TÜRKİYE CUMHURİYETİ
YILDIZ TEKNİK ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

İNTERNET TABANLI ETKİLEŞİMLİ ÇİZGE KURAMI ARACI

19011018 — Bedrettin Şamil ÖZTÜRK
19011029 — Alper EREN

BİLGİSAYAR PROJESİ

Danışman
Dr. Ahmet ELBİR

Mayıs, 2023

TEŞEKKÜR

Proje danışmanlığı ve akademik rehberlik gibi kritik alanlarda Sayın Ahmet Elbir'in sınırsız desteği ve yakın ilgisi derslerimizde olduğu gibi proje sürecimizde de önemli bir rol oynamıştır. Kendisi, her zaman öğrencilerinin başarılarına odaklanarak, yoğun temposunda bizimle vakit ayırıp ilgilenerek yardımını esirgememiştir.

Bu nedenle, Sayın Ahmet Elbir hocamıza en içten teşekkürlerimizi sunarız. Kendisi, her zaman yanımızda olan, sorunlarımıza çözümler bulan ve öğrencilerinin başarısı ve başarısına giden yolda yardım adına için elinden gelen her şeyi yapan bir danışman olmuştur.

Bedrettin Şamil ÖZTÜRK
Alper EREN

İÇİNDEKİLER

KISALTMA LİSTESİ **v**

ŞEKİL LİSTESİ **vi**

ÖZET **vii**

ABSTRACT **ix**

1 Giriş **1**

2 Ön İnceleme **3**

2.1 Çizge Teorisi ve Algoritmalar 3

2.2 Çizge Görselleştirme ve Etkileşim Araçları 3

2.3 Çizge Teorisi için Web Tabanlı Eğitim Platformları 4

3 Fizibilite **5**

3.1 Teknik Fizibilite 5

3.1.1 Yazılım Fizibilitesi 5

3.1.2 Donanım Fizibilitesi 6

3.2 Ekonomik Fizibilite 6

3.3 Yasal Fizibilite 6

3.4 İş-Zaman Fizibilitesi 7

4 Sistem Analizi **8**

4.1 İşlevsel gereksinimler: 8

4.1.1 Çıktı Biçimi 8

4.1.2 Arayüz 9

4.1.3 Veri 9

4.2 Kullanım Senaryosu 10

4.3 Aktivite Diyagramı 13

5 Sistem Tasarımı **14**

5.1 Yazılım Tasarımı 14

5.1.1 Problem Çözümü / Çözüm Tasarımı 14

5.1.2 Son Kullanıcı (Arayüz) Odaklı Tasarım 14
5.1.3 Sistem Şeması 15
5.1.4 Algoritmik Yapı 15
5.2 Veri Tabanı Tasarımı 15
5.3 Girdi-Çıktı Tasarımı 15

6 Uygulama **16**

7 Deneysel Sonuçlar **20**

8 Performans Analizi **21**
8.1 Algoritma Performansı 21
8.2 Kullanıcı Deneyimi ve Yanıt Süresi 21
8.3 Ölçeklenebilirlik 21

9 Sonuç **23**

Referanslar **24**

Özgeçmiş **26**

KISALTMA LİSTESİ

JSON	JavaScript Object Notation
CSS	Cascading Style Sheets
DFS	Depth First Search
BFS	Breadth First Search
HTML	HyperText Markup Language

ŞEKİL LİSTESİ

Şekil 3.1 GANTT Diyagramı 7
Şekil 4.1 Kullanım Senaryosu Diyagramı 12
Şekil 4.2 Aktivite Diyagramı 13
Şekil 6.1 Dosya Organizasyonu 16
Şekil 6.2 Çizge Örneği 18
Şekil 6.3 Graph Info Bileşeni 19
Şekil 7.1 Import/Export Modülü Testi 20
Şekil 8.1 Performans Analizi 22

ÖZET

İNTERNET TABANLI ETKİLEŞİMLİ ÇİZGE KURAMI ARACI

Bedrettin Şamil ÖZTÜRK
Alper EREN

Bilgisayar Mühendisliği Bölümü
Bilgisayar Projesi

Danışman: Dr. Ahmet ELBİR

Çizge teorisi, başta bilgisayar bilimi olmak üzere birçok disiplinde uygulanan ve önemli pratik sorunların çözülmesine yardımcı olan önemli bir veri yapısı teorisidir. Çizgeler, öğeleri veya varlıkları temsil eden düğümlerden ve bu nesneler veya varlıklar arasındaki ilişkileri gösteren bağlantı kenarlarından oluşur. Yönlü-yönsüz, ağırlıklı-ağırlıksız çizgeler gibi birkaç kategoriye giren çizgeler ilgili bir dizi sorunun çözülmesini sağlayan yöntemler sunar. Bunlardan bazıları, bir çizgedeki kaynak ve hedef düğümler arasındaki en kısa yolu hesaplamayı, her düğümü taramayı ve her düğümü bağlamak için en kısa kenar setini bulmayı içerir.

Bu proje kapsamında eğitim ve analiz dahil olmak üzere birincil hedefler belirlendi ve web tabanlı bir etkileşim çizge teorisi aracı oluşturuldu. Kullanıcıların diledikleri gibi çizgeler oluşturabilmelerine, çizge üzerinde diledikleri gibi düzenlemeler yapabilmelerine ve çizge üzerinde çizge algoritmaları yapabilmelerine olanak tanıyan bu araçta kullanıcı deneyimi, işlem doğruluğu, görsel tasarım gibi bazı hususlar dikkatle analiz edilmiştir. Uygulama bu kriterlere göre geliştirilmiştir.

Çizge görsel yapısındaki kullanıcı arayüzü, kullanıcıların düğümler, kenarlar ve kenarlar için ağırlıklar oluşturmasına, istenen esnekliği sunmasına, oluşturulan çizgenin özelliklerine görüntülemesine, DFS, BFS, Kruskal's, Prim's, Dijkstra's gibi herhangi bir çizge algoritmasını simüle etmesine olanak tanır.

Anahtar Kelimeler: Çizge, dügüm, kenar, çizge algoritmaları, simülasyon, web tabanlı, çizge özellikleri, kenar ağırlığı

ABSTRACT

INTERNET-BASED INTERACTIVE GRAPHIC THEORY TOOL

Bedrettin Şamil ÖZTÜRK
Alper EREN

Department of Computer Engineering
Computer Project

Advisor: Dr. Ahmet ELBİR

Graph theory is an important theory of data structure that is applied in many disciplines, especially computer science, and helps solve important practical problems. Graphs are composed of nodes, which represent items or entities, and connecting edges, which show the relationships between these objects or entities. Graphs fall into several categories, such as directed and undirected, weighted and unweighted graphs, and provide methods for solving a number of related problems. Some of these include computing the shortest path between source and destination nodes in a graph, scanning each node, and finding the shortest set of edges to connect each node.

In this project, the primary objectives, including training and analysis, were identified and a web-based interaction graph theory tool was created. In this tool, which allows users to create graphs as they wish, edit the graph as they wish, and perform graph algorithms on the graph, some aspects such as user experience, processing accuracy, and visual design have been carefully analyzed. The application was developed according to these criteria.

The user interface in the graph visual structure allows users to create weights for nodes, edges and edges, offer desired flexibility, view the properties of the created graph, simulate any graph algorithm such as DFS, BFS, Kruskal's, Prim's, Dijkstra's, etc.

Keywords: Graph, node, vertex, edge, weighted graph, graph theory, graph

algorithms, simulations, web-based interaction, web-based graph drawing

1
Giriş

Verilerin karar verme, yenilik ve problem çözme konusundaki yeteneği dünya için giderek daha önemli hale geliyor. En önemli veri yapılarından; matematiğin ve bilgisayar biliminin kilit alanlarından biri olan çizge teorisi, sosyal ağlar, ulaşım, bilgisayar ağları ve daha fazlası dahil olmak üzere çeşitli bağlamlardaki şeyler arasındaki karmaşık ilişkileri anlamak için gereklidir [1]. Öte yandan, çizge algoritmaları, bu etkileşimleri verimli bir şekilde değerlendirmemizi, anlamamızı ve çeşitli gerçek dünya sorunlarına en iyi yanıtları belirlememizi sağlar.

