Bilgisayar Uygulamalarında Çizge Kuramı Projesi

Emre Yıldız

25 Ocak 2025

Özet

Bu projede, çizge kuramının temel kavramlarını Python programlama dili kullanarak uyguladık. Proje, çizgelerin oluşturulması, analizi ve manipülasyonu için gerekli temel işlevleri içermektedir. Özellikle, tam çizge, döngü çizge, çark çizge ve n-küp çizge gibi özel çizge türlerinin oluşturulması ve analizi için gerekli algoritmalar implement edilmiştir. Projede herhangi bir harici kütüphane kullanılmamıştır. Tüm algoritmalar, veri yapıları ve sınıflar sıfırdan geliştirilmiştir.

İçindekiler

1 Giriş

Çizge kuramı, matematiğin önemli alanlarından biridir ve bilgisayar bilimlerinde geniş uygulama alanına sahiptir. Bu projede, çizge kuramının temel kavramlarını Python programlama dilinde uygulayarak, çizgelerin oluşturulması, analizi ve manipülasyonu için bir yazılım kütüphanesi geliştirdik.

2 Tasarım ve Implementasyon

2.1 Veri Yapıları

Projede iki temel veri yapısı kullanılmıştır:

2.1.1 Tepe (Vertex) Sınıfı

Tepe sınıfı, çizgedeki her bir düğümü temsil eder:

- `label`: Tepenin etiketi (string)
- `neighbors`: Komşu tepelerin kümesi (Set[str])

Python'un `dataclass` özelliği kullanılarak implement edilmiştir. Bu sayede sınıf otomatik olarak `__init__`, `__repr__` ve `__eq__` metodlarına sahip olur.

2.1.2 Çizge (Graph) Sınıfı

Ana çizge sınıfı şu özelliklere sahiptir:

- `vertices`: Tepeler sözlüğü (Dict[str, Vertex])
- `input_type`: Giriş formatı (Matrix/List)

3 Algoritmalar ve Analiz Metodları

3.1 Çizge Türü Belirleme

`determine_graph_type` metodu, bir çizgenin özel türlerden biri olup olmadığını kontrol eder:

Algorithm 1 Çizge Türü Belirleme

Input: Graph G
Output: (is_complete, is_cycle, is_wheel, is_hypercube)

// Tam çizge kontrolü
if her tepenin derecesi (n-1) ise **then**
  is_complete= true
end if

// Döngü çizge kontrolü
if n $\geq$ 3 ve her tepenin derecesi 2 ve çizge bağlı ise **then**
  is_cycle= true
end if

// Çark çizge kontrolü
if n $\geq$ 4 **then**
  merkez= derece(n-1) olan tepe
  if merkez varsa ve kalan tepeler döngü oluşturuyorsa **then**
    is_wheel= true
  end if
end if

// n-küp çizge kontrolü
if n= 2^k ve her tepenin derecesi k ise **then**
  is_hypercube= true
end if

3.2 İki Parçalı Çizge Kontrolü

İki parçalı çizge kontrolü için BFS tabanlı bir renklendirme algoritması kullanılmıştır:

Algorithm 2 İki Parçalı Çizge Kontrolü

```
Input: Graph G
Output: (is_bipartite, (set1, set2))

colors = {} // Tepe → renk (0 veya 1)
for her başlangıç tepesi v do
  if v renklendirilmemişse then
    queue = [(v, 0)]
    while queue boş değil do
      vertex, color = queue.pop()
      if vertex renklendirilmişse then
        if rengi farklıysa then
          return (false, None)
        end if
        continue
      end if
      vertex'i color ile renklendir
      komşuları karşıt renkle queue'ya ekle
    end while
  end if
end for
set0 = rengi 0 olan tepeler
set1 = rengi 1 olan tepeler
return (true, (set0, set1))
```

3.3 Bağlı Bileşenler

Çizgenin bağlı bileşenlerini bulmak için BFS algoritması kullanılmıştır:

Algorithm 3 Bağlı Bileşenleri Bulma

Input: Graph G
Output: List[Set[str]] (bağlı bileşenler listesi)

```
unvisited = tüm tepeler
components = []
while unvisited boş değil do
  start = unvisited'dan bir tepe seç
  component = {}
  queue = [start]
  while queue boş değil do
    vertex = queue.pop()
    if vertex ziyaret edilmemişse then
      component'e ekle
      unvisited'dan çıkar
      ziyaret edilmemiş komşuları queue'ya ekle
    end if
  end while
  components'e component'i ekle
end while
return components
```

4 Performans Analizi

Algoritmaların karmaşıklık analizi:

- **Çizge Türü Belirleme**: O(V + E)
- **İki Parçalı Kontrol**: O(V + E)
- **Bağlı Bileşenler**: O(V + E)
- **Tam İki Parçalı Kontrol**: O(V^2)

Burada V tepe sayısını, E kenar sayısını göstermektedir.

5 Test ve Doğrulama

Proje kapsamlı bir test paketi içermektedir. unittest modülü kullanılarak yazılan testler şunları içerir:

- **Özel Çizge Testleri**:
  - Tam çizge (Kn) oluşturma ve doğrulama
  - Döngü çizge (Cn) oluşturma ve doğrulama
  - Çark çizge (Wn) oluşturma ve doğrulama
  - n-küp çizge (Qn) oluşturma ve doğrulama
- **Çizge Özellikleri Testleri**:
  - Soyutlanmış tepe bulma
  - Asılı tepe bulma
  - Bağlılık kontrolü
  - İki parçalı çizge kontrolü
- **Dosya İşlemleri Testleri**:
  - Matris formatı okuma/yazma
  - Liste formatı okuma/yazma

6 Sonuç

Bu projede, çizge kuramının temel kavramları Python programlama dili kullanılarak başarıyla uygulanmıştır. Özellikle:

- Farklı çizge türlerinin oluşturulması ve analizi
- Verimli veri yapıları ve algoritmalar
- Kapsamlı test ve doğrulama
- Detaylı dokümantasyon

başarıyla gerçekleştirilmiştir. Proje, çizge kuramı konusunda eğitim ve araştırma amaçlı kullanılabilir.

A Kaynak Kodları

Projenin tam kaynak kodları ayrı bir dokümanda (code_appendix.tex) verilmiştir.