Çizge Renklendirme ile Ders Programı Hazırlama

Proje 2 - Yazılım Geliştirme Laboratuvarı I

Binnur ÖZCAN
191307059
191307059@kocaeli.edu.tr
Kocaeli Üniversitesi – Bilişim
Sistemleri Mühendisliği
https://github.com/binnurozcan/yazlab

Bu proje, ders programı oluşturma sürecini otomatize etmek ve optimize etmek amacıyla geliştirilmiştir. Projede, bir ağ yapısı kullanılarak derslerin ve akademisyenlerin ilişkileri modellenmiş ve programın kullanıcı ara yüzü ile etkileşimi sağlanmıştır.

Anahtar Kelimeler—NetworkX, Matplotlib, PyOBDC, C#(Windows Forms), Veritabanı(SQL Server), Ders Programı, Renklendirme Algoritması, Graf Teorisi

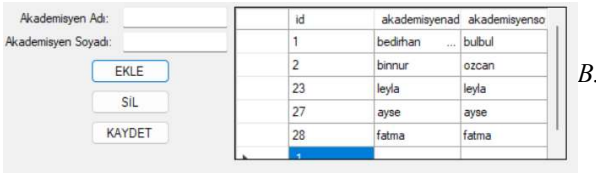
I. INTRODUCTION (*HEADING 1*)

Bu proje, bir üniversitenin ders programı oluşturma sürecini optimize etmeyi hedefleyen bir yazılım geliştirme laboratuvarı projesidir. Geliştirilen sistem, akademisyenlerin, derslerin ve bu öğeler arasındaki ilişkilerin yönetilmesini sağlayarak, bir çatışma çözümü ve etkili bir ders programı oluşturma amacını taşımaktadır. Projenin temel bileşenleri arasında Python dilinde NetworkX kütüphanesi, Matplotlib görselleştirme kütüphanesi, PyODBC veritabanı bağlantı kütüphanesi ve C# ile geliştirilmiş bir Windows Forms kullanıcı ara yüzü bulunmaktadır. Bu bileşenler, ders programının oluşturulmasında kullanılan araçları ifade etmektedir. Proje, bir Microsoft SQL Server veritabanı üzerinde akademisyen, ders ve kısıt verilerini depolamakta; ardından bu verileri kullanarak bir ağ yapısı oluşturmakta ve çeşitli kısıtlamalar altında çatışmasız bir ders programı elde etmeye yönelik algoritmalar içermektedir. Optimizasyon, renklendirme algoritmaları ve çakışma çözümü gibi kavramlar, projenin temel odak noktalarını oluşturmaktadır. Bu proje, üniversitelerin ders programı oluşturma süreçlerini daha etkili ve yönetilebilir hale getirmeyi amaçlamaktadır.

II. KULLANIM

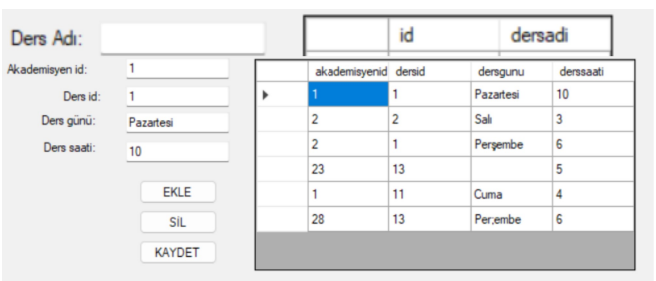
A. *Kullanıcı Dostu Arayüz*

Proje, Windows Forms kullanıcı ara yüzü ile tasarlanmıştır. Kullanıcı dostu bir arayüz, kullanıcıların projeyi rahatça anlamalarına ve kullanmalarına olanak tanır.



B. *Veritabanı Entegrasyonu*

PyODBC kütüphanesi aracılığıyla sağlanan veritabanı entegrasyonu sayesinde, akademisyen, ders ve kısıt verileri kolayca yönetilebilir. Kullanıcılar, veritabanındaki bilgileri güncelleyebilir, ekleyebilir veya silebilir.



C. *Grafiksel Gösterim*

Ders programının oluşturulmasında kullanılan ağ yapısı, Matplotlib kütüphanesi ile grafiksel olarak gösterilmektedir. Bu, kullanıcıların programı daha iyi anlamalarına ve çakışmaları görsel olarak belirlemelerine yardımcı olur.

