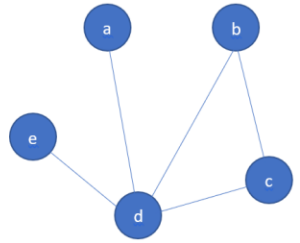
Uygulama

- Geliştirilen projede kullanılan yazılım dili Java'dır
- Geliştirme platformu olarak IntelliJ IDEA kullanılmıştır
- Projede veri tabanı kullanımına ihtiyaç duyulmamıştır
- Proje bir konsol uygulamasıdır
- Uygulama çıktıları konsol ekranından okunarak değerlendirilmiştir
- Kullanılan veriler
  - *Küçük boyutlu grafikler*
  - *Büyük boyutlu grafikler*

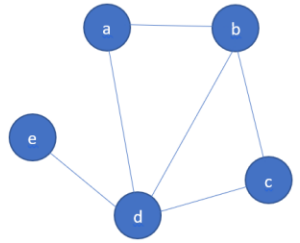
GRAFİK 1



GRAFİK 2



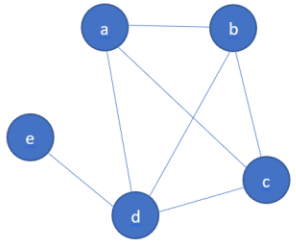
GRAFİK 3



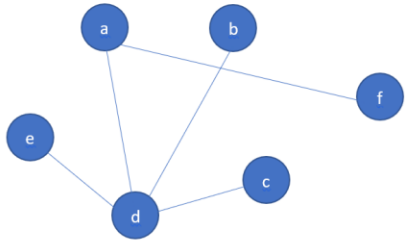
Kullanılan Veriler

- Küçük boyutlu grafikler
  - *En fazla 8 düğüm ve 12 kenardan oluşan 10 grafik*
  - *Kodun içinde array olarak tanımlı*

