ANKARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

KARGO DAĞITIMLARINDA ROTA PLANLAMA VE OPTİMİZASYONU

Melih GÖÇER

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

ANKARA
2023

ÖZET

Yüksek Lisans Tezi

KARGO DAĞITIMLARINDA ROTA PLANLAMA VE OPTİMİZASYONU

Melih GÖÇER

Ankara Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Semra GÜNDÜÇ

Topluluk Tespiti ve bir topluluktaki en kısa mesafeyi bulmak yıllardır bilgisayar bilimi, biyoloji, kimya, coğrafya gibi birçok farklı disiplinde belirli avantajlar sağlayabileceği için en göze çarpan konulardan biri olmuştur. Günümüzde lojistik sektörünün de hızlı bir şekilde büyümesi ve hayatımızda önemli bir rol oynaması nedeniyle ürün teslimatlarının en hızlı ve etkili bir şekilde gerçekleştirilmesi şirketler tarafından ulaşılmak istenen en önemli konulardan biridir. Daha önce literatürde çalışılan Gezgin Satıcı Problemi bunun en çok bilinen örneği olarak gösterilebilir. Tez çalışması kapsamında literatürde daha önce bu problem üzerine yapılan çalışmalar incelenmiştir. Louvain algoritmasından faydalanılarak oluşturulan topluluklara en kısa yol bulma algoritmaları uygulanmıştır. Buna ek olarak oluşan toplulukların da kendi aralarındaki en kısa mesafe de en kısa yol bulma algoritmaları ile hesaplanmış ve sonuçlar kıyaslanmıştır. Böl ve Fethet yöntemi olarak bilinen bu uygulama ile çizgenin topluluk tespiti yapılmadan önceki hali ile topluluk tespiti yapıldıktan sonraki hali arasında algoritmaların performanslarının iyileştiği gözlemlenmiştir. Bu metot ile optimum sonuca ulaşma performansının arttığı görülmüştür.

Ekim 2023, 52 sayfa

Anahtar Kelimeler: Topluluk Tespiti, Optimizasyon Problemleri, En Kısa Yol Bulma, Çizge Teorisi

ABSTRACT

Master Thesis

ROUTE PLANNING AND OPTIMIZATION FOR CARGO DELIVERY

Melih GÖÇER

Ankara University
Graduate School of Natural and Applied Science
Department of Computer Engineering

Supervisor: Prof. Dr. Semra GÜNDÜÇ

Community Detection and finding the shortest distance in a community has been one of the hottest topics for years as it can provide certain advantages in many different disciplines such as computer science, biology, chemistry, geography. Today, due to the rapid growth of the logistics industry and the fact that it plays an important role in our lives, the fastest and most effective delivery of products is one of the most important issues that companies want to achieve. The Traveling Salesman Problem, which has been studied in the literature before, can be shown as the most well-known example of this. Within the scope of the thesis study, previous studies on this problem were examined in the literature. Shortest path finding algorithms were applied to the communities created by using the Louvain algorithm. In addition, the shortest distance between the formed communities was calculated with the shortest path finding algorithms and the results were compared. With this application, known as the Divide and Conquer method, it was observed that the performance of the algorithms improved between the state of the graph before the ensemble detection and the state after the ensemble detection. It has been observed that the performance of reaching the optimum result increases with this method.

October 2023, 52 pages

Key Words: Community Detection, Optimization Problems, Finding The Shortest Path, Graph Theory

ÖNSÖZ ve TEŞEKKÜR

Yüksek lisans tez çalışmalarım süresince değerli zamanını benden esirgemeyen, bilgi ve tecrübesi ile her konuda bana yol gösteren ve tez sürecim boyunca beni sürekli motive eden değerli danışmanım Sayın Prof. Dr. Semra GÜNDÜÇ'e (Ankara Üniversitesi Bilgisayar Mühendisliği Anabilim Dalı);

Bugünlere gelmemde en büyük pay sahibi olan sevgili anneme, babama;

Varlıklarıyla bana büyük moral kaynağı olan sevgili arkadaşlarım Osman TAŞDELEN, Şenol BAŞAR ve Yunus Emre ÇİLOĞLU'na teşekkürü borç bilirim.

Melih GÖÇER
Ankara, Ekim 2023

İÇİNDEKİLER

TEZ ONAY SAYFASI
ETİK ..... **i**
ÖZET ..... **ii**
ABSTRACT ..... **iii**
ÖNSÖZ ve TEŞEKKÜR ..... **iv**
SİMGELER DİZİNİ ..... **vii**
ŞEKİLLER DİZİNİ ..... **viii**
ÇİZELGELER DİZİNİ ..... **ix**
1. GİRİŞ **1**
1.1 Tezin Amacı **3**
1.2 Çizge Kuramı **4**
1.3 Literatürdeki Çalışmalar **6**
1.3.1 Sosyal ve biyolojik ağlarda topluluk yapısı **6**
1.3.2 Ağlarda topluluk yapısını bulma ve değerlendirme **7**
1.3.3 Çizgelerde topluluk tespiti **7**
1.3.4 Çok büyük ağlarda topluluk yapısı bulma **8**
2. ÇİZGE ÇEŞİTLERİ VE ÇİZGE TEORİSİ UYGULAMALARI **9**
2.1 Çizge Çeşitleri **9**
2.1.1 Yönlü çizge **10**
2.1.2 Yönsüz çizge **10**
2.1.3 Karışık çizge **11**
2.1.4 Ağırlıklı çizge **11**
2.1.5 Düzenli çizge **11**
2.1.6 Tam çizge **12**
2.2 Teknolojide Çizge Teorisi Uygulamaları **13**
2.2.1 Veri madenciliğinde çizge teorisi **14**
2.2.2 Ağ tasarımı **14**
2.2.3 Dil işleme **14**
2.2.4 Kod çözümleme **15**
2.2.5 Elektronik çip tasarımı **15**
2.2.6 Kimya **15**
2.2.7 Biyoloji **16**
2.2.8 Coğrafya **16**
3. EN KISA YOL BULMA PROBLEMİ **17**

3.1 En Kısa Yolu Bulma Algoritmaları 17
3.1.1 Dijkstra algoritması 17
3.1.2 Bellman ve Ford algoritması 18
3.1.3 Floyd algoritması 19
3.1.4 Kruskal algoritması 20
3.1.5 Johnson's algoritması 20
3.1.6 En kısa yol bulma algoritmalarının kıyaslanması 21
3.2 En Kısa Yolu Bulmadaki Zorluklar 23
3.3 Gezgin Satıcı Problemi 23
4. TOPLULUK TESPİTİ 28
4.1 Topluluk Tespiti Algoritmaları 30
4.1.1 Kenar aralığı algoritması (Edge Betweenness) 31
4.1.2 Fastgreedy algoritması 32
4.1.3 Infomap algoritması 33
4.1.4 Etiket yayma (Label Propagation) 34
4.1.5 Leading Eigenvector algoritması 35
4.1.6 Multilevel algoritması 36
4.1.7 Walktrapp algoritması 37
4.2 Böl ve Fethet Metodu (Divide and Conquer) 39
5. ARAŞTIRMA BULGULARI 40
6. TARTIŞMA VE SONUÇ 43
KAYNAKLAR 47
ÖZGEÇMİŞ 52

