Bölüm 11: Çizge

Veri Yapıları

Çizge

- Noktaların (düğümler) ve bu noktaları birbirine bağlayan kenarların (yollar) bir araya gelmesiyle oluşturulan bir veri yapısıdır.
- Karmaşık ilişkileri, ağları ve yapıları temsil etmek için kullanılır.

Nokta (Düğüm)

- Bir çizgenin temel birimleridir.
- Genellikle şehirler, kişiler, bilgisayarlar gibi şeyleri temsil eder.
- Örnek:
  - Bir şehirler ağındaki şehirler.

Kenar (Yol)

- İki düğümü birbirine bağlayan bağlantıdır.
- Kenarlar düğümler arasındaki ilişkiyi temsil eder.
- Örnek:
  - Şehirler arasındaki yol veya internet üzerindeki bağlantı.

Derece (Degree)

- Bir düğümün sahip olduğu kenar sayısını ifade eder.
- Derecesiz (0 derece) düğümler de olabilir.
- Örnek:
  - Bir kavşaktaki yolların sayısı.

Yönlü Çizge

- Kenarlar, bir düğümden diğerine sadece belirli bir yönde gider.
- İkinci düğüme geri dönüş yolu olmayabilir.
- Örnek:
  - İnternet sayfaları ve bağlantıları.

Döngü (Cycle)

- Bir düğümün başlangıç noktasına geri döndüğü bir yol veya dizi.
- Döngüler çizge içerisinde olabilir.
- Örnek:
  - Bir uçuş rotası, nereden başladıysanız oraya geri dönme.

Kullanım Alanları

- Haritalar, sosyal ağlar, ulaşım ağları, çevrimiçi alışveriş, siber güvenlik, dil analizi ve daha birçok alanda kullanılır.
- Bilgisayar biliminde ve mühendislikte karmaşık yapıları modellemek için kullanılır.

Gezinme Algoritmaları

- Genişlik Öncelikli Arama (BFS)
- Derinlik Öncelikli Arama (DFS)

Genişlik Öncelikli Arama (BFS)

- Genişlik İlk Arama (BFS), bir graf veya ağaç yapısındaki düğümleri seviye seviye gezerek keşfeden bir arama algoritmasıdır.
- En yakın komşuları önce ziyaret eder ve giderek daha uzak düğümlere ilerler.

Genişlik İlk Arama Adımları

- Başlangıç düğümünü işaretle ve kuyruğa ekleyin.
- Kuyruktan bir düğümü çıkarın, ziyaret edin ve komşularını kuyruğa ekleyin.
- Kuyruğun başına geri dönün ve işlemi tekrarlayın.
- Kuyruk boşalana kadar devam edin.

Genişlik İlk Aramanın Özellikleri

- Kuyruk yapısı kullanılır.
- Her düğüm sadece bir kez ziyaret edilir.
- Başlangıç düğümünden başlayarak uzaklığa göre sıralı gezinir.

Avantajlar ve Dezavantajlar

- **Avantajlar**
  - En kısa yolu bulma konusunda etkilidir.
  - En yakın düğümleri hızla keşfeder.
  - Mantıklı bir sıralama sağlar.
- **Dezavantajlar**
  - Bellek kullanımı yüksek olabilir.
  - En kısa yol bulma problemi olmayan durumlarda zaman kaybına neden olabilir.

Derinlik Öncelikli Arama (DFS)

- Derinlik İlk Arama (DFS), bir graf veya ağaç yapısındaki düğümleri bir yol boyunca keşfeden bir arama algoritmasıdır.
- Derinlik önceliği olan bir yaklaşımı temsil eder.

Derinlik İlk Arama Adımları

- Başlangıç düğümünü işaretle ve ziyaret et.
- Başlangıç düğümünün bir komşusu varsa, bir komşuyu seç ve tekrarlayın.
- İlk düğümün tüm komşularını ziyaret ettikten sonra geriye dönün ve ziyaret edilmemiş komşuları arayın.
- Başlangıç düğümünün tüm komşularını ziyaret edene kadar bu işlemi tekrarlayın.

Derinlik İlk Aramanın Özellikleri

- Yığın (stack) veya rekürsif çağrılar kullanılabilir.
- Her düğüm sadece bir kez ziyaret edilir.
- Derinlemesine keşfeder ve daha sonra yüzey düğümlere ilerler.