Çizge teorisi, öğrencilere algoritmalar ve veri yapıları oluşturmak için gereken temel becerileri kazandırdığı için bilgisayar bilimi eğitiminin çok önemli bir bileşeni haline geldi. Çizge topolojileri ve özellikleri, yapay zeka, makine öğrenimi, ağ optimizasyonu ve biyoinformatik dahil birçok alanda doğrudan uygulamalara sahip olduğundan, bunların altını çizmek önemlidir. Sonuç olarak, öğrencilerin çizge teorisinin karmaşıklıklarını anlamalarına ve anladıklarını gerçek dünya ortamlarında uygulamalarına yardımcı olabilecek eğitim kaynaklarına yönelik bir talep vardır.

Çizge teorisinin artan önemi göz önüne alındığında, kullanıcıların kolayca çizge oluşturmasını, düzenlemesini, analiz etmesini ve görselleştirmesini kolaylaştıran etkileşimli, web tabanlı araçlara yönelik bir talep vardır. Bu projenin amacı, kullanıcıların çeşitli çizge algoritmaları oluşturmak, düzenlemek ve çalıştırmak için kullanabilecekleri çizge teorisi için kapsamlı bir web tabanlı etkileşimli araç oluşturmaktır. Uygulama aynı zamanda görsel olarak çekici ve kullanımı basit bir kullanıcı arayüzü sunar, bu da kullanıcı deneyimini geliştirir ve kullanıcıların önemli fikirleri anlamalarına yardımcı olur. Geliştirilen web uygulamasında dört ana modül bulunmaktadır.

Bunlar İçe/Dışa Aktarma modülü, Çizge Oluşturma modülü, Çizge Güncelleme modülü, Algoritmalar modülüdür.

Kruskal Algoritması, Prim Algoritması, Derinlik Öncelikli Arama (DFS), Genişlik Öncelikli Arama (BFS) ve Dijkstra Algoritması gibi iyi bilinen çizge algoritmalarını destekleyecektir.
Uygulama, kullanıcıların algoritmaları anlamalarını kolaylaştırmak için çeşitli görsel sunumlar ve animasyonlar sunarken büyük ölçekli çizgeleri işlemek üzere tasarlanacaktır.

Bu rapor; Projenin amacını ve önemini açıklayan Giriş bölümü ile başlar, ardından literatürün ve mevcut benzer çalışmaların gözden geçirildiği Ön İnceleme bölümü gelir. Yazılım Fizibilitesi bölümünde projenin yazılım açısından fizibilitesi ele alınırken Donanım Fizibilite bölümünde donanım gereksinimleri de değerlendirilir.

Yasal Fizibilite bölümü, projenin yasal gerekliliklere ve limitlere uygunluğu ile ilgilenirken, Ekonomik Fizibilite bölümü, projenin maliyet ve getiri açısından uygulanabilirliğine bakar. Sistem Tasarımı bölümü projenin yazılım ve veri tabanı tasarımını anlatırken, Sistem Analizi bölümü projenin fonksiyonel ve teknik ihtiyaçlarına bakar. Uygulama bölümü, projenin uygulanmasını ve kodlama prosedürlerini özetler. Her bölüm, ilgili proje öğelerini derinlemesine inceleyecek ve geliştirme sürecinin ve beklenen sonuçların bir özetini sunacaktır.

2
Ön İnceleme

Bu raporun literatür taraması bölümünün amacı, çizge teorisi, çizge algoritmaları ve etkileşimli öğrenme ortamları alanlarındaki son gelişmelere genel bir bakış sunmaktır. Önerilen web tabanlı etkileşimli çizge teorisi aracının oluşturulmasına, bu incelemenin alandaki mevcut en son teknolojilerin, sorunların ve olasılıkların tanımlanması rehberlik edecektir.

2.1 Çizge Teorisi ve Algoritmalar

18. yüzyılın başlangıcından bu yana, çizge teorisi hem matematik hem de bilgisayar teknolojisinde büyük bir ilgi gördü. Diğer birçok alanda uygulamaları olan çok sayıda çizge algoritmasının oluşturulması ve analizi, kapsamlı araştırmaların konusu olmuştur. Literatürde araştırılan anahtar algoritmalar şunları içerir:

Kruskal Algoritması [2], Prim Algoritması [3] , Derinlik Öncelikli Arama [4] , Genişlik Öncelikli Arama [5] ve Dijkstra Algoritması [6]. Bu algoritmalar, ağ tasarımı, yönlendirme, zamanlama ve optimizasyon problemleri gibi alanlarda yaygın olarak kullanılmaktadır.

2.2 Çizge Görselleştirme ve Etkileşim Araçları

Çizgeleri anlamak ve kullanmak, çok fazla etkileşim ve görselleştirme gerektirir. Zamanla, çizgeleri oluşturmayı, düzenlemeyi ve görüntülemeyi kolaylaştırmak için bir dizi araçlar ve kütüphaneler oluşturulmuştur. Bu araçlara örnek olarak, çizge ve ağ analizi için açık kaynaklı bir program olan Gephi [7] verilebilir . Esnek çoklu görev mimarisi sayesinde, devasa ağları gerçek zamanlı olarak görüntülemek için bir 3B motor kullanır ve karmaşık veri kümeleriyle başa çıkmak için yeni fırsatlar yaratır. Diğer bir örnek, entegre bir kavramsal çerçeve oluşturmak için yüksek verimli ifade verilerini ve diğer moleküler durumları biyomoleküler etkileşim ağlarıyla birleştiren açık kaynaklı bir yazılım projesi olan Cytoscape'dir [8].

Çizge çizim araçlarının bir koleksiyonu olan ve dot, neato, fdp, twopi gibi toplu düzenleme programlarını içeren Graphviz [9] da bir örnek olarak gösterilebilir. Çizge görselleştirme için iyi bilinen birkaç çerçeve D3.js [10] ve Sigma.js'dir [11]. Bu araçlar ve kütüphaneler sayesinde, kullanıcıların karmaşık çizim yapılarını daha başarılı bir şekilde oluşturması ve görselleştirmesi konusu önemli ölçüde gelişmiştir.

2.3 Çizge Teorisi için Web Tabanlı Eğitim Platformları

Bilgisayar bilimlerinin eğitimine artan vurgu ve çizge teorisinin artan önem ile beraber, öğrencilerin kavramları kavramalarına ve pratikte uygulamalarına yardımcı olmak için çeşitli web tabanlı eğitim platformları ortaya çıkmıştır. Bazı dikkate değer örnekler arasında Sharif Teknoloji Üniversitesi Matematik Bölümü'nde "GraphLab" adıyla geliştirilmeye başlanan, zengin bir çizge düzenleme çerçevesi olan GraphTea , istatistiksel çıkarım yoluyla ağların büyük ölçekli yapısını tanımlayan algoritmalar içeren Graph-Tool ve günlük dinamik problem modelleme ve klasik çizge yönetimi görevleri için geliştirilmiş bir Java tabanlı kütüphane olan GraphStream yer alır. Bu platformlar, çeşitli çizge işlemlerini ve görselleştirmeleri destekleyen etkileşimli öğrenme ortamları sağlar. Bununla birlikte, kapsamlı programlama geçmişine sahip olmayanlar da dahil olmak üzere daha geniş bir kullanıcı yelpazesine hitap eden daha kapsamlı ve erişilebilir araçlara ihtiyaç duyulmaktadır.

Öğrencilerin ilkeleri anlamalarına ve bunları uygulamaya koymalarına yardımcı olmak için çok sayıda web tabanlı öğretim platformu bir araya getirilmiştir. Bunun nedeni, bilgisayar bilimi eğitiminin ve çizge teorisinin giderek daha önemli hale gelmesidir. Örnekler arasında Sharif Teknoloji Üniversitesi Matematik Bölümü'nde "GraphLab" adı altında oluşturulan gelişmiş bir çizge düzenleme çerçevesi olan GraphTea [12], Graph-Tool [13] ve GraphStream [14] yer alır. Bu platformlar, farklı çizge işlemlerini ve görselleştirmeleri barındıran ilgi çekici öğrenme alanları sunar. Kapsamlı programlama becerilerine sahip olmayanlar da dahil olmak üzere daha geniş bir kullanıcı yelpazesi, daha eksiksiz ve ulaşılabilir araçlar gerektirir.