SİMGELER DİZİNİ

G	Çizge
V(G)	G çizgesinin köşe kümesi
E(G)	G çizgesinin kenar kümesi
u	Düğüm sayısı
K_n	n köşeli tam çizge
$K_{r,s}$	iki parçalı tam çizge
c(G)	G çizgesinin bileşen sayısı
P	Çizge üzerinde bir yol

ŞEKİLLER DİZİNİ

Şekil 1.1 Topluluk tespiti uygulanmış bir çizge 3
Şekil 1.2 Königsberg köprülerinin gösterimi 5
Şekil 1.3 Königsberg Köprüleri probleminin çizge ile ifade edilmesi 5
Şekil 2.1 3 düğüm ve 3 kenardan oluşan bir çizge 9
Şekil 2.2 Çizge üzerinde yol gösterişim 9
Şekil 2.3 Yönlü çizge 10
Şekil 2.4 Yönsüz çizge 10
Şekil 2.5 Karışık çizge 11
Şekil 2.6 Ağırlıklı çizge 11
Şekil 2.7 Düzenli çizge 12
Şekil 2.8 Tam çizge 12
Şekil 2.9 Bilgisayar bilimi alanında çizge teori uygulamaları 13
Şekil 3.1 Dijkstra algoritmasının davranışı 18
Şekil 3.2 Bellman ve Ford algoritmasının uygulanabileceği bir çizge 19
Şekil 5.1 Topluluk tespitinden önceki sonraki en kısa yol grafiği 42
Şekil 5.2 Bütün bileşenlerin grafiği 42

ÇİZELGELER DİZİNİ

Çizelge 3.1 En kısa yol bulma algoritmalarının kıyaslanması 22
Çizelge 5.1 Çalışma Kapsamında Elde Edilen Sonuçlar 41

1. GİRİŞ

Başarılı bir şirketin kilit yönlerinden biri, tedarik zincirindeki değişikliklere hızlı ve etkili bir şekilde tepki verebilme yeteneğidir. Tedarik zinciri, çeşitli faaliyet alanları için bağımsız ekonomik konular sistemidir. Sistemin iç yapısı karmaşıktır ve bireysel iç unsurlar birçok farklı şekilde birbirine bağlıdır (Scott ve Carrington, 2011). Bu nedenle satış, sipariş ve lojistik yönetimine yönelik sistematik bir yaklaşıma olan talep artmıştır. Online alışverişin neredeyse tüm toplumlarda alışkanlık haline geldiği günümüzde kargo firmaları da hızlı bir adaptasyon ve dönüşüm sürecine girmiştir. Özellikle COVID-19 salgınının başlamasından sonra online alışveriş günümüzde sıradan bir alışkanlığı haline geldi. Pek çok farklı internet sitesi ve cep telefonu uygulaması sayesinde insanlar dışarı çıkmalarına gerek kalmadan istedikleri her şeyi kolayca kapılarına kadar getirebilmektedir. Bu durum müşteri memnuniyetini arttırmak adına bu hizmeti veren kurumlar için hızlı gönderi iletimini oldukça önemli hale getirmiştir. Bu tür sitelere olan talebin artması, hizmet kalitesinin ve operasyonel hizmetlerin üst düzeyde tutulmasını zorlaştırabilir. Güzergah planlaması, hem tedarik hem de dağıtımla ilgili tedarik zinciri yönetiminin önemli bir parçasıdır (Xie vd. 2011).

Bir internet sitesinden birden fazla ürün siparişi verdiğinde kargosunun aynı anda gelmediğini, hatta aynı kargo firması tarafından aynı gün içerisinde birkaç kez ziyaret edildiğini birçok kişi tecrübe etmiştir. Bu durum çoğu zaman müşteri memnuniyeti açısından sorun teşkil etmese de kargo firmaları için ekstra operasyonel maliyet ve insan gücü anlamına gelmektedir. Bu kargo dağıtım sürecini optimize etmek ve uygun rotayı planlamak kargo firmaları için daha hızlı ve verimli hizmet verebilmek için hayati bir ihtiyaç haline gelmiştir. Böyle bir uygulamanın ilk etapta firmalar tarafından kullanıldığı düşünülse de aslında durum tam tersidir. Operasyonel seviyedeki problemler genellikle stratejik ve taktiksel problemlerden daha az yapılandırılmıştır (Xie vd. 2011). Şirketler şehirlerarası veya uluslararası kargo gönderileri için bu tür planlamayı kullanır. Ancak nihai dağıtım noktası olarak adlandırılan iletim merkezlerine gelen kargolar tamamen o bölgede çalışan distribütörün tecrübe ve tercihlerine göre dağıtılmaktadır. Aslında bu göreve alınacak kişilerin o bölgede yaşıyor olması ve bölgeyi bilmesi işe alımlarda bir önceliktir. Bu tezde planlanan optimizasyon ve rota planlaması sayesinde işe yeni

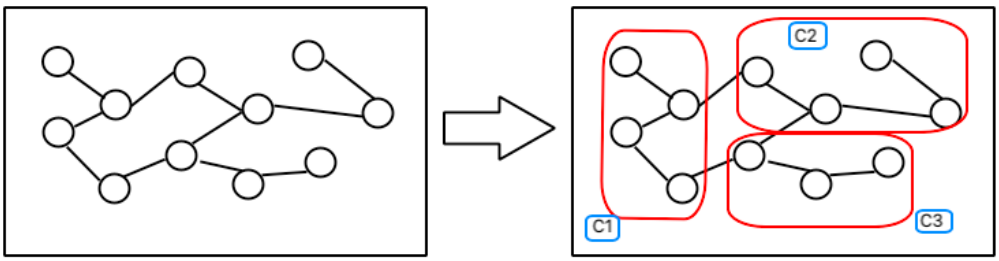
başlayan ve çalıştığı bölgeyi iyi bilmeyen bir kişi bile herhangi bir dağıtım şirketinde rahatlıkla çalışabilir. Yukarıda belirtilen gerçek dünya problemlerini çözmek için pek çok yöntem vardır ancak gerçek dünyadaki karmaşıklık çok yüksek olduğu için bu alternatif yöntemler her duruma çözüm bulamamaktadır. Bu tezde mevcut yöntemlere ek olarak yeni bir çözüm bulunması amaçlanmaktadır.