Avantajlar ve Dezavantajlar

- Avantajlar
  - Hafıza kullanımı genellikle düşüktür.
  - Bir yolun sonuna kadar gitme yeteneği sağlar.
  - Yüzey düğümlerine hızla ulaşabilir.
- Dezavantajlar
  - En kısa yol problemi için etkili değildir.
  - Bellek taşması riski vardır.

Dijkstra Algoritması

- Dijkstra Algoritması, tek bir başlangıç düğümünden diğer düğümlere olan en kısa yolunuzu bulmak için kullanılan bir yol arama algoritmasıdır.
- Ağırlıklı grafiklerde kullanılır ve pozitif kenar ağırlıkları gerektirir.

Dijkstra Algoritması Adımları

- Başlangıç düğümünü seçin ve onu işaretleyin.
- Başlangıç düğümüne olan tüm yolların maliyetini sonsuz (∞) olarak ayarlayın ve kendisine olan maliyeti 0 olarak ayarlayın.
- Henüz işaretlenmemiş en düşük maliyetli düğümü seçin.
- Seçilen düğümü işaretleyin ve komşularını inceleyin.
- Başlangıç düğümünden bu düğüme olan maliyet ile bu düğümden komşu düğüme olan maliyeti toplayın.
- Eğer bu toplam maliyet, mevcut komşu düğümün mevcut maliyetinden daha düşükse, maliyeti güncelleyin.
- İşareti kaldırılan düğümü göz ardı edin ve geriye dönün.
- Hedef düğüme ulaşıncaya kadar 3-7 adımlarını tekrarlayın.

Dijkstra Algoritmasının Özellikleri

- İkinci en kısa yolu bulma yeteneğine sahiptir.
- Ağırlıklı grafiklerde kullanılır.
- Negatif ağırlıkları olan kenarlarda çalışmaz.

Dijkstra Algoritması Kullanım Alanları

- Harita yönlendirme uygulamaları.
- Ağ yönlendirme ve trafiği optimize etme.
- Bilgisayar ağlarının en kısa yol hesaplamaları.

Avantaj ve Dezavantajlar

- Avantajlar
  - En kısa yolu etkili bir şekilde hesaplar.
  - En iyi yolu bulma garantisi verir.
- Dezavantajlar
  - Negatif ağırlıklarla çalışmaz.
  - Birden fazla hedef düğüm varsa, her biri için hesaplama gerekebilir.

Floyd-Warshall Algoritması

- Floyd-Warshall Algoritması, çoklu düğümlü bir grafikteki her iki düğüm arasındaki en kısa yolu hesaplamak için kullanılan bir yol arama algoritmasıdır.
- Pozitif veya negatif kenar ağırlıkları olan grafiklerde çalışabilir.
- Tüm çift düğüm mesafelerini hesaplar.

Floyd-Warshall Algoritması Adımları

- Birinci adımda, her düğüm arasındaki mesafeleri ve düğümleri belirlemek için bir matris oluşturulur.
- İkinci adımda, her düğüm arasındaki en kısa yolu güncellemek için üçlü bir döngü oluşturulur.
- Her üçlü döngü iterasyonunda, daha kısa bir yol bulunursa matris güncellenir.
- Son adımda, en kısa yollar matrisi elde edilir.

Avantaj ve Dezavantajlar

- Avantajlar
  - Tüm çift düğüm mesafelerini hesaplar.
  - Negatif kenar ağırlıklarıyla çalışabilir.
- Dezavantajlar
  - Büyük veri kümeleri için hesaplama karmaşıklığı yüksek olabilir.
  - Diğer yöntemlere göre daha fazla hafıza kullanabilir.

Topolojik Sıralama

- Topolojik Sıralama Algoritması, bir yönlendirilmiş asiklik grafik içindeki düğümleri bağımlılıklarına göre sıralayan bir algoritmadır.
- Yönlendirilmiş asiklik grafikler (DAG) üzerinde çalışır ve iş akışları, sıralama sorunları ve proje yönetimi için kullanılır.
- Avantajları bağımlılıkları netleştirmesi ve güvenilir sıralamalar sunmasıdır.