Literatürün incelenmesi, bilgisayar bilimleri eğitiminde çizge teorisi ve algoritmaların önemini, bu fikirleri öğretmek, uygulamak için verimli ve kolay erişilebilir web tabanlı araçlara olan talebi vurgulamaktadır. Önerilen web tabanlı etkileşimli çizge teorisi aracı, mevcut araçların en iyi yönlerini ve web teknolojilerindeki, görselleştirmedeki ve kullanıcı deneyimi tasarımındaki gelişmeleri birleştirerek bu gereksinimi karşılamayı amaçlamaktadır. Bu çaba, alanda devam eden çalışmaları ilerletecek ve çizge teorisi ve onun diğer disiplinlerdeki kullanımları hakkındaki bilgileri derinleştirecektir.

3
Fizibilite

Projenin gelmiş olduğu aşamada geçmiş olduğu bir dizi fizibilite aşamaları mevcuttur. Bunlar, çalışmanın başarıya ulaşması için alternatifleri arasından nasıl seçildiği, zaman planlaması, mali yönden ele alınması ve başarıya ulaşması durumunda sağlayacağı imkanlardır. Fizibilite çalışmaları farklı alt başlıklarla incelenecek ve açıklanacaktır.

3.1 Teknik Fizibilite

Teknik fizibilite, kendi içinde Yazılım Fizibilitesi ve Donanım Fizibilitesi şeklinde gruplanır.

3.1.1 Yazılım Fizibilitesi

Yazılım fizibilitesi bölümü, uygun kitaplıklar, geliştirme kolaylığı, sürdürülebilirlik ve ölçeklenebilirlik gibi faktörleri göz önünde bulundurarak önerilen web tabanlı etkileşimli çizge teorisi aracının teknik uygulanabilirliğini değerlendirir.

3.1.1.1 Çizge Görselleştirme ve Etkileşim

D3.js kitaplığı, proje için gerekli olan çok çeşitli görselleştirme tekniklerini ve etkileşimli özellikleri desteklediğinden, yüksek kaliteli çizge görselleştirme ve etkileşim için kullanılacaktır.

3.1.1.2 Çizge Algoritmaları Uygulaması

Çizge algoritmalarının uygulanmasında verimli veri yapıları ve algoritmalar kullanılacaktır. JavaScript, Map and Set gibi yerleşik veri yapıları sunar ve lodash veya underscore.js gibi kitaplıklar, ortak işlemleri basitleştirebilir.

3.1.1.3 Ölçeklenebilirlik ve Performans

Uygulama, performansı ve ölçeklenebilirliği optimize ederek büyük ölçekli çizgeleri işlemek için tasarlanacaktır. Verimli algoritmalar, veri yapıları ve modüler tasarım, yanıt verebilirliği ve gelecekteki genişlemeyi sağlayacaktır.

Sonuç olarak, yazılım fizibilite değerlendirmesi, uygun kitaplıkların ve çerçevelerin mevcudiyeti göz önüne alındığında, önerilen aracın teknik olarak uygulanabilir olduğunu göstermektedir. Uygulamanın, çizge teorisi kavramlarını öğrenmek ve uygulamak için sağlam, ölçeklenebilir ve kullanıcı dostu bir çözüm sunması beklenmektedir.

3.1.2 Donanım Fizibilitesi

Önerilen web tabanlı etkileşimli çizge teorisi aracının donanım fizibilitesi, uygulama bir web sunucusunda barındırılacağından, öncelikle istemci tarafı gereksinimlerine odaklanır. Masaüstü ve mobil cihazlardaki modern web tarayıcıları, karmaşık görselleştirmeleri işleme ve JavaScript kodunu verimli bir şekilde yürütme yeteneğine sahiptir. Sonuç olarak, uygulamaya çok çeşitli cihaz ve donanım yapılandırmalarına sahip kullanıcılar tarafından erişilebilir ve kullanılabilir. Araç, uygulamanın performansını ve yanıt verebilirliğini optimize ederek, son kullanıcılara önemli donanım kısıtlamaları getirmeden çeşitli platformlarda sorunsuz ve kullanıcı dostu bir deneyim sağlayacaktır. Projede çalıştırılan fonksiyonların gerektirdiği işlemci gücü görece fazlasıyla azdır ve önemli bir kısıt oluşturmamaktadır. Kullanılan kişisel bilgisayarın teknik özellikleri Apple M1 Chip, 16 GB RAM.

3.2 Ekonomik Fizibilite

Önerilen web tabanlı etkileşimli çizge teorisi aracının ekonomik fizibilitesi, geliştirme ve bakım maliyetlerini azaltan açık kaynak teknolojileri kullanılarak geliştirilmiştir. Erişilebilir ve kullanıcı dostu platform, bilgisayar bilimi eğitim kalitesini artırırken ve yetenekli bir işgücünü teşvik ederken ve çizge teorisine ve ilgili algoritmalara dayanan endüstrilerde yeniliği teşvik eder. Dolayısıyla kullanıcılara zaman ve kaynak tasarrufu sağlar.

3.3 Yasal Fizibilite

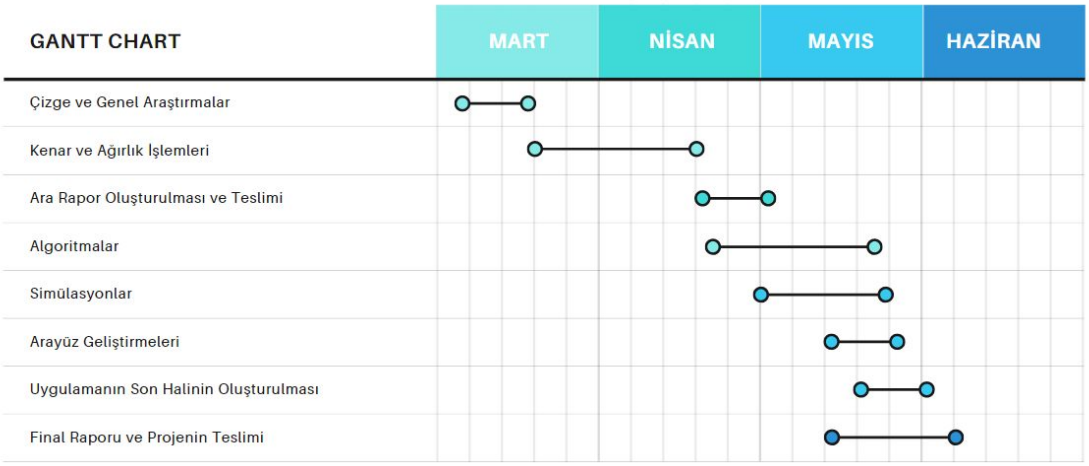
Web tabanlı etkileşimli çizge teorisi aracının yasal fizibilitesi, ilgili yasalara, düzenlemelere ve lisanslama gerekliliklerine uygunluğun sağlanmasını içerir. Açık kaynak teknolojilerini ve kitaplıkları kullanarak, proje potansiyel fikri mülkiyet

çatışmalarını önler. Bu yasal hususların dikkatle değerlendirilmesi, potansiyel risklerin azaltılmasına yardımcı olur ve uygulamanın sorunsuz çalışmasını sağlar.

3.4 İş-Zaman Fizibilitesi

Web tabanlı etkileşimli çizge teorisi aracının iş ve zaman fizibilitesi, verimli proje yönetimine, kaynakların tahsisine ve net bir geliştirme zaman çizelgesine dayanır. Projeyi yönetilebilir görevlere bölerek ve ekip üyelerine sorumluluklar vererek proje verilen zaman çerçevesi içinde tamamlanabilir. Etkili iletişim ve işbirliği ile birlikte iyi yapılandırılmış bir geliştirme süreci, uygulamanın kaliteden ödün vermeden zamanında teslim edilmesini sağlar.

Proje sürecine uygun olarak planlanan ve takip edilen zaman programı **Şekil 3.1**'deki gibidir.