En kısa yol problemi, yapay sinir ağları, genetik algoritmalar, parçacık sürüsü optimizasyonu gibi birçok alanda akıllı optimizasyon teknikleri kullanılarak kapsamlı bir şekilde çalışılmış bir konu olmuştur (Danon vd. 2005). Aynı zamanda bu süreçte lojistik kargo sektörünün önemi de artış göstermiş ve özellikle 2019'da başlayan pandemiden beri sektör çok daha hızlı bir şekilde büyümeye devam etmektedir. Bu süreçte kargo istasyonlarının konumu, kargo araçlarının güzergâhlarının doğru planlanması gibi faktörler kritik öneme sahiptir. Günümüzde e-ticaret, çok fazla rekabet olmasından dolayı tüketici memnuniyetini arttırmak ve aynı zamanda olası müşterilere ulaşabilmek için çok hızlı değişim geçirmektedir. Sektördeki büyüme ve müşteri taleplerindeki hızlı teslimat talebi karşısında kaynakların daha verimli yönetilmesi gerekmektedir. Kargo şubelerinin yer seçimi ve kargo araçlarının rotalamak kaynakların doğru yönetilmesi ve daha iyi bir hizmet verebilmek için çok kritiktir (Potamia vd. 2009). Ağ analizi ve rota planlamasının taşımacılık sektöründeki uygulamalarının temel amacı, işletmelerin maliyetini minimize etme ve zamanı en verimli şekilde kullanma hedefine yönelik olarak en optimal rotaları belirlemesini sağlamaktır. Bu noktada son yıllarda giderek daha popüler hale gelen Topluluk Tespit Etme (Community Detection) algoritmaları bu alanda önemli bir yer edinmiştir.

Topluluklar her büyük çizgelerde doğal olarak bulunan yapılardır. İster bir sosyal ağda yer alan insanlar ister bir bölgede yaşayan insanlar olsun her büyük çizge kendisi içerisinde doğası gereği topluluklar barındırır. Bu toplulukları belirlemek ve analiz etmek birçok fayda sağlamaktadır. Bu tez kapsamında da en kısa yol bulma algoritmaları topluluk belirleme yöntemi ile birleştirilerek gerçek hayatta çok yüksek karmaşıklığa sahip olabilecek çizgelerin sadeleştirilmesi ve en hızlı şekilde bu topluluğun çözümlenmesi üzerine çalışılmıştır.

1.1 Tezin Amacı

Bu çalışma günümüzde var olan en kısa yol bulma algoritmalarını ve topluluk tespitini bir araya getirerek yeni bir en kısa yol bulma metodu önermektedir. Gerçek hayatta var olan çizgelerin karmaşıklığı yolma bulma algoritmalarının performanslarını bu çizgeler büyüdükçe düşürmektedir. Çizge ne kadar büyük olursa, hangi algoritma uygulanırsa uygulansın, düğüm sayısını artması karmaşıklığın da artışını beraberinde getirecektir. Özellikle gerçek dünyadan örnek vermek gerekirse İstanbul gibi büyük metropollerde çok dar bir alanda çok fazla şehirleşme olması bu noktaların dijital dünyadaki karşılığı sayabileceğimiz düğüm sayısı çok yüksek sayılara ulaşabilmektedir. Bu çizgeler Şekil 1'de de gösterildiği gibi topluluklara ayrılırsa tek bir seferde algoritmanın uygulanacağı toplam düğüm (node) sayısı önemli ölçüde azalacağından uygulanacak algoritmanın da performansı iyileştirilebilir.



Şekil 1.1 Topluluk tespiti uygulanmış bir çizge

Şekil 1.1'de görülebileceği gibi C1, C2 ve C3 olarak belirtilen topluluklar da artık tek bir düğüm gibi ele alınıp kendi aralarında da en kısa yol belirlenebilir. Hem toplulukların kendi içerisindeki en kısa yol hem de topluluklar arasındaki en kısa yol tek bir seferde büyük bir çizgeyi ele almaktan daha verimli bir yöntem olarak sunulmuştur.

1.2 Çizge Kuramı

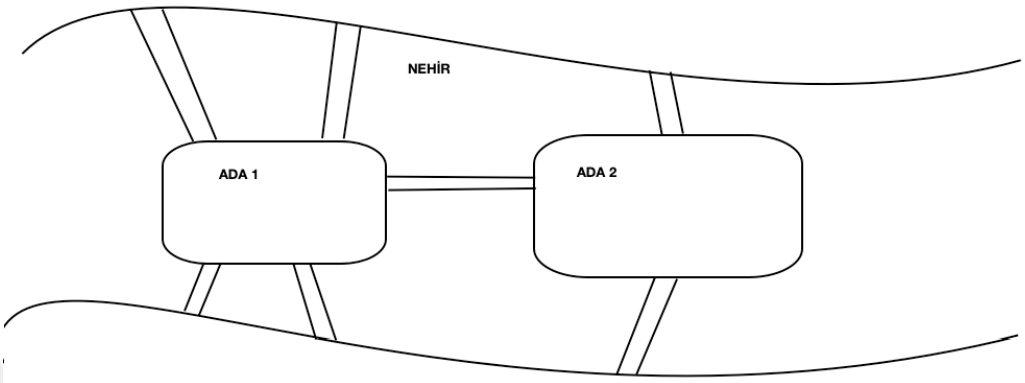
İnsanlık tarihi boyunca, insanlar her zaman ticaret amaçlı başka şehirlere ya da ülkelere ziyaretlerde bulunmuşlardır. Kendi dönemlerinin koşullarına göre her zaman en kısa yoldan hedeflerine varmak insanlık için önemli bir amaç olmuştur. Teknoloji ilerledikçe ve daha iyi yollar ve daha hızlı araçlar gibi imkanlar da geliştikçe insanların başka coğrafyalara seyahati kolaylaşmıştır. Bununla birlikte insanlığın gelişmesinin sonucu olarak yerleşim yerlerinin sayısı da giderek artmıştır.