D. *Hızlı Veri Girişi*

Kullanıcılar, akademisyenlerin, derslerin ve kısıtların hızlı bir şekilde eklenmesi veya güncellenmesi için özel butonlar ve formlar aracılığıyla kolayca işlem yapabilirler.

Bu kullanım kolaylığı özellikleri, proje kullanıcılarının ders programı oluşturma sürecini daha verimli ve kullanıcı dostu bir şekilde yönetmelerine olanak tanımaktadır.

III. KULLANILAN TEKNOLOJILER

A. *app.py*

1. NetworkX: Python programlama dilinde, ağ analizi ve graf teorisi için kullanılan bir kütüphanedir. Projede, ders programı oluşturmak için kullanılan ders-hoca ilişkilerini temsil etmek ve çeşitli graf algoritmalarını uygulamak için NetworkX kullanılmıştır.

2. Matplotlib: Python dilinde, grafik oluşturmak için kullanılan bir kütüphanedir. Projede, NetworkX tarafından oluşturulan ders-hoca ilişkileri grafiği Matplotlib kullanılarak görsel olarak ifade edilmiştir.

3. PyODBC: Python dilinde, Open Database Connectivity (ODBC) aracılığıyla veritabanlarına bağlanmak için kullanılan bir kütüphanedir. Bu projede,

4. Microsoft SQL Server veritabanına bağlanmak ve akademisyen, ders ve kısıt verilerini çekmek için PyODBC kullanılmıştır.Microsoft SQL Server: İlişkisel bir veritabanı yönetim sistemi olan SQL Server, projede akademisyen, ders ve kısıt verilerini depolamak için kullanılmıştır.

5. Windows Forms: C# programlama dilinde, Windows uygulamaları geliştirmek için kullanılan bir teknolojidir. Projede, kullanıcı arayüzünün oluşturulması ve kullanıcıların veritabanıyla etkileşimde bulunması için Windows Forms kullanılmıştır.

B. form1.cs

1. Windows Forms (WinForms): Windows Forms, Microsoft tarafından geliştirilen bir .NET framework teknolojisidir. Bu teknoloji, masaüstü uygulamaları geliştirmek için kullanılır. Form1 sınıfı, Windows Forms uygulamasının ana formunu temsil eder.

2. C#: Microsoft tarafından geliştirilen .NET framework ile uyumlu, genel amaçlı bir programlama dilidir. Bu proje C# dilinde yazılmıştır.

3. SQL Server ve ADO.NET: Veritabanı bağlantısı için SQL Server kullanılmıştır. ADO.NET (ActiveX Data Objects .NET), .NET framework içinde bulunan bir teknolojidir ve veritabanlarına erişim sağlar. SqlConnection, SqlDataAdapter ve diğer ADO.NET sınıfları bu projede kullanılmıştır.

4. DataGridView: Windows Forms uygulamalarında tablo verilerini göstermek için kullanılan bir kontrol. Bu kontrol, veritabanındaki tabloları görüntülemek ve kullanıcı ile etkileşimde bulunmak için kullanılmıştır.

5. MessageBox: Kullanıcıya bilgi, uyarı veya hata mesajları göstermek için kullanılan bir Windows Forms kontrolüdür. Bu projede, kullanıcıya kaydetme işlemiyle ilgili bilgi mesajları göstermek için MessageBox kullanılmıştır.

6. DataSet ve Data Binding: DataSet, veritabanından alınan verileri geçici olarak depolamak için kullanılan bir sınıftır. Data Binding ise veri kaynağından gelen verileri doğrudan kullanıcı arayüzüne bağlamak için kullanılır. Bu projede, DataSet ve Data Binding kullanılarak verilerin yönetimi sağlanmıştır.

IV. Kod Açıklamaları

A. *app.py*

- *networkx, matplotlib ve pyodbc kütüphaneleri projeye eklenmiştir.*

```
import networkx as nx
import matplotlib.pyplot as plt
import pyodbc
```

- *MySQL bağlantısı*

```
# MySQL bağlantısı
conn = pyodbc.connect(
    'DRIVER={SQL Server};'
    'SERVER=LAPTOP-CBRO5FHT;'
    'DATABASE=dersProgrami;'
    'PWD=1234;'
    'Trusted_Connection=True;'
)
cursor = conn.cursor()
```

- *Veritabanından akademisyenlerin isimleri çekiliyor.*

```
# Akademisyen isimlerini veritabanından çek
cursor.execute("SELECT id, akademisyenadi FROM akademisyen")
akademisyenler = cursor.fetchall()
```

- *Veritabanından ders-hoca ilişkileri çekiliyor.*

```
# Ders-Hoca ilişkilerini al
cursor.execute("SELECT dersid, akademisyenid FROM kisitlar")
ders_akademisyen = cursor.fetchall()
```

- *Veritabanından kısıtlar çekiliyor.*

```
# Kısıtları al
cursor.execute("SELECT akademisyenid, dersgunu, derssaati FROM kisitlar")
kisitlar = cursor.fetchall()
```

- *Veritabanından ders isimleri çekiliyor.*

```
# Ders isimlerini veritabanından çek
cursor.execute("SELECT id, dersadi FROM ders")
dersler = cursor.fetchall()
```

- *Graf oluştur.*

```
# Graf oluştur
G = nx.Graph()
```

- *Akademisyen düğümleri oluşturuluyor.*

```
for akademisyen in akademisyenler:
    G.add_node(f"Akademisyen_{akademisyen[0]}", label=f"{akademisyen[1]}", tur="Akademisyen", akademisyenid=akademisyen[0])
```

- *Ders düğümleri oluşturuluyor.*

```
for ders in dersler:
    G.add_node(f"Ders_{ders[0]}", label=f"{ders[1]}", tur="Ders", dersid=ders[0])
```

- *Ders-hoca ilişkileri oluşturuluyor.*

```
for iliski in ders_akademisyen:
    G.add_edge(f"Ders_{iliski[0]}", f"Akademisyen_{iliski[1]}")
```

- *Kısıtlar ekleniyor.*

```
for kisit in kisitlar:
    hoca_node = f"Akademisyen_{kisit[0]}"
    G.nodes[hoca_node]["dersgunu"] = kisit[1]
    G.nodes[hoca_node]["derssaati"] = kisit[2]
```

- *Renklendirme fonksiyonu*

```
# renklendirme fonksiyonu
def custom_coloring(G):
    coloring = {}
    for node in G.nodes:
        if satisfies_constraints(G.nodes[node], coloring):
            color = assign_color(node, coloring)
            coloring[node] = color
    return coloring
```

- *Kısıtlamaları kontrol eden fonksiyon*

```
def satisfies_constraints(node, coloring):
    hoca_gun = node["dersgunu"]
    hoca_saat = node["derssaati"]
    hoca_id = node ["akademisyenid"]
    ders_id = node ["dersid"]
    for other_node, other_props in G.nodes.items():
```

- .Eğer aynı gün, saat, akademisyen ve ders ID'sine sahip bir başka düğüm varsa

```
if (
    # Eğer aynı gün, saat, akademisyen ve ders ID'sine sahip bir başka düğüm varsa
    other_node != node and
    other_props.get("dersgunu") == hoca_gun and
    other_props.get("derssaati") == hoca_saat and
    other_props.get("akademisyenid") == hoca_id and
    other_props.get("dersid") == ders_id
):
    return False
```

- Eğer aynı gün ve saatte bir başka düğüm varsa

```
if (
    # Eğer aynı gün ve saatte bir başka düğüm varsa
    other_node != node and
    other_props.get("dersgunu") == hoca_gun and
    other_props.get("derssaati") == hoca_saat
):
```

- Eğer aynı gün, saat ve akademisyene sahip bir başka düğüm varsa

```
if ( # Eğer aynı gün, saat ve akademisyene sahip bir başka düğüm varsa
    other_node != node and
    other_props.get("dersgunu") == hoca_gun and
    other_props.get("derssaati") == hoca_saat and
    other_props.get("akademisyenid") == hoca_id
):
    return False
```

- Eğer aynı gün, saat ve ders ID'sine sahip bir başka düğüm varsa

```
if (# Eğer aynı gün, saat ve ders ID'sine sahip bir başka düğüm varsa
    other_node != node and
    other_props.get("dersgunu") == hoca_gun and
    other_props.get("derssaati") == hoca_saat and
    other_props.get("dersid") == ders_id
):
    return False
```

- Yukarıdaki koşulların hiçbiri sağlanmıyorsa, kısıtlamalar sağlanıyor demektir

```
    # Eğer yukarıdaki koşulların hiçbiri sağlanmıyorsa, kısıtlamalar sağlanıyor demektir
return True
```

- Düğümlere renk atayan fonksiyon

```
def assign_color(node, coloring):
    if satisfies_constraints(node, coloring):
        color = assign_color(node, coloring)
        return color
    else:
        return 1
```

- Greedy renklendirme algoritması kullanılarak düğümlere renk atama ve düğüm etiketleri

```
# Greedy renklendirme algoritması kullanılarak düğümlere renk atama
coloring = nx.coloring.greedy_color(G, strategy="largest_first")
# Düğüm etiketleri
labels = {node: G.nodes[node]['label'] for node in G.nodes}
```

- Graf çizimi

```
nx.draw(
    G,
    with_labels=True,
    labels=labels,
    font_weight="bold",
    node_color=list(coloring.values()),
    cmap=plt.cm.rainbow,
    node_size=100,
    edge_color='gray',
    width=1.5,
    node_shape='s',
)
```

B. form1.cs

- *Form1 Sınıfı ve Constructor:*
  Form1 sınıfı Form sınıfından miras alır. Constructor (Form1()) içinde, formun başlatılması için InitializeComponent() metodu çağrılır. Bu metot, tasarım görünümündeki bileşenlerle ilgili tanımlamaları içerir.

```
3 başvuru
public partial class Form1 : Form
{

    1 başvuru
    public Form1()
    {
        InitializeComponent();

    }
```

- *Load Olayları:*
  Form1_Load_1 metodu, form yüklendiğinde gerçekleşen olaydır. Bu metot içinde, Fill metotları kullanılarak veritabanındaki tabloların yüklenmesi sağlanır.

```
1 başvuru
private void Form1_Load_1(object sender, EventArgs e)
{
    // TODO: Bu kod satırı 'dersProgramiDataSet.kisitlar' tablosuna veri yük
    this.kisitlarTableAdapter.Fill(this.dersProgramiDataSet.kisitlar);
    // TODO: Bu kod satırı 'dersProgramiDataSet.ders' tablosuna veri yükler.
    this.dersTableAdapter.Fill(this.dersProgramiDataSet.ders);
    // TODO: Bu kod satırı 'dersProgramiDataSet.akademisyen' tablosuna veri
    this.akademisyenTableAdapter.Fill(this.dersProgramiDataSet.akademisyen);

}
```

- *Veri Ekleme, Düzenleme ve Silme İşlemleri:*
  Bu metotlar, kullanıcının form üzerindeki butonlara tıklaması durumunda gerçekleşen olayları işler. Yeni veri ekleme, düzenleme ve silme işlemlerini yapar.

```
private void button4_Click(object sender, EventArgs e)
{
    // Yeni akademisyen ekleme
    akademisyenBindingSource.AddNew();
}
```

```
private void button7_Click(object sender, EventArgs e)
{

    // Akademisyen verilerini güncelleme ve kaydetme
    // Veri düzenlemeyi bitir
    akademisyenBindingSource.EndEdit();

    akademisyenTableAdapter.Update(dersProgramiDataSet);

    // Başarıyla güncellendi
    MessageBox.Show("KAYDEDILDI", "Bilgi", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

```
private void button5_Click(object sender, EventArgs e)
{
    // Seçili akademisyeni silme
    akademisyenBindingSource.RemoveCurrent();

}
```

- *Veritabanı Güncelleme İşlemleri:*
  Bu metot, ders verilerini güncelleme ve kaydetme işlemini gerçekleştirir.

```
1 başvuru
private void button9_Click(object sender, EventArgs e)
{
    // Ders verilerini güncelleme ve kaydetme
    dersBindingSource.EndEdit();
    dersTableAdapter.Update(dersProgramiDataSet);

    // Başarıyla güncellendi
    MessageBox.Show("KAYDEDILDI", "Bilgi", MessageBoxButtons.OK, MessageBoxIcon.Information);
}
```

- *Kısıtlar İşlemleri:*
  Bu metotlar, seçili kısıtları silme ve kısıtlar verilerini güncelleme ve kaydetme işlemlerini gerçekleştirir.

```
0 başvuru
private void button11_Click(object sender, EventArgs e)
{
    // Seçili kısıtları silme
    kisitlarBindingSource.EndEdit();

    kisitlarTableAdapter.Update(dersProgramiDataSet);
    kisitlarBindingSource.RemoveCurrent();
}

0 başvuru
private void button12_Click(object sender, EventArgs e)
{
    // Kısıtlar verilerini güncelleme ve kaydetme
    // Veri düzenlemeyi bitir
    kisitlarBindingSource.EndEdit();

        // Veritabanını güncelle
    kisitlarTableAdapter.Update(dersProgramiDataSet);


        // Başarıyla güncellendi
    MessageBox.Show("KAYDEDILDI", "Bilgi", MessageBoxButtons.OK, MessageBoxIcon.Information);

}
```