Şekil 3.1 GANTT Diyagramı

4
Sistem Analizi

Bu bölümde; web tabanlı etkileşimli çizge teorisi aracının işlevsel gereksinimleri, çıktı formatı, arayüzü ve veri ile ilgili yönleri ele alınacaktır.

4.1 İşlevsel gereksinimler:

Sistemin işlevsel gereksinimleri, sistemin davranışı ve sistem ile kullanıcıları arasındaki etkileşimleri ele alır. Bu proje için aşağıdaki işlevsel kriterler karşılanmalıdır:

- Düğümler, kenarlar ve görsel iyileştirmeler içeren çizgeler üretmek.
- Düğümler ve kenarlar ekleyerek, silerek veya değiştirerek mevcut çizgelerin düzenlenmesi.
- Oluşturulan çizgeler üzerinde BFS, DFS, Dijkstra, Kruskal ve Prim algoritmaları gibi çizge algoritmalarının kullanımı.
- Çizge algoritmalarını adım adım simüle edilmesi ve simülasyon hızının ayarlanması.
- Çizgelerin JSON biçiminde dışa ve içe aktarılması.

4.1.1 Çıktı Biçimi

Sistemin çıktı formatı, çizgelerin görsel sunumunu içeren bir HTML kanvası olacaktır. Sistem ayrıca çizgeyle ilgili verilerin yanı sıra, en kısa yol veya en az yayılan ağaç gibi verileri de gösterir. Algoritma simülasyonlarının hem metin hem de görsel olarak çıktısı verilir.

4.1.2 Arayüz

4.1.2.1 Girdiler:

Sistem, düğmeler, kaydırıcılar ve metin alanları gibi etkileşimli kullanıcı arayüzü öğeleri aracılığıyla kullanıcıdan gelen girdileri kabul edecektir. Kullanıcı, bu girdileri kullanarak çizgeler oluşturabilir, değiştirebilir ve algoritma simülasyonlarını kontrol edebilir.

4.1.2.2 Çıktılar:

Sistem, görselleştirilmiş çizge ve algoritma sonuçları da dahil olmak üzere çıktıları HTML kanvasında ve web sayfasındaki diğer kullanıcı arayüzü bileşenlerinde görüntüler.

4.1.3 Veri

4.1.3.1 Format:

Veriler hem giriş hem de çıkış için JSON formatında olacaktır. Bu yapı işlevsellik, doğrudan veri gösterimi ve manipülasyonu, içe ve dışa aktarma özelliklerine olanak sağlar.

4.1.3.2 Sıklık:

Veriler, uygulama içindeki kullanıcı eylemlerine göre gerektiği şekilde iletilir ve alınır. Örneğin, kullanıcı bir düğüm eklediğinde çizge verileri buna göre güncellenecektir.

4.1.3.3 Doğruluk:

Verilerin doğruluğu, kullanıcının girdileri ve kullanılan çizge algoritmaları tarafından belirlenir. Sistem, algoritma çıktısının ve çizge verilerinin her ikisinin de uygulanabilir olmasını sağlar. Son derece doğru ve kesin olmasını sağlamalıdır.

4.1.3.4 Veri Akışı:

Kullanıcı uygulama ile etkileşime girdikçe, çizgeler oluştururken, değiştirirken ve algoritmalar çalıştırırken veriler sistem içinde akacaktır.

4.1.3.5 Depolama:

Programın istemci tarafı, durum bilgisi olan bir araç olması amaçlandığından, bilgiler bir kaynakta tutulmaz. Kullanıcı çalışmalarını gelecekte kullanabilir. Çizge verilerini kaydetmek istiyorsa JSON formatında dışa aktarabilir.

4.2 Kullanım Senaryosu

Kullanım Senaryosu, kullanıcı ile web tabanlı etkileşimli çizge teorisi aracı arasındaki etkileşimleri göstermeye yardımcı olacaktır. Bu senaryo, kullanıcının gerçekleştirdiği eylemleri ve sistemin yanıtlarını açıklayarak, uygulamanın işlevselliğini ve akışını net bir şekilde anlaşılır kılar.

İlgililer ve Tanımları:

- **Kullanıcı**: Çeşitli çizge algoritmaları kullanarak çizgeler oluşturarak yazılımı organize etmek ve analiz etmek için yazılımla etkileşime giren kişi.
- **Sistem**: Kullanıcıların çizgelerle çalışması ve algoritma simülasyonları çalıştırması için tasarlanmış işlevsellik sağlayan web tabanlı etkileşimli çizge teorisi aracı.

Ön Koşullar:

Uygulamayı kullanmak için kullanıcının bir web tarayıcısına ve internet bağlantısına ihtiyacı vardır.

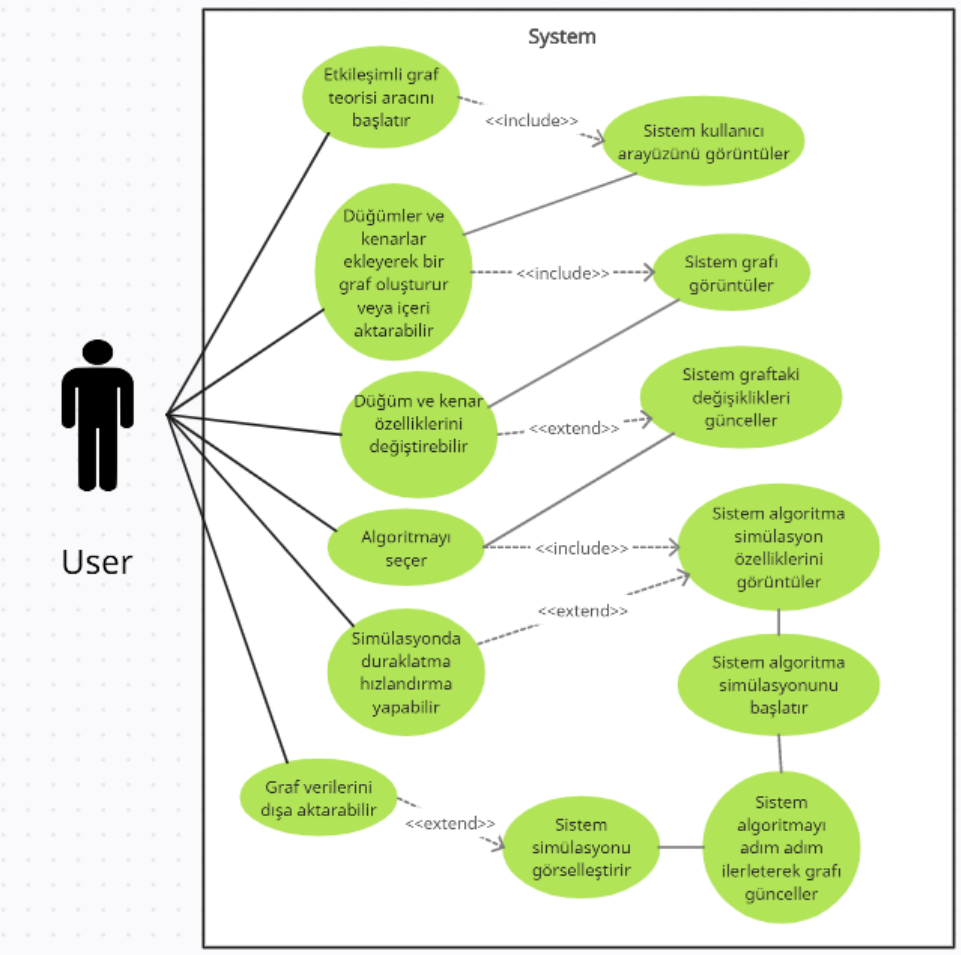
Baş Aktör:

Bu kullanım öyküsünde birincil aktör kullanıcıdır.

Ana Senaryo:

1. Kullanıcı web tarayıcısını başlatır ve etkileşimli çizge teorisi aracını başlatır.
2. Sistem, çizge için tuval, bir araç çubuğu widget'ı ve algoritma kontrollerini içeren kullanıcı arayüzünü sunar.
3. Kullanıcı, araç çubuğunun düğümlerini ve kenarlarını kullanarak bir çizge oluşturur.
4. Sistem, oluşturulan çizgeyi görüntülemek için çizge kanvasını günceller.
5. Kullanıcı, düğümleri ve kenarları silme veya düğüm özelliklerini değiştirme seçeneğine sahiptir.
6. Sistem, kullanıcı arayüzünü bu değişiklikleri yansıtacak şekilde günceller.
7. Algoritma kontrollerini kullanarak, kullanıcı bir çizge algoritması seçer.
8. Sistem, algoritma simülasyonu için hız ayarını ve kontrollerini gösterir.
9. Kullanıcı, algoritmanın simülasyonunu başlatır.
10. Sistem, algoritmanın ilerleyişini görsel olarak temsil etmek için çizgeyi güncelleyerek simülasyonu adım adım çalıştırır.
11. Kullanıcı, sağlanan kontrolleri kullanarak simülasyonu duraklatabilir, devam ettirebilir, hızını ayarlayabilir veya sıfırlayabilir.
12. Yanıt olarak, sistem görselleştirmeyi gerektiği gibi değiştirir.
13. Kullanıcı, ileride kullanmak üzere çizge verilerini JSON biçiminde dışa aktarabilir veya üzerinde çalışmaya devam etmek için önceden kaydedilmiş bir çizgeyi içe aktarabilir.
14. Sistem, içe aktarılan çizgeyi görüntülemek için çizge kanvasını güncelleyerek veya dışa aktarılan çizge verileri için indirilebilir bir dosya sağlayarak içe ve dışa aktarma işlevini yönetir.
15. Kullanıcı, gerektiğinde farklı çizgeler ve algoritmalarla çalışarak 3-9 arasındaki adımları istediği gibi tekrarlayabilir.

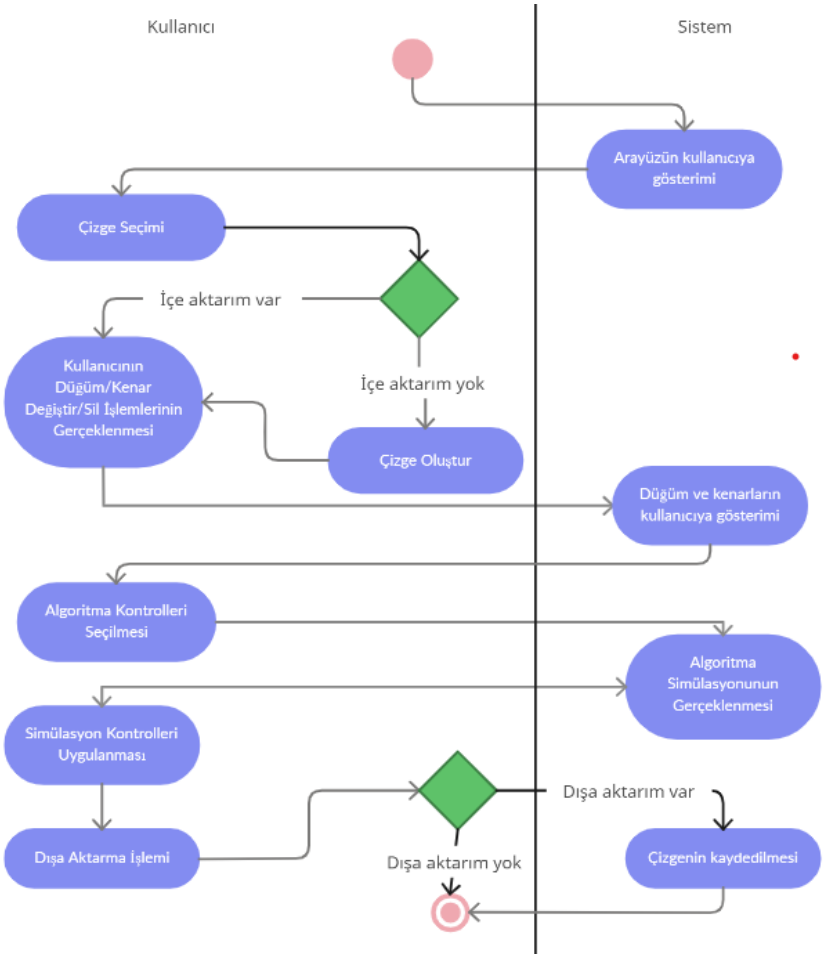
Detaylandırılan kullanım senaryosunun diyagramı **Şekil 4.1**'deki gibidir.



Şekil 4.1 Kullanım Senaryosu Diyagramı

4.3 Aktivite Diyagramı

Projeye uygun olarak kullanıcı ve sistemin işlemlerini gösteren bir aktivite diyagramı hazırlanmıştır. Kullanıcının uygulamayı nasıl kullanacağına ve sistemle nasıl etkileşime geçeceğine dair iş akışını görsel olarak temsil eder. Kullanıcı, uygulamayı kullanarak çeşitli görevleri yerine getirirken, sistem de bu görevlere yanıt verir ve ilgili işlemleri gerçekleştirir. Diyagram **Şekil 4.2**'deki gibidir.



Şekil 4.2 Aktivite Diyagramı

5
Sistem Tasarımı

Web tabanlı etkileşimli çizge teorisi aracı senaryosunun kullanımı, yazılım tasarımı, veri tabanı tasarımı, girdi-çıktı tasarımı ve diğer özellikleri Sistem Tasarımı bölümünde ele alınmaktadır. Bu tasarım, sistemin genel yapısının ve organizasyonunun anlaşılmasına yardımcı olacaktır.

5.1 Yazılım Tasarımı

Sistemin yazılımı, kullanıcı arabirimleri oluşturmak için modüler bir mimari ve iyi bilinen bir JavaScript paketi olan React kullanılarak oluşturulmuştur. Program, GraphCanvas ve AlgorithmControls gibi sistem işlevselliğinin çeşitli yönlerini kontrol eden bölümlerden oluşur. Tasarım, kullanılabilirlik, ölçeklenebilirlik ve sürdürülebilirliğe öncelik verir.

5.1.1 Problem Çözümü / Çözüm Tasarımı

Sistem, çizge yapıları ve algoritmalarla çalışmak için etkileşimli ve kullanıcı dostu bir çözüm sağlamak üzere tasarlanmıştır. Bunu, çeşitli çizge işleme ve algoritma simülasyon işlevlerini destekleyen bir kullanıcı arabirimi sunarak yapar ve aynı zamanda süreç boyunca kullanıcıya görsel geri bildirim sağlar.

5.1.2 Son Kullanıcı (Arayüz) Odaklı Tasarım

Arayüz, son kullanıcı düşünülerek ve deneyimi keyifli kılmaya vurgu yapılarak oluşturulmuştur. Uygulama, çizge oluşturmayı ve düzenlemeyi kolaylaştırır. Erişilebilir kontrolleri kullanma ve algoritmaların simülasyonlarını çalıştırmak ve kontrol etmek için özel bir paneli vardır.

5.1.3 Sistem Şeması

Uygulamanın sistem şeması, kullanıcının hiyerarşisinin yanı sıra birincil bileşenleri (Uygulama, GraphCanvas, Araç Çubuğu ve Algoritma Kontrolleri), bunların uygulama durumu ve kullanıcı girişleriyle etkileşimlerini gösterir.

5.1.4 Algoritmik Yapı

Sistemin algoritmik yapısı, BFS, DFS, Dijkstra, Kruskal ve Prim gibi çizge algoritmalarının uygulamaya entegre edilerek kullanıcıların simülasyonları çalıştırmasına ve çizgeleri analiz etmesine olanak sağlayan çizge algoritmalarının uygulanmasını içerir.

5.2 Veri Tabanı Tasarımı

Etkileşimli web tabanlı çizge teorisi aracı, bir istemci tarafı programı olduğu için bir veritabanı kullanmaz. Bunun yerine, program tarafından, uygulama durumu biçimini alan verilerini işlemek için kullanıcının tarayıcısı kullanılır. Kullanıcının rahatlığı için, çizge verileri kompakt ve uyarlanabilir bir depolama formatı olarak kullanılır. İçe ve dışa aktarmalar JSON dosya yapısıyla yapılır.