Bu durum bir noktadan başlayıp birden fazla noktaya uğrayarak hedef noktasına ulaşmak isteyen insanlar için doğal olarak en kısa yoldan hedefe varma sorununa da ortaya çıkarmıştır. Çünkü her insan en az enerjiyi harcayarak en yüksek verimi almak istemektedir. Çizge kuramı da tam olarak bu noktada önem kazanmaktadır. Bir noktadan başlayıp başka bir yere ulaşırken en kısa yolu belirlemek için çeşitli modeller ve yöntemler geliştirilmiştir. Çizge kuramı günümüzde gerek bilgisayar mühendisliğinde gerek ekonomide ve pazarlamada önemli bir araç olarak kullanılır. En kısa yol bulma problemi de çizgelere uyarlanarak modellenip çözüme ulaştırılmaya çalışılmıştır. İster şehir içi ister şehirlerarası yollar olsun bu modellemeler insanların en kısa yolu bulmasına yardımcı olmaktadır.

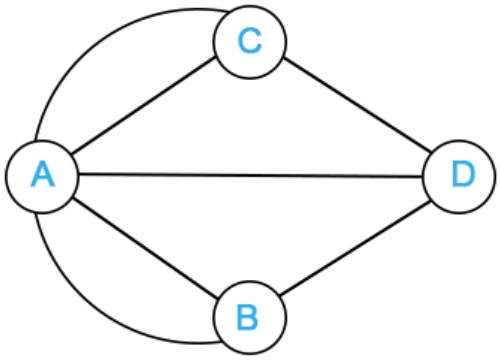
Köşeler (vertices) ve bu köşeleri çizgelerle (edges) birleştiren düğümlerden (nodes) oluşan yapı Çizge olarak isimlendirilir. Çizge teorisinin kavramı tarihte ilk kez 1735'te Koinsber Köprüsü probleminin ortaya atılmasıyla karşımıza çıkmaktadır (Sarma, 2012). Koinsber Köprüsü probleminin çözümüne ulaşmak isteyen Euler çizge kavramını hayatımıza kazandırmıştır. Daha sonrasında Euler. Koningsberg Köprüsü problemine odaklanmış ve bu problemi çözmek için Euler Çizgesi olarak adlandırılan yeni bir yapı geliştirmiştir.

Königsberg köprüleri problemi şehrin içinden geçen ve Pregel nehrinin üzerinde bulunan 7 köprüden her köprüden sadece bir kez geçmek şartıyla ve başlanılan noktaya geri dönmek zorunda olunan bir problemdir. Bu şartları sağlayacak bir yol bulmak problemin temel amacıdır. Şekil 1.2, bahsi geçen bu problemin konusu olan ve her iki adayı birbirine

bağlayan nehirlerin üzerindeki yedi adet köprünün görselleştirilmesidir. Şekil 1.3'te de bu problemi bir Çizge ile nasıl ifade edilebileceği gösterilmiştir.



Şekil 1.2 Königsberg köprülerinin gösterimi



Şekil 1.3 Königsberg Köprüleri probleminin çizge ile ifade edilmesi

Euler, bir gerçek hayat en kısa yol bulma problemini matematiksel olarak ifade ederek bu alanda önemli bir adım atmıştır. Çünkü bir sorunun matematiksel olarak ifade edilmesi artık o soruna olan bakış açısını değiştirmekte ve o problem üzerine çalışmasına başlamayı sağlamaktadır. Daha sonra A.F Mobius, 1840'ta iki çizge kavramını, yani tam çizge ve iki parçalı çizge kavramını verdi (Core ve Ijcem, 2014). Bu iki çizgeyi de rekreasyonel problemlerde kullanan Kuratowski rekreasyonel problemler alanında her iki çizgenin de düzlemsel olduğunu kanıtlamıştır (Voloshin, 2009). Gustav Kirchho, 1845'te döngüsüz ağaç yapısı fikrini ortaya attı (Kocay ve Kreher, 2017). Ayrıca yazarlar çizge

teorisi kavramının elektrik devrelerinde akım hesabında kullanımını açıklamışlardır. 1852'de Thomas Gutherie, günümüz bilgisayar uygulamalarında son derece kullanışlı olan dört renk problemini ortaya koydu (Mondal ve De, 2017). Daha sonra, 1856 yılında Hamiltom çizgesi, Thomas tarafından keşfedildi. P. Kirkman ve Hamiltonian ve bu tür çizgeler literatürde yaygın olarak kullanılmaktadır (Robertson vd. 1994). Bulmaca problemi, 1913'te H. Dudeney tarafından Çizge Teorisinin önceki kavramlarını kullanarak tanıtıldı(Bisen, 2017). 1878'de James Joseph Sylvester "Çizgeler" kelimesini tanıttı (Tyagi, 2014). İki cebir kavramı, yani ortak değişkenler ve niceliksel değişmezler kullandı ve aralarında bir analoji çizdi. Ramsey, 1941 yılında renklendirmeler üzerine yaptığı çalışmalarla dış çizge teorisi kavramını geliştirdi ve birçok bilim insanı da çizge teorisi üzerine çalışmalar yapmaya başlamıştı.

1.3 Literatürdeki Çalışmalar

Topluluk Tespiti ve karmaşık çizgelerde en kısa yol bulma problemi bilgisayar bilimlerindeki en popüler konulardan biridir. Bilinen en eski Topluluk Tespiti algoritması Forsyth ve arkadaşları tarafından 1946 yılında geliştirilmiştir. Bu alanda yapılan çalışmaların sayısı çok fazla olmakla beraber en bilinen ve en çok göze çarpan çalışmalar bu bölümde anlatılmıştır.

1.3.1 Sosyal ve biyolojik ağlarda topluluk yapısı

Bu çalışma bu alanda birçok çalışmaya imza atmış olan M. Girvan ve M.E.J Newman tarafından 2001 yılında gerçekleştirilmiş olup bu alanda bir çok çalışmaya imza atmışlardır. Birçok ağda bulunan ve ağ düğümleri sıkı sıkıya örülmüş gruplar halinde birleştiren, aralarında daha gevşek bağlantıların bulunduğu topluluk yapısının özelliği vurgulanmıştır. Topluluk sınırlarını belirlemek için merkezilik indekslerini kullanma fikri temel alınmış ve bu tür toplulukları tespit etmek için bir yöntem önerilmiştir. Bu yöntem, topluluk yapısı zaten bilinen bilgisayar tarafından oluşturulmuş ve gerçek dünya çizgeleri üzerinde test edilmiş ve yöntemin bu bilinen yapıyı yüksek hassasiyet ve güvenilirlikle saptadığını görülmüştür. Yöntem, topluluk yapısı iyi bilinmeyen iki ağa uygulanmış ve

her iki durumda da önemli ve bilgilendirici topluluk bölünmeleri saptadığını görülmüştür (Girvan ve Newman, 2002).