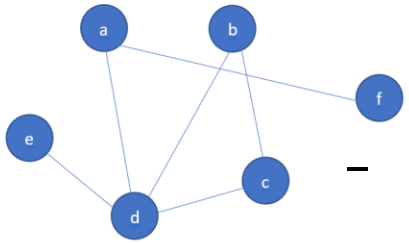
GRAFİK 4



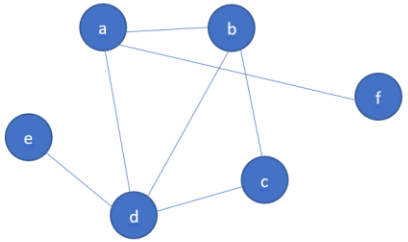
GRAFİK 5



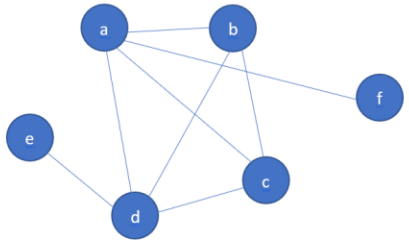
GRAFİK 6



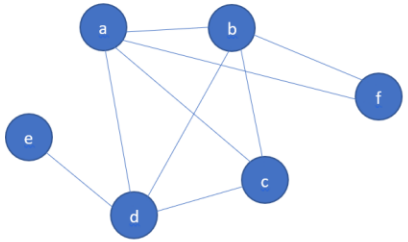
GRAFİK 7



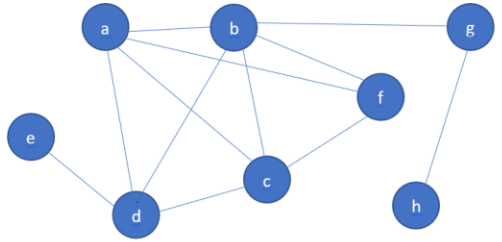
GRAFİK 8

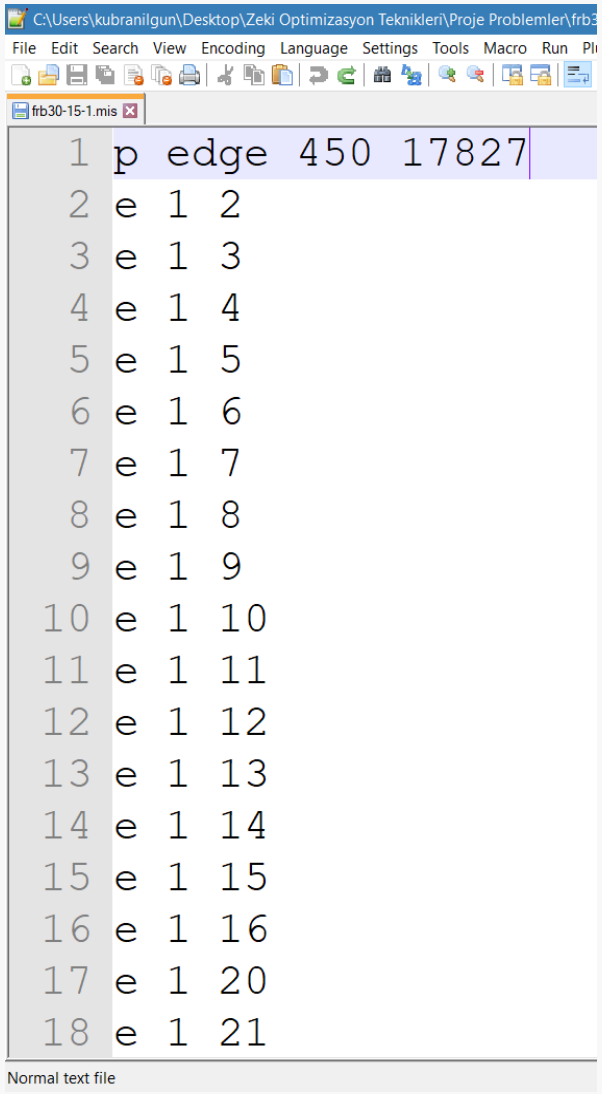


GRAFİK 9



GRAFİK 10





Kullanılan Veriler

- Büyük boyutlu grafikler
  - *Pekin'de bulunan Beihang Üniversitesi'nin sağladığı grafik veri seti*
  - *5 adet «.mis» uzantılı text dosyası*
  - *Her grafikte 450 düğüm ve en fazla 17874 kenar*

Uygulama Adımları

- Çözülecek problem seçilir
- Başlangıç çözümü üretilir
- Başlangıç çözümü uzun dönemli hafızaya alınır
- Başlangıç çözümünün ceza puanı hesaplanır
- Durma şartı sağlanana kadar en iyi çözümün aranması için aşağıdaki adımlar tekrarlanır
  - *Komşu çözümler belirlenir*
  - *En iyi komşu çözüm seçilerek, aday çözüm yapılır*
  - *Aday çözüm tabu değilse veya tabu yıkma kriterini karşılıyorsa aşağıdaki adımlara geçilir*
    - Aday çözüm mevcut çözüm yapılır
    - Aday çözüm en az en iyi çözüm kadar iyiyse, en iyi çözüm yapılır
    - Eklenmesi gereken tabu varsa tabu listesi güncellenir
    - En iyi çözüm uzun dönemli hafızaya eklenir

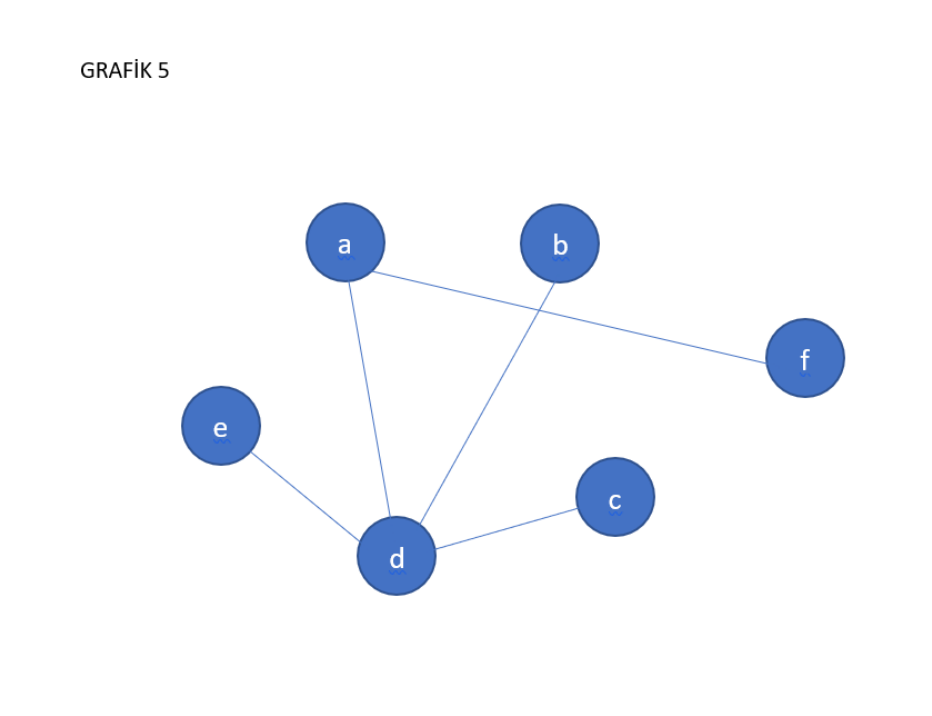
Uygulama İşleyişi

- Başlangıç çözümünün üretilmesi
  - *Her kenarın iki ucundan rastgele bir düğüm seçilerek başlangıç çözümü üretilir*
- Komşu çözümlerin üretilmesi
  - *En iyi çözümün rastgele bir düğümü değiştirilerek komşu çözüm listesi oluşturulur*
- Çözümlerin değerlendirilmesi
  - *Gelen çözüm grafikteki tüm kenarlara ulaşılma durumu ve örtülmüş düğüm sayısına göre değerlendirilir*
    - Çözüm tüm kenarlara ulaşmıyorsa, çözüme problemdeki düğüm sayısının 10 katı kadar ceza puanı verilir
    - Ceza puanı, çözümde seçili düğüm sayısı kadar arttırılır
- Küçük grafikler için durma şartı
  - *En iyi çözümün ceza puanı, problemdeki düğüm sayısı kadar iterasyon boyunca sabit kaldıysa, durma şartı sağlanmış sayılır*
- Büyük grafikler için durma şartı
  - *En iyi çözümün ceza puanı, problemdeki düğüm sayısının 15'te biri kadar iterasyon boyunca sabit kaldıysa, durma şartı sağlanmış sayılır*

Uygulama İşleyişi

- Tabu listesinde, en iyi çözümde gerçekleşmiş olan değişiklikler tutulur
- Küçük grafikler için tabu listesinin güncellenmesi
  - *Tabu listesindeki eleman sayısı problemdeki düğüm sayısına ulaşmışsa, ilk eleman tabu listesinden çıkarılır (FIFO)*
- Büyük grafikler için tabu listesinin güncellenmesi
  - *Tabu listesindeki eleman sayısı problemdeki düğüm sayısının 15'te birine ulaşmışsa, ilk eleman tabu listesinden çıkarılır (FIFO)*
- Tabu yıkma şartının sağlanması
  - *Aday çözüm en az en iyi çözüm kadar iyiyse, tabu yıkma kriterini sağlamış sayılır*

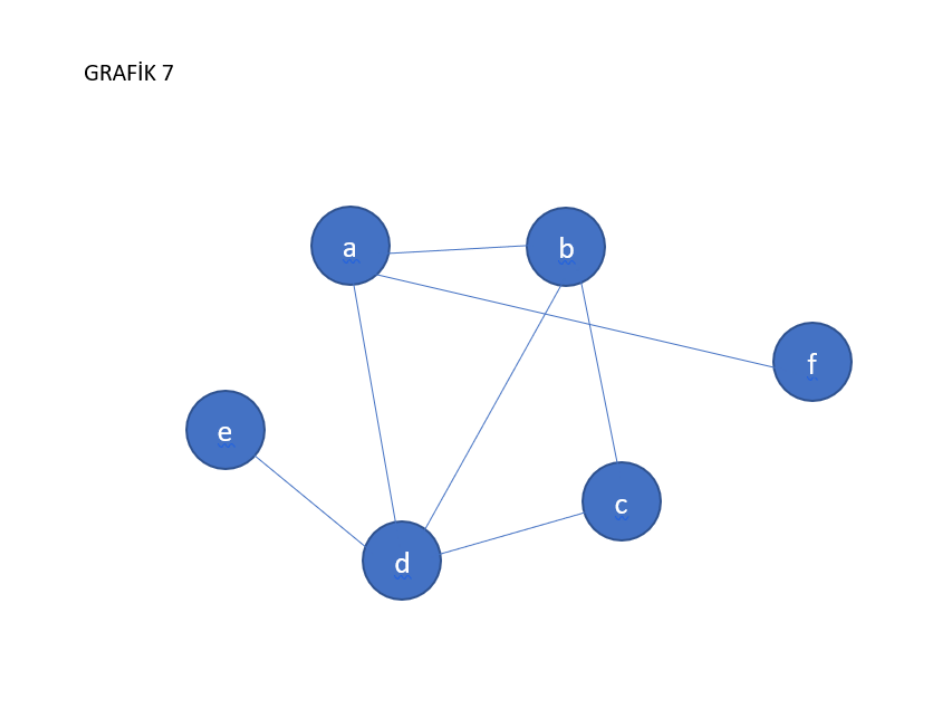
Sonuçlar



- Küçük grafikler
- Üretilen çözümler
  - ***a, d***
  - *Iteration number: 8*
  - *Penalty point of best solution: 2*

  - ***d, f***
  - *Iteration number: 6*
  - *Penalty point of best solution: 2*

Sonuçlar



- Küçük grafikler
- Üretilen çözümler
  - ***a, b, d***
  - *Iteration number: 6*
  - *Penalty point of best solution: 3*

  - ***a, c, d***
  - *Iteration number: 9*
  - *Penalty point of best solution: 3*

  - ***b, d, f***
  - *Iteration number: 8*
  - *Penalty point of best solution: 3*

Sonuçlar

- Büyük grafikler
- Üretilen çözümler
  - *Iteration number: 65*
  - *Penalty point of ultimate solution: 430*
  - *Iteration number: 81*
  - *Penalty point of ultimate solution: 429*
  - *Iteration number: 71*
  - *Penalty point of ultimate solution: 430*
  - *Iteration number: 74*
  - *Penalty point of ultimate solution: 431*

Bulgular

- Uygulamanın hem küçük hem de büyük grafikler için ürettiği çözümlerin yaklaşık %99 oranında başarı sağladığı görülmüştür
  - *Tabu arama algoritması minimum düğüm örtme problemi için başarılı sonuçlar üretebilmektedir*
  - *Tabu arama algoritması kodlaması kolay ve işlem maliyeti düşük bir algoritmadır*
  - *Tabu arama algoritmasının minimum düğüm örtme probleminin çözümünde tercih edilebilecek bir algoritma olduğu sonucuna ulaşılmıştır*

Kaynaklar

- H. A. Dawood, «International Conference on Advanced Computer Science Applications and Technologies,» Graph Theory and Cyber Security, Erbil, 2014.
- A. Çevik, «Mühendis Beyinler,» 02 Aralık 2017. [Çevrimiçi]. https://www.muhendisbeyinler.net/graf-teorisi-nedir/. [Ziyaret Tarihi: 04 Haziran 2020].
- R. Li, S. Hu, Y. Wang ve M. Yin, «A local search algorithm with tabu strategy and perturbation mechanism for generalized vertex cover problem,» Neural Comput & Applic, cilt 28, p. 1775-1785, 2017.
- A. C. Çınar, «Ahmet Cevahir Çınar,» 07 Haziran 2017. [Çevrimiçi]. https://www.ahmetcevahircinar.com.tr/2017/06/07/np-np-complete-np-hard-nedir/. [Ziyaret Tarihi: 23 Nisan 2020].
- «Java T Point,» [Çevrimiçi]. https://www.javatpoint.com/daa-approximation-algorithm-vertex-cover. [Ziyaret Tarihi: 14 Haziran 2020].
- X. X. Y. Z. Lei Xu, «P2P Botnet Detection Using Min-Vertex Cover,» Journal of Networks, cilt 7, no. 8, pp. 1176-1181, 2012.
- C. S. K. P. Benjamin Armbruster, «A packet filter placement problem with application to defense against spoofed denial of service attacks,» European Journal of Operational Research, cilt 176, pp. 1283-1292, 2007.
- K. Xu, «BHOSLIB: Benchmarks with Hidden Optimum Solutions for Graph Problems (Maximum Clique, Maximum Independent Set, Minimum Vertex Cover and Vertex Coloring),» [Çevrimiçi]. http://sites.nlsde.buaa.edu.cn/~kexu/benchmarks/graph-benchmarks.htm. [Ziyaret Tarihi: 02 Haziran 2020].
- J. Brownlee, «Tabu Search,» Clever Algorithms: Nature Inspired Programming Recipes, 2015.
- İ. Çayıroğlu, «İbrahim Çayıroğlu,» [Çevrimiçi]. http://www.ibrahimcayiroglu.com/Dokumanlar/IleriAlgoritmaAnalizi/IleriAlgoritmaAnalizi-9.Hafta-TabuAramaAlgoritmasi.pdf. [Ziyaret Tarihi: 23 Mayıs 2020].
- Ö. Gürbüz, Tabu Arama Algoritmasının Kuyruk Problemine Uygulanması, Ankara: Hacettepe Üniversitesi, Yüksek Lisans Tezi, 2015.
- M. A. Akçayol, «Zeki Optimizasyon Teknikleri,» [Çevrimiçi]. http://w3.gazi.edu.tr/~akcayol/files/ZOT_L4StochasticAlgorithms.pdf. [Ziyaret Tarihi: 02 Haziran 2020].

Teşekkürler

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help MVCTabuSearch [C:\Users\kubranilgun\Desktop\Zeki Optimizasyon Teknikleri\MVCTabuSearch] - ...\src\Main.java - IntelliJ IDEA

MVCTabuSearch > src > Main

Main

Project

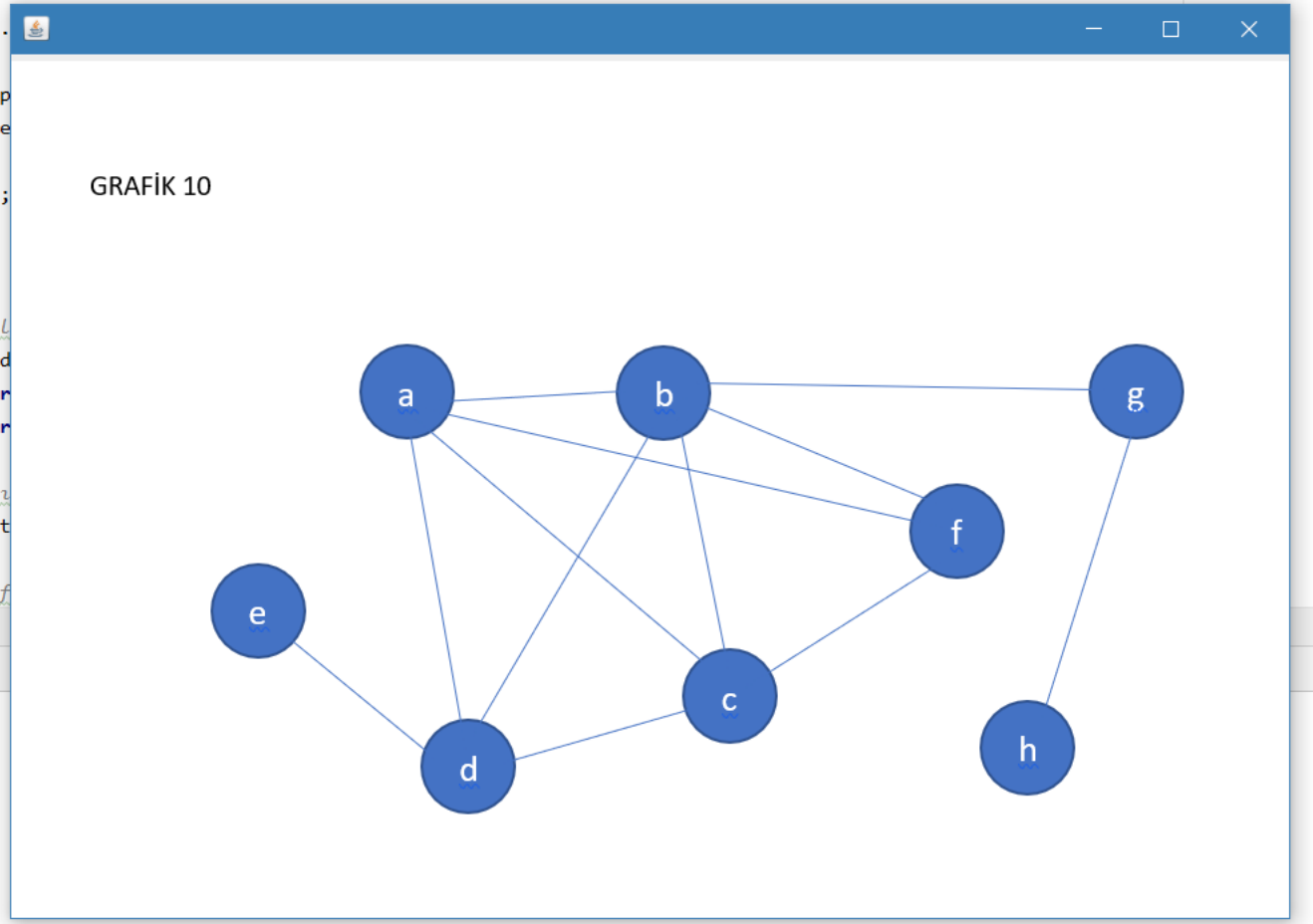
MVCTabuSearch C:\Users\kubranilgun\Desktop\Ze

External Libraries

Scratches and Consoles

Main.java Tabu.java

```
1  import javax.imageio.ImageIO;
2  import javax.swing.*;
3  import java.awt.*;
4  import java.awt.image.
5  import java.io.File;
6  import java.io.IOExcep
7  import java.lang.Syste
8  import java.util.*;
9  import java.util.List;
10
11 public class Main {
12
13     //Public değişkenl
14     public static Rand
15     public static char
16     public static char
17
18     //Tabu listesi (ku
19     public static List
20
21     //Uzun dönemli haf
```



Run: Main

```
true
false
Marked vertices of ultimate solution:
a
c
d
f
g
Penalty point of ultimate solution: 5
```

4: Run 6: TODO Terminal Event Log

All files are up-to-date (a minute ago) 101:1 CRLF UTF-8 4 spaces

21:06 15.06.2020

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help MVCTabuSearchLarger [C:\Users\kubranilgun\Desktop\Zeki Optimizasyon Teknikleri\MVCTabuSearchLarger] - ...\src\Main.java - IntelliJ IDEA

MVCTabuSearchLarger > src > Main

Main

Project

MVCTabuSearchLarger C:\Users\kubranilgun\Desk

External Libraries

Scratches and Consoles

Main.java Tabu.java

```
1  import java.io.*;
2  import java.lang.System;
3  import java.util.*;
4  import java.util.List;
5
6  public class Main {
7
8      //Public değişkenler tanımla
9      public static Random rand =
10     public static int[] vertices
11     public static int[][] edges;
12
13     //Tabu listesi (kısa dönemli
14     public static List<Tabu> sho
15
```

C:\Users\kubranil...

File Edit Search View Encoding
Language Settings Tools Macro Run
Plugins Window ? X

frb30-15-1.mis

```
1  p edge 450 17827
2  e 1 2
3  e 1 3
4  e 1 4
5  e 1 5
6  e 1 6
7  e 1 7
8  e 1 8
9  e 1 9
10 e 1 10
11 e 1 11
12 e 1 12
13 e 1 13
14 e 1 14
15 e 1 15
16 e 1 16
17 e 1 20
18 e 1 21
19 e 1 27
20 e 1 34
21 e 1 41
22 e 1 42
23 e 1 44
24 e 1 94
25 e 1 96
26 e 1 97
```

Windows (CR LF) UTF-8 INS

Run: Main

```
Iteration number: 70
Penalty point of best solution: 430

Iteration number: 71
Penalty point of best solution: 430

Iteration number: 72
Penalty point of best solution: 430

Iteration number: 73
Penalty point of best solution: 430

Penalty point of ultimate solution: 430

Process finished with exit code 0
```

4: Run 6: TODO Terminal Event Log

All files are up-to-date (8 minutes ago) 231:1 CRLF UTF-8 4 spaces

21:08 15.06.2020