5.3 Girdi-Çıktı Tasarımı

Sistemin girdi-çıktı tasarımı sayesinde kullanıcılar programla etkileşime girebilmektedir. Çizgeler, kullanıcıdan gelen girdiler kullanılarak oluşturulabilir, değiştirilebilir ve analiz edilebilir. Uygulama girdileri işler ve sonuçlar görselleştirilerek kanvasta gösterilir.

6
Uygulama

Projenin uygulama aşamasında web tabanlı çizge görselleştirme aracı geliştirilmektedir. Kullanıcı arayüzleri ve web uygulamaları geliştirmek için, Javascript kütüphanesi olan bileşen tabanlı React [15] ile oluşturulmuştur.

Proje için dosya organizasyonu **Şekil 6.1**'deki gibidir.

Şekil 6.1 Dosya Organizasyonu

Bileşenleri, algoritmaları ve diğer uygulama dosyalarını ilgili dizinlerine koyarak, bu dosya organizasyonu projeye pratik bir yapı kazandırır.

Aracın birincil işlevleri, uygulamanın üç ana tasarım öğesi tarafından temsil edilir. Bunların ikisi, GraphCanvas ve AlgorithmControls olmakla birlikte, üçüncü bileşen olan Uygulama bileşeni, diğer bileşenleri düzenlemek ve görmek için kullanılır.

GraphCanvas bileşeni, çizge görselleştirme oluşturmak için bir kanvas görevi

görürken, Araç Çubuğu bileşeni, düğümleri ve kenarları eklemek ve değiştirmek için düğmelere ve kontrollere sahiptir. AlgorithmControls bileşeni, sunulan çizge üzerinde farklı çizge algoritmalarını seçmek ve çalıştırmak için bir dizi düğme ve kontrol sağlar.

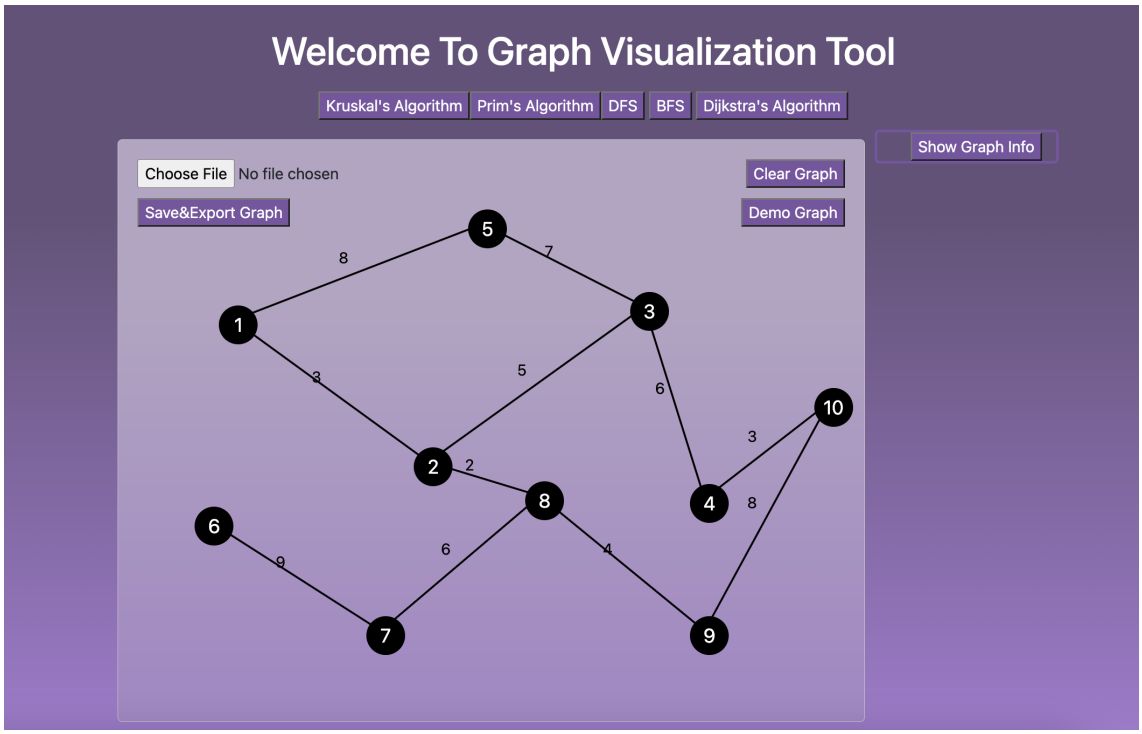
- **App:** Uygulamayı başlatan ve ana düzeni oluşturan temel bileşen.
- **GraphCanvas:** İnteraktif çizge görselleştirmesini görüntülemek ve yönetmekten sorumludur.
- **AlgorithmControls:**Simülasyon için mevcut çizge algoritmalarının seçimini ve kontrolünü ele alır.

Kullanıcı etkileşimlerini ve çizgenin durumlarını kontrol etmek için hooklar geliştirildi. Çizgedeki düğümler ve kenarlar, bu hooklar sayesinde güncellenebilir, silinebilir ve eklenebilir. Ek olarak, kullanıcı girişine yanıt olarak gerçek zamanlı değişiklikleri güncelleyerek çizgeyi mevcut durumunda tutar.

Kruskal, Prim, DFS, BFS ve Dijkstra algoritmaları gibi çizge algoritmalarının uygulanması, ilk temel atıldıktan sonra geliştirme sürecinin ileri adımlarından biridir. Bu algoritmalar programın içine yerleştirilmiştir ve AlgorithmControls bileşeni, kullanıcıların bunları gösterilen arayüzde seçip çalıştırmasını sağlar.

GraphCanvas bileşeni, düğümleri ve kenarları ekleme ve değiştirme ya da kanvas üzerinde yeniden konumlandırma gibi çizgeyle kullanıcı etkileşimini etkinleştirmek için olay dinleyicilerine sahiptir. Bu, uygulamanın mevcut etkileşimli özelliklerini geliştirir.

Kanvas üzerine arayüzde çizilmiş yahut import edilmiş bir çizge örneği **Şekil 6.2**'deki gibidir.

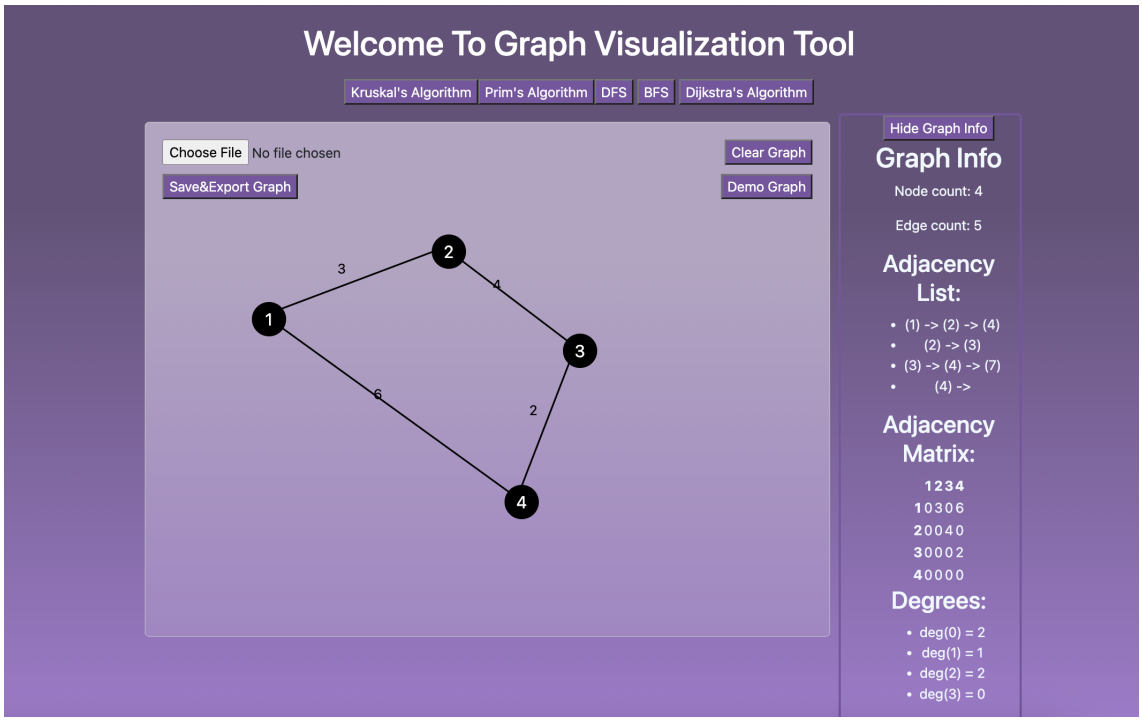


Şekil 6.2 Çizge Örneği

Oluşturulan çizgelerin özniteliklerini görüntülemek için "GraphInfo" bileşeni eklenmiştir. Kullanıcılar çizgeyle etkileşim kurduğunda ve algoritmalar kullandığında, bu modül, düğüm ve kenar sayısı, komşuluk listesi, komşuluk matrisi ve derece bilgileri gibi çizge öznitelikleri üzerindeki verileri dinamik olarak günceller.

Ana uygulama düzeni daha sonra GraphInfo bileşenini içerecek şekilde güncellenerek kullanıcıların aracı kullanırken çizge özelliklerine hızla erişmesini sağlar. Bu güncelleme genel olarak kullanıcı deneyimini geliştirir ve uygulanan algoritmalar ve çizge yapıları hakkında ayrıntılar verir.

Kanvastaki çizge hakkında bilgilerin listelendiği "Graph Info" bileşeni **Şekil 6.3**'deki gibidir.



Şekil 6.3 Graph Info Bileşeni

Program ayrıca kullanıcı dostu ve sezgisel bir arayüz oluşturmak için CSS ile biçimlendirilmiştir. Programın, özellikle önceden çizge teorisi bilgisi olmayan kullanıcılar için kullanımını ve anlaşılmasını kolaylaştırmak için, kullanıcı arayüzüne ve kullanıcı deneyimine odaklanıldı.

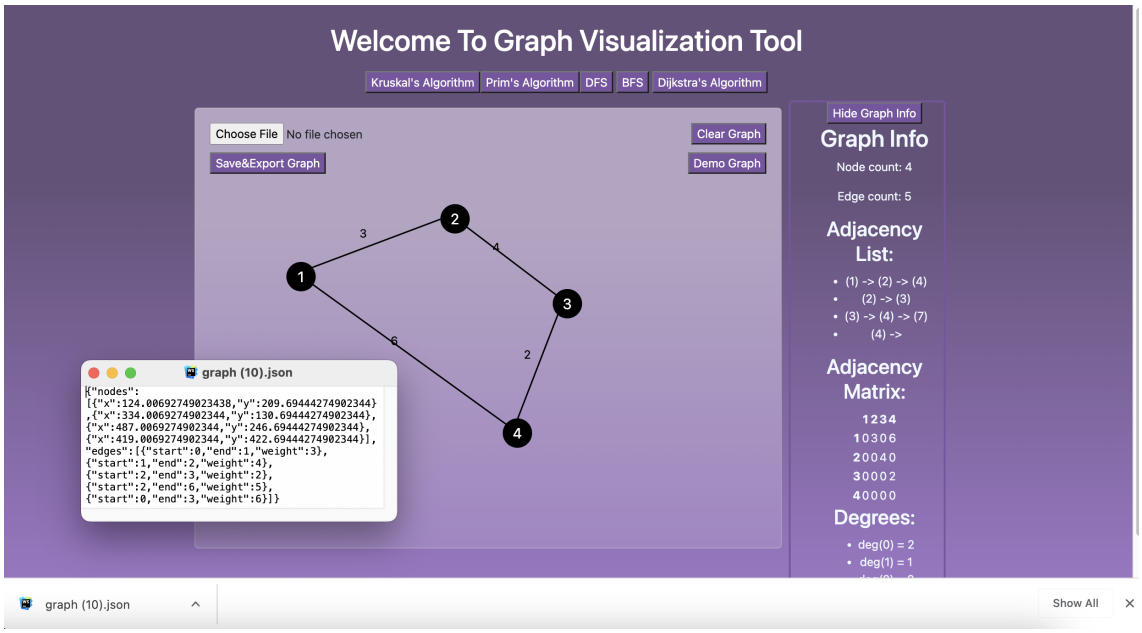
Proje özelliklerine uygun olarak web tabanlı bir çizge görselleştirme aracı oluşturmaya odaklanıldı. İlgi çekici bir deneyim sunmak için uygulama, React ve modüler bileşenler kullanılarak oluşturuldu. Görsel olarak basit bir arayüz geliştirmek için temel işlevler ve çizge algoritmaları kullanılır. Çizge teorisi konseptlerini ve algoritmalarını keşfetmek, gerçeklemek ve anlamak için kullanışlı bir web uygulaması geliştirilmiştir.

7
Deneysel Sonuçlar

Etkileşimli çizge teorisi aracı, kullanıcının hızlı bir şekilde çizge oluşturmasını ve çizge üzerinde çeşitli analizler gerçekleştirmesini sağlar. Bu nedenle uygulamanın diğer fonksiyonlarının test edilmesi gerekir.

Örneğin Import/Export modülünün doğru şekilde sonuç alıp almadığı kontrol edilmiş, dışa aktarılan çizge tekrardan içe aktarıldığında aynı çizgenin oluşup oluşmadığına bakılmıştır.

Başarıyla sonuçlanan bu deneyin ekran görüntüsü **Şekil 7.1**'deki gibidir.



Şekil 7.1 Import/Export Modülü Testi

Mevcut çizgedeki birleştirilmiş düğümlerin silinmesi durumunda çizgenin yeni halinin doğru şekilde olup olmayacağının deneyi yapılmış ve bu kontrol de eksiksiz ve hatasız şekilde yerine getirilmiştir. Algoritmaların çalıştırılması sırasında da farklı düğüm-kenar olasılıklarındaki sonuç tutarlıkları da kontrol edilmiş, uç durumlar için gerekli kontroller yapılmıştır.

8
Performans Analizi

Bu projenin amacı, çizge verileri üzerinde farklı algoritmaları etkili bir şekilde görselleştirmektir. Performans analizi, bu algoritmaların uygulanmasının ve görselleştirilmesinin etkinliğini ve hızını değerlendirir. Ek olarak, uygulamanın ölçeklenebilirliğine ve genel kullanıcı deneyimine bakar.

8.1 Algoritma Performansı

Projede DFS, BFS ve Kruskal algoritmaları esastır. Uygulamanın toplam performansı, bu algoritmaların hızından ve etkinliğinden doğrudan etkilenir. Bu, düğümlerin ve kenarların sayısı arttığında, algoritmaların çalışma sürelerinin de aynı şekilde doğrusal olarak artacağını gösterir.

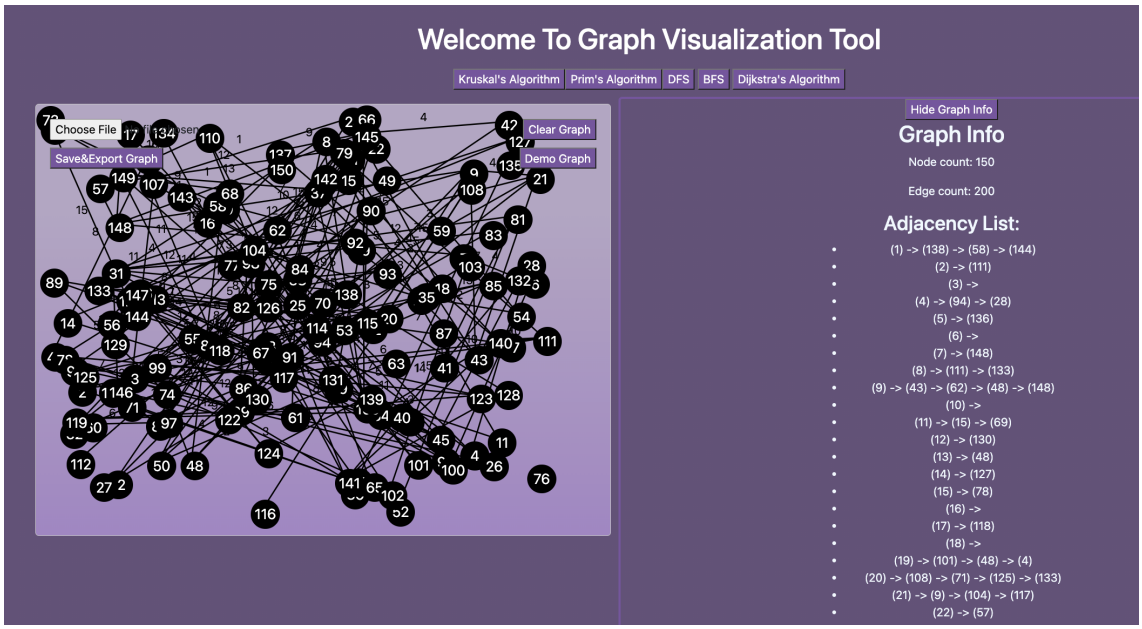
8.2 Kullanıcı Deneyimi ve Yanıt Süresi

Performans, kullanıcı arabiriminden ve uygulamanın yanıt süresinden büyük ölçüde etkilenir. Uygulama, her bir algoritmik adımı gerçek zamanlı olarak görselleştirdiğinden, kullanıcı girişine anında tepki vermelidir. Uygulama, React ve Redux ile kurulur ve kullanıcı eylemlerine anında tepki verecek şekilde tasarlanmıştır. Azaltılmış sistem kaynağı tüketimi elde etmek için hafif bir tasarım kullanılır.

8.3 Ölçeklenebilirlik

Bir uygulamanın artan taleplere nasıl uyum sağlayacağı, ölçeklenebilirliğine bağlıdır. Ölçeklenebilirlik, bu projenin tasarımında dikkate alınan bir konuydu. Büyük çizge verileriyle çalışırken, uygulamanın performansını yavaşlatabilecek çeşitli kısıtlamalar vardır. Özellikle, ek düğümler ve kenarlar eklemek, algoritmaların yürütme süresini uzatabilir ve daha fazla sistem kaynağı tüketebilir.

150 düğüm 200 kenarlı bir çizge **Şekil 8.1**'deki gibidir.



Şekil 8.1 Performans Analizi

Görüldüğü üzere 150 düğüm 200 kenarlı bir algoritmayı tasarladığımız sistem sorunsuz ve akıcı bir şekilde oluşturmaktadır.

9
Sonuç

Çizge teorisi ve algoritmaları günümüzde birçok alanda kullanılmaktadır ve veri yapısı nitelikleri kullanılarak somut bir şekilde gösterilebilir. Bu özellik, öğrenme aşamasında çizge teorisinin görsel olarak analiz edilmesine ve çalışılmasına olanak tanır. Bu nedenle web tabanlı interaktif bir çizge teorisi aracı, özellikle veri yapılarını inceleyen ve algoritma geliştiren kullanıcılar için çok uygundur. Kullanıcılar uygun niteliklere sahip bir çizgeyi hızlı bir şekilde oluşturabilir veya kendi çizgelerini tasarlayabilirler. Kullanıcı deneyimi ve çizgedeki yapısal değişiklikler pratikte iyi çalışır. Çizge ile etkileşim derecesi ve yeniden düzenlenebilirlik açısından avantajları, mevcut örnekler analiz edildiğinde bu aracı rekabette bir adım öne çıkarmaktadır.

Algoritmalardan birini seçerek, etkileşimli çizge teorisi aracının algoritma simülasyon modülündeki aşamaları simüle edebilirsiniz. Özellikle çizge teorisine odaklanan bu program, çeşitli konulara genişletilerek bilgisayar bilimleri ve veri yapıları için interaktif bir öğretim aracına dönüştürülebilir. Kullanıcı katılımı artırılarak ve çizge özelliklerinin düzenlenebilir ve ayarlanabilir seçenekleri genişletilerek ek algoritmalar ve simülasyonlar içeren daha kapsamlı bir araç üretilebilir. Bu temele dayanan ve yukarıda bahsedilen yetenek ve modülleri içeren araç, projeyi desteklemek amacıyla gelecekte geliştirilmeye açıktır.

Referanslar

[1] F. Riaz and K. M. Ali, "Applications of graph theory in computer science," in *2011 Third International Conference on Computational Intelligence, Communication Systems and Networks*, 2011, pp. 142–145. DOI: 10.1109/CICSyN.2011.40.

[2] J. B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," *Proceedings of the American Mathematical society*, vol. 7, no. 1, pp. 48–50, 1956.

[3] J. Medak, "Review and analysis of minimum spanning tree using prim's algorithm," *International Journal of Computer Science Trends and Technology (IJCST)–Volume*, vol. 6, 2018.

[4] Wikipedia. "Depth-first search." (2023), [Online]. Available: https://www.wikiwand.com/en/Depth-first_search.

[5] B. Hareketi, "Bölüm 6: Genişlik öncelikli arama (breadth first search – bfs)," *Medium*, 2019. [Online]. Available: https://medium.com/bili%C5%9Fim-hareketi/b%C3%B6l%C3%BCm-6-geni%C5%9Flik-%C3%B6ncelikli-arama-breadth-first-search-bfs-49476484c07b.

[6] B. Kavramları. "Dijkstra algoritması." (2010), [Online]. Available: https://bilgisayarkavramlari.com/2010/05/13/dijkstra-algoritmasi-2/.

[7] M. Bastian, S. Heymann, and M. Jacomy, "Gephi: An open source software for exploring and manipulating networks," *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 3, no. 1, pp. 361–362, 2009. DOI: 10.1609/icwsm.v3i1.13937. [Online]. Available: https://ojs.aaai.org/index.php/ICWSM/article/view/13937.

[8] P. Shannon *et al.*, "Cytoscape: A software environment for integrated models of biomolecular interaction networks," *Genome research*, vol. 13, no. 11, pp. 2498–2504, 2003. DOI: 10.1101/gr.1239303. [Online]. Available: https://doi.org/10.1101/gr.1239303.

[9] J. Ellson, E. Gansner, L. Koutsofios, S. C. North, and G. Woodhull, "Graphviz—open source graph drawing tools," in *Graph Drawing*, P. Mutzel, M. J"unger, and S. Leipert, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 483–484, ISBN: 978-3-540-45848-7.

[10] M. Bostock, V. Ogievetsky, and J. Heer. "D3: Data-driven documents." (2011), [Online]. Available: https://d3js.org/.

[11] J. Coene. "Sigmajs." (2014), [Online]. Available: https://www.sigmajs.org/.

[12] M. A. Rostami, A. Azadi, and M. Seydi, “Graphtea: Interactive graph self-teaching tool,” *Communications, Circuits and Educational Technologies*, Jan. 2014.

[13] T. Peixoto. “The graph-tool python library.” (Jan. 2014).

[14] Y. Pigné, A. Dutot, F. Guinand, and D. Olivier, *Graphstream: A tool for bridging the gap between complex systems and dynamic graphs*, 2008. arXiv: 0803.2093 [cs.MS].

[15] Facebook. “React.” (2013), [Online]. Available: https://legacy.reactjs.org/.

Özgeçmiş

BİRİNCİ ÜYE

İsim-Soyisim: Bedrettin Şamil ÖZTÜRK
Doğum Tarihi ve Yeri: 04.10.1999, İstanbul
E-mail: samil.ozturk@std.yildiz.edu.tr
Telefon: 0536 566 00 00
Staj Tecrübeleri: TÜBİTAK

İKİNCİ ÜYE

İsim-Soyisim: Alper EREN
Doğum Tarihi ve Yeri: 13.06.2001, Bursa
E-mail: alper.eren@std.yildiz.edu.tr
Telefon: 0534 075 9892
Staj Tecrübeleri:

Proje Sistem Bilgileri

Sistem ve Yazılım: Windows/MacOS İşletim Sistemi, JavaScript
Gerekli RAM: 2GB
Gerekli Disk: 256MB