1.3.2 Ağlarda topluluk yapısını bulma ve değerlendirme

Girvan ve Newman tarafından yapılmış olan bu çalışmada ağlardaki topluluk yapısını keşfetmek için bir dizi algoritma önerilmiş ve incelenmiştir. Algoritmaların tümü iki kesin özelliği paylaşır: birincisi, ağdan topluluklara bölmek için kenarların yinelemeli olarak kaldırılmasını içerir, kaldırılan kenarlar, bir dizi olası "arada olma(betweenness)" ölçüsünden herhangi biri kullanılarak tanımlanır ve ikincisi, bu ölçüler, en önemlisi, her kaldırmadan sonra yeniden hesaplanır. Bunun yanı sıra, algoritmaların bulduğu topluluk yapısının gücünü ölçmek amacıyla bir ölçüt de sunulmuştur. Bu ölçüt, bir ağın bölünmesi gereken topluluk sayısını belirlemek için bir objektif bir ölçüm sunar. Algoritmaların hem bilgisayar tarafından üretilen hem de gerçek dünya ağ verilerindeki topluluk yapısını keşfetmede son derece etkili olduğunu gösterilmiştir (Newman ve Girvan, 2004).

1.3.3 Çizgelerde topluluk tespiti

Fortunato tarafından 2010 yılında yapılmış olan bu çalışmada modern ağ bilimi, karmaşık sistemler anlayışımıza önemli ilerlemeler getirilmiştir. Gerçek sistemleri temsil eden çizgelerin en alakalı özelliklerinden biri, topluluk yapısı veya kümelemeleri olarak tanımlanmıştır, yani, aynı kümenin köşelerini birleştiren birçok kenar ve farklı kümelerin köşelerini birleştiren nispeten az sayıda kenar ile kümelerdeki köşelerin organizasyonu olarak tanımlanabilir (Fortunato, 2010). Bu tür kümeler veya topluluklar bir grafiğin bağımsız bir örneği olarak düşünülebileceği gibi bu toplulukların belirlenmesi sistemlerin grafiksel olarak temsil edilebilmesi açısından büyür bir öneme sahiptir. Son birkaç yılda üzerinde çalışan büyük bir disiplinler arası bilim insanı topluluğunun büyük çabalarına rağmen henüz tatmin edici bir şekilde çözümlenemeyen bu probleme çözüm önerisi getirilmiştir. Problemin temel ögelerinin tanımından fizikçiler tarafından geliştirilen özel bir odaklanma ile çoğu yöntemin sunumuna kadar kümelemenin önemi ve yöntemi nasıl tespit edilip birbirleriyle karşılaştırılması gerektiği gibi önemli konular ele alınmıştır.

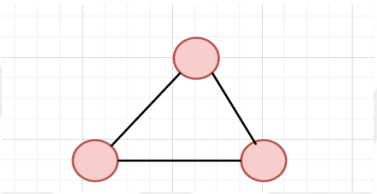
Ayrıca, gerçek dünya ağlarına uygulamaların açıklaması da yapılarak konu kapsamlı bir şekilde açıklanmıştır.

1.3.4 Çok büyük ağlarda topluluk yapısı bulma

O zaman kadar önerilen yöntemlerin çoğunun büyük ağlara uygulamasının maliyetli olmasından ayrışan Aaron Clause ve arkadaşları tarafından yapılan bu çalışmada fizik alanında oldukça ilgi gören bir konu olan ağlardaki topluluk yapısının keşfi ve analizi üzerine çalışılmıştır. Bu çalışmada, birçok rakip algoritmadan daha hızlı olan topluluk yapısını tespit etmek için hiyerarşik bir kümeleme algoritması sunulmuştur. Bu algoritmanın uygulanmasına bir örnek olarak da, büyük bir çevrimiçi perakendecinin web sitesinde satılık bir ürün ağını analiz etmek için kullanılmıştır; ağdaki ürünler, eğer aynı alıcı tarafından sıklıkla satın alınıyorsa, bağlantılıdır. Ağın 400 000'den fazla köşesi ve 2×10^6 kenarı vardır. Bu çalışma kapsamında geliştirilen algoritma ile, müşterilerin satın alma düzenleri ve mevcut olan büyük ölçekli benzerlikleri gösterilmeye çalışılmış ve bu ağdan bize faydalı bilgiler sağlayabilecek topluluklar çıkarılabileceği gösterilmiştir (Clauset vd. 2004).

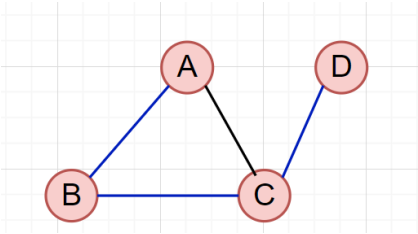
2. ÇİZGE ÇEŞİTLERİ VE ÇİZGE TEORİSİ UYGULAMALARI

Çizge çeşitleri ve uygulamalarını anlatmadan önce Çizge'nin ne olduğunu tanımlanmalıdır. Literatürdeki genel tanıma dayanarak bir çizge düğümler(vertices) ve kenarlar(edges) kümesi olarak matematiksel bir ifade ile $G = (V,E)$ şeklinde tanımlanabilir. Çizgeleri yönlü olup olmamasına göre Yönlü Çizge ve Yönsüz Çizge olarak ya da kenarlarının ağırlık değeri alıp almamasına göre ağırlıklı ve ağırlıksız çizgeler olarak sınıflandırabiliriz. Şekilde 2.1'de 3 düğüm ve 3 kenardan oluşan bir çizge örneği görülebilir.



Şekil 2.1 3 düğüm ve 3 kenardan oluşan bir çizge

Çizge üzerinde düğümler arasından geçen rotaya 'yol' denir. Örneğin Şekil 2.2'de gösterilen çizge A noktasından D noktasına ulaşmak için sırasıyla geçilen rota {A,B,C,D} yolu olarak ifade edilir.



Şekil 2.2 Çizge üzerinde yol gösterişim

2.1 Çizge Çeşitleri

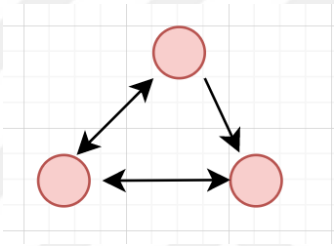
Literatüre geçmiş birçok çizge çeşidi bulunmaktadır. Bu kısımda bu çizgelerden en sık kullanılan ve en bilinenle tanıtılmıştır.

2.1.1 Yönlü çizge

Çizge teorisinde, düğümlerin hepsinin birer yöne sahip kenarlardan oluşan çizgeye veril isimdir. Biçimsel hareketle, bir çizge *G = (V,A)* sıralı çiftiyle ifade edilir:

- V düğümler ya da noktalar kümesidir,
- A Sıralı düğüm çiftlerinden oluşur ve oklar ya da kenarlar grubu olarak adlandırılır.