Topolojik Sıralama Algoritması Adımları

- İşlenmemiş düğümleri içeren bir veri yapısı (genellikle yığın veya kuyruk) oluşturun.
- Başlangıç düğümünü alın ve işlenmemiş düğümler arasında bağımlılığı olmayan düğümleri işleyin ve sıralamaya ekleyin.
- İşlenen düğümleri grafikten kaldırın.
- İşlenmemiş düğümler kalmayana kadar 2-3 adımlarını tekrarlayın.

Topolojik Sıralamanın Özellikleri

- Yönlendirilmiş asiklik grafikler (DAG) üzerinde çalışır.
- Düğümleri bağımlılıklarına göre sıralar.
- Düğümler arasında döngü içermez.

Avantaj ve Dezavantajlar

- Avantajlar
  - Bağımlılıkları ve iş akışlarını anlamada etkilidir.
  - Döngülerin olmaması, tutarlı ve güvenilir sıralamalar sağlar.
- Dezavantajlar
  - Yönlendirilmiş asiklik grafik (DAG) gerekliliği, bazı senaryolarda sınırlayıcı olabilir.
  - Karmaşıklığı, grafik boyutuna bağlı olarak artabilir.

Prim-Jarník Algoritması

- Prim-Jarník Algoritması, bir ağırlıklı bağlantı grafiğindeki en küçük ağaçları (minimum ağaç) bulmak için kullanılan bir yol arama algoritmasıdır.
- Minimum ağaç, tüm düğümleri bağlayan ve toplam ağırlığı minimum olan bir alt grafik olarak tanımlanır.

Prim-Jarník Algoritması Adımları

- Başlangıç düğümünü seçin ve minimum ağaca ekleyin.
- Minimum ağacın henüz eklenmemiş düğümleri arasında, minimum ağırlıklı bir kenarı seçin ve bu kenarı minimum ağaca ekleyin.
- İkinci adımı minimum ağacın düğümlerini kapsayacak şekilde tekrarlayın.
- Minimum ağaç tamamlandığında işlemi sonlandırın.

Prim-Jarník Algoritmasının Özellikleri

- Minimum ağacı bulmak için kullanılır.
- Yönlendirilmiş veya yönlendirilmemiş grafiklerde işleyebilir.
- Her adımda en düşük ağırlıklı kenarı seçer.

Avantaj ve Dezavantajlar

- Avantajlar
  - Minimum ağacı bulma garantisi verir.
  - Minimum ağırlıklı kenarları hızla seçer.
- Dezavantajlar
  - Büyük grafiklerde hesaplama karmaşıklığı yüksek olabilir.
  - Grafikte negatif kenar ağırlıkları varsa yanlış sonuçlar üretebilir.

Kruskal Algoritması

- Kruskal Algoritması, bir ağırlıklı bağlantı grafiğindeki minimum ağaçları (minimum kesici ağaç) bulmak için kullanılan bir yol arama algoritmasıdır.
- Minimum kesici ağaç, tüm düğümleri bağlayan ve toplam ağırlığı minimum olan bir alt grafik olarak tanımlanır.

Kruskal Algoritması Adımları

- Tüm kenarları ağırlıklarına göre sıralayın.
- Başlangıçta minimum kesici ağaç boş olacak şekilde bir ağaç oluşturun.
- Sıralanmış kenarları tek tek inceleyin ve her kenarı minimum kesici ağaca ekleyin.
- Kenarı eklerken, döngü oluşturmamak için hedef düğümünün minimum kesici ağaçta olup olmadığını kontrol edin.
- Tüm düğümler minimum kesici ağaca eklendiğinde işlemi sonlandırın.

Kruskal Algoritmasının Özellikleri

- Minimum kesici ağacı bulmak için kullanılır.
- Yönlendirilmiş veya yönlendirilmemiş grafiklerde işleyebilir.
- Her adımda ağırlığı en düşük olan kenarı seçer.

Avantaj ve Dezavantajlar

- Avantajlar
  - Minimum kesici ağacı bulma garantisi verir.
  - Düğümleri hızla birleştirir ve ağırlığı en düşük kenarları seçer.
- Dezavantajlar
  - Büyük grafiklerde hesaplama karmaşıklığı yüksek olabilir.
  - Grafikte negatif kenar ağırlıkları varsa yanlış sonuçlar üretebilir.

SON