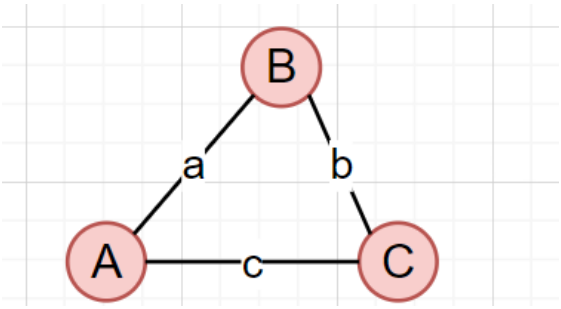
Yönlü çizge, kenarları sırasız düğüm çiftlerinden oluşan yönsüz çizgelerden ayrılır.



Şekil 2.3 Yönlü çizge

2.1.2 Yönsüz çizge

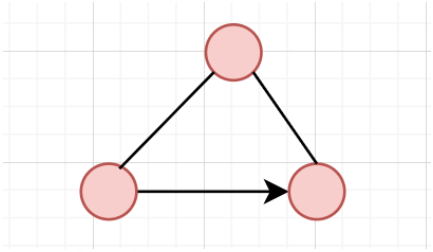
Bir çizgede bulunan kenarların yön bildirmemesi durumunda bu çizgeye yönsüz çizge denilir. Bu durumda iki düğüm arasında bulunan kenar, her iki yönlü de hareket edilebileceğini ifade eder. Örneğin Şekil 2.4'deki çizge yönsüzdür. Bu çizgede A ile B düğümleri arasında bir a kenarı bulunmaktadır. Bu kenar yönsüz olduğu için hem A'dan B düğümüne hem de B'den A düğümüne hareket etmenin mümkün olduğunu gösterir.



Şekil 2.4 Yönsüz çizge

2.1.3 Karışık çizge

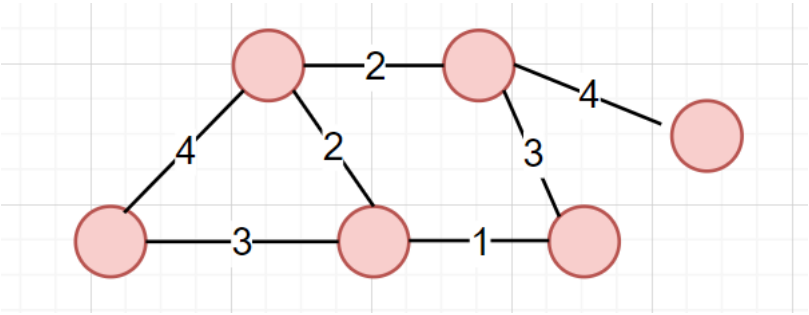
Çizge teorisinde, karışık bir çizge *G = (V, E, A)*, bir dizi *V* köşesi, bir dizi (yönsüz) kenar *E* ve bir dizi yönlendirilmiş kenar (veya yay) *A*'dan oluşan bir çizgedir (Beck vd, 2013).



Şekil 2.5 Karışık çizge

2.1.4 Ağırlıklı çizge

Ağırlıklı çizge her kenara bir sayının (ağırlık) atandığı bir çizgedir (Fletcher vd. 1991). Bu tür ağırlıklar, üzerinde çalışılan konuya göre farklılık göstermekle birlikte genellikle maliyetlere, uzunluklara veya kapasitelere karşılık gelir. Bu tür çizgeler pek çok bağlamda, örneğin Gezgin Satıcı Problemi gibi en kısa yol problemlerinde ortaya çıkar.

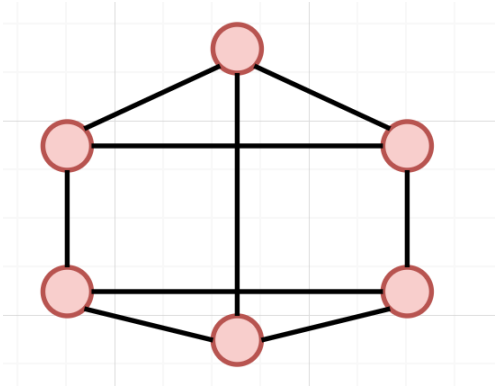


Şekil 2.6 Ağırlıklı çizge

2.1.5 Düzenli çizge

Çizge teorisinde, normal bir çizge, her tepe noktasının aynı sayıda komşuya sahip olduğu bir çizgedir; yani her tepe noktası aynı dereceye veya değerliliğe sahiptir. Düzenli yönlendirilmiş bir çizge, her bir iç köşenin derece ve dış derecesinin birbirine eşit olduğu

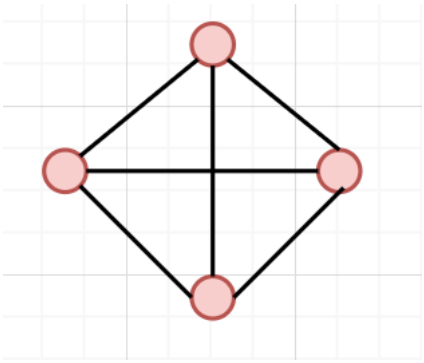
şeklindeki daha güçlü koşulu da sağlamalıdır(Chen, 1997). Köşeleri k derecesine sahip normal bir çizgeye k-düzenli çizgesi veya k derecesinin normal çizgesi denir



Şekil 2.7 Düzenli çizge

2.1.6 Tam çizge

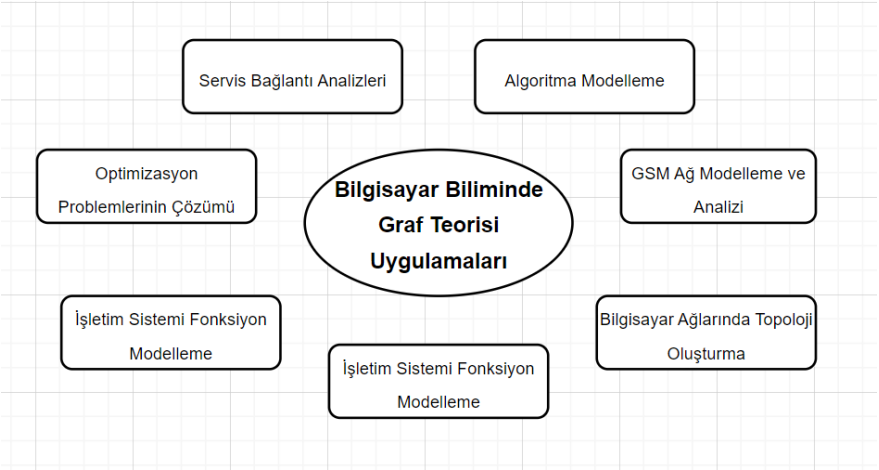
Çizge teorisinin matematiksel alanında, tam bir çizge, her farklı köşe çiftinin benzersiz bir kenarla birbirine bağlandığı basit, yönsüz bir çizgedir. Her farklı köşe çiftinin bir çift benzersiz kenarla (her yönde bir tane) birbirine bağlandığı yönlendirilmiş bir çizge Tam bir di-çizge olarak adlandırılır. (Bang-Jensen ve Gutin, 2018).



Şekil 2.8 Tam çizge

2.2 Teknolojide Çizge Teorisi Uygulamaları

Bir dizi çalışma, bilimin ve sosyal ağların benzersiz yönlerinde çizge teorisi uygulamalarını bildirmiştir. Çizge Teorisini kullanan bu iki alanla ilgili benzersiz yönler şunlardır: kriptografi (Sciences, 2015), kuantum kriptografi (Ganzha ve Maciaszek, 2019) kodlama teorisi (Polak ve Roma, 2013), asimptotik olarak en kısa k-yarıçap dizilerinin oluşturulması (Jaromczyk vd. 2012), bilgilendirici analiz için mahremiyeti koruyan çizge yayını (Yuan vd., 2015), çizgeler aracılığıyla insan beyni anatomik ağının incelenmesi (Itturia vd. 2008), peyzaj bağlantısı ve koruma planlaması (Minor ve Urban, 2008), görüntü işleme (Chrysochoos e Louche, 2000), bilgi alma (Salembier ve Garrido, 2000), sosyal ağ analizi (Campbell vd. 2013), sinyal işleme çizgeleri (Shuman vd. 2013), sosyal ağlarda bilgi keşfi (Cordero vd. 2015), robotik (Lee, 1989), dünyadaki metro sistemlerinin ağ analizi (Derrible ve Kennedy,2009) yarı kesin programlama (De Klerk, 2010), görüntü bölümleme (Saerens vd. 2004), kümeleme (Qiantt ve Suent, 2000), veri bilimi (Durant vd. 2013) ve örüntü tanıma (Siquera vd. 2014). Bunlar, diğerlerinin yanı sıra, iyi bilinen çizge teorisinin benzersiz uygulamalarıdır. Çizge Teorisi uygulamalarının pratik uygulanabilirliği bu çalışmalar tarafından kapsamlı bir şekilde rapor edilmemiştir. Ayrıca, literatürde hem bilgisayar biliminde hem de sosyal ağlarda Çizge Teorisinin tüm günlük hayata uygulanabilir kapsayan tek bir çalışma bulunmamaktadır.



Şekil 2.9 Bilgisayar bilimi alanında çizge teori uygulamaları

2.2.1 Veri madenciliğinde çizge teorisi

Veri madenciliği, çok büyük verilerden ihtiyaç duyulan verilerin farklı yöntemler yardımıyla kavranması sürecidir. Veri biliminde ele alınan veriler genellikle çizge yapısına dönüştürülebilmektedir. Bu çizgeler, bu amaç için tasarlanmış algoritmalar aracılığıyla daha iyi analiz ve işlem yapabilmek için, çıkarılabilir. Sosyal ağlarda her birey ağdaki bir düğümü temsil ederken aralarındaki ilişkiler kenarlarla ifade edilir. Çizge teorisi, bu alandaki sorunları çözmek için güçlü bir teori ve bir dizi algoritma ile desteklenen bir model sunmada kullanılmaktadır.

2.2.2 Ağ tasarımı

Web sitesi tasarımı, web sayfalarının köşelerle ve aralarındaki köprülerin çizgedeki kenarlarla adlandırıldığı bir çizge olarak yapılandırılabilir ve bu kavram web grafiği olarak isimlendirilir. Çizgelerin diğer uygulama alanları web topluluğundadır. Köşelerin nesne sınıflarını oluşturduğu ve her köşenin bir tür nesneyi simgelediği ve her köşenin, başka türdeki nesneleri simgeleyen her köşeye bağlı olduğu yer. Arama ve topluluk keşfi gibi web sitesi geliştirmede çizge teorisinin birçok faydası vardır, Yönlendirilmiş Çizge web sitesi kullanışlılık değerlendirmesinde ve bağlantı yapısında kullanılır.

2.2.3 Dil işleme

Birçok uygulamada bir kullanıcı tarafından girilen cümlenin doğru sözdizimsel yapıya sahip olup olmadığını anlamak çizge teorisinden faydalanılır. Bu ayrıştırma ağacı, sözcük varlıkları üzerinde yönlendirilmiş döngüsel olmayan çizgeden yapılmıştır. Girdinin doğru yapısını belirlemek ve dilin tüm işlemesine yardımcı olmak için buraya çizge yerleştirilmiştir.

2.2.4 Kod çözümleme

Genellikle, modern kodlama teorisinde, kanaldan alınan kod kelimelerinin kodunu çözmek için ikili çizge uygulanır. Bu alanda kullanılmak için geliştirilen Tanner çizgesi, iki parçalı çizgenin bir uygulamasıdır, bu nedenle, köşeler, ilk ikiye bölmenin kod kelimesinin basamağını temsil ettiği ve diğer yan ikiye bölmenin, hatalar dışında bir kod sözcüğünde toplamı sıfır olması beklenen rakamların kombinasyonunu temsil ettiği iki kısma ayrılır.

2.2.5 Elektronik çip tasarımı

Elektronik çip tasarımı da çizgelerin kullanıldığı alanlardan biridir ve bu yapıda her bir bileşen için çizgenin tepe noktası olarak ele alınmaktadır. Bu bileşenler arasındaki bağlantı baskılı devre kartına dönüştürmek için kullanılan bir makine, kenarların bileşen çiftleri arasında bir ilişki olduğunu anlamına gelen bir çizge biçiminde girdi alır. Kart üzerinde bu bağlantıları oluşturan baş, daha sonra talep edilen sonuç devresini elde etmek için çip boyunca en uygun şekilde hareket etmek için gerek yolları belirler.

2.2.6 Kimya

Kovalent bağlı bileşiklerin yapısal formülleri çizgelerdir; anayasal çizgeler olarak bilinirler. Çizge kuramı tanımlama, numaralandırma, sistematizasyon, kodlama, terminoloji, korelasyon ve bilgisayar programlama için temel sağlar. Yapısal bilgiler ile ilişki olan kimyasal bilgiler tutarlı ve benzersiz bir şekilde üs olabilir ve itfa edilebilir. Kimyasal yapılar terminoloji kurallarına göre kelimelere çevrilir. Çizgeler polimer için kritik bir öneme sahiptir. Kimya alanında çizge teorisinin önemi, özellikle kimyasal yapı teorisinin rasyonelleştirilmesinde izomerizm olayının varlığına dayanmaktadır. Bu teori, tüm yapısal izomerleri hesaplamak için tamamen çizge teorisi yöntemlerini kullanır. Bu sayede, moleküllerin farklı yapılarının ve izomerlerinin anlaşılması ve analizi mümkün hale gelebilmektedir.

2.2.7 Biyoloji

Çizge teorisi gerçekten geniş bir alandır ve biyolojik ağların analizinde de yaygın olarak kullanılır. Biyoloji analizinde, sistemin bileşenleri arasındaki etkileşimler ağ olarak modellenir ve genellikle binlerce düğümün çeşitli etkileşimlerle bağlandığı çizgeler olarak gösterilir. Çizgeler protein-protein etkileşim ağları, düzenleyici ağlar gibi bir çok analizde yaygın olarak kullanılır.

Çizgeler bu bileşenlerin analizinde kullanıldığında çizge teorisindeki yapılar ile benzer ağ yapıları oluşturulabilir. Biyolojik analizde iki çizge arasında yapısal benzerlikleri tespit etmek için çizge izomorfizmi yöntemi kullanılabilir. Eğer iki çizge izomorfikse bu iki biyolojik ağın aynı yapısal özelliklere sahip olduğu sonucuna varılabilir.

2.2.8 Coğrafya

Bir ülkenin haritasını ele aldığımızda, her ülke bir köşe olsun ve bu ülkeler bir sınırı paylaşıyorsa, iki köşeyi bir kenarla birleştirsin. Yüz yılı aşkın bir süredir çözülemeyen ünlü bir problem, dört renk problemiydi. Kabaca bu, herhangi bir haritanın komşu iki ülke aynı renge sahip olmayacak şekilde en fazla 4 renkle boyanabileceğini belirtir. Çizge teorisinin gelişmesini önemli ölçüde ilerleten bu problem 1976 yılında bilgisayar yardımı ile kanıtlanmıştır.

3. EN KISA YOL BULMA PROBLEMİ

En kısa yol bulma problemi literatürde en çok çalışılan konulardan biridir. Bu problem üzerine birçok algoritma geliştirilmiş olup her bir algoritmanın kendine ait avantajları ve dezavantajları bulunmaktadır. Bizim problemimizde ise geleneksel yaklaşımın aksine algoritmaları büyük bir çizgeye uygulamak yerine bu çizgeyi topluluklara bölüp algoritmaların topluluğa göre avantajından faydalanmak üzerine çalışılmıştır. Öncelikle literatürde en çok üzerinden durulan algoritmalar bu bölümde tanıtılacaktır.

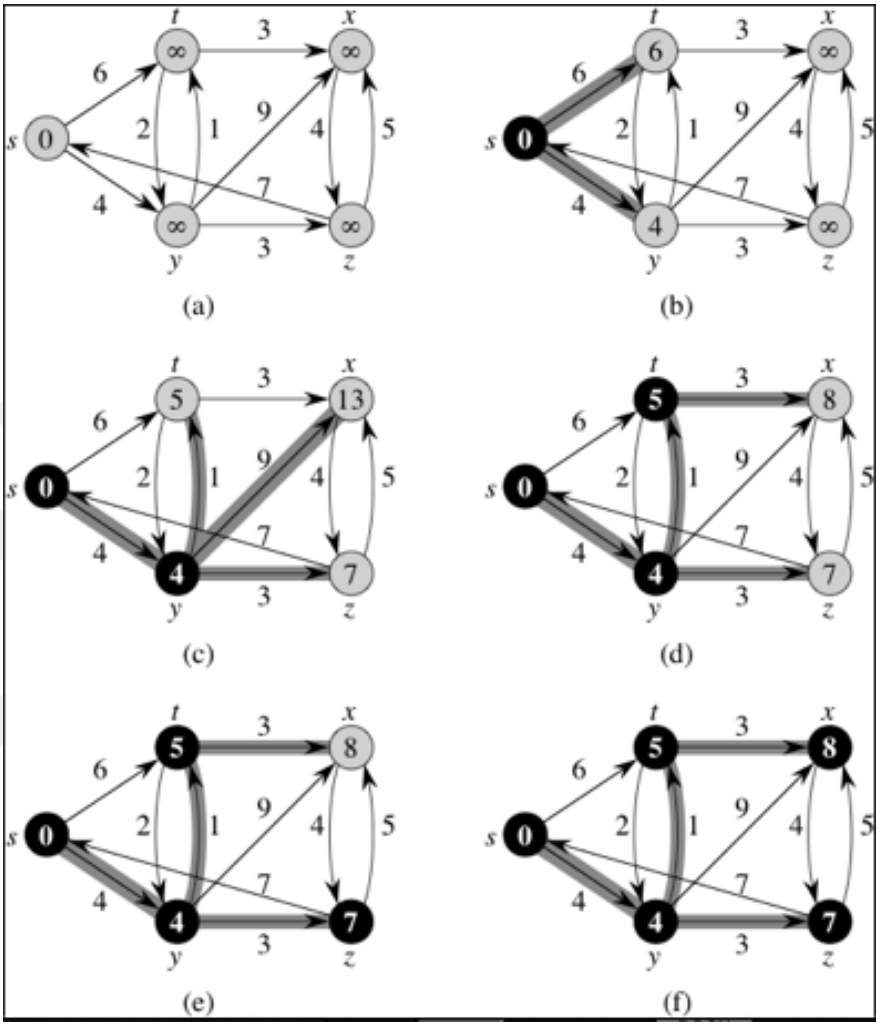
3.1 En Kısa Yolu Bulma Algoritmaları

Bilgisayar biliminin en popüler alanlarından biri olan en kısa yol bulma problemi için birçok algoritma geliştirilmekle birlikte hala üzerinde çalışılan bir konudur. Bu kısımda bu alanda bilinen algoritmalar tanıtılmış ve performansları kıyaslanmıştır.

3.1.1 Dijkstra algoritması

En kısa yol bulma algoritmalarından belki en çok bilinenlerinden biri olan Dijkstra algoritması başlangıçta belirlenmiş bir noktadan en kısa yolu bulan bir algoritmadır. Ağırlıklı ve yönlü çizgeler için geliştirilmiş olan bu algoritma, belirlenen bir başlangıç düğümünden, diğer tüm düğümlere olan en kısa yolu belirler. Çizge üzerindeki her bir kenarın ağırlığı sıfır veya sıfırdan büyük olması bu algoritmanın çalışması için gereklidir. Dijkstra'nın algoritmasının bir çizge içerisindeki en kısa yolu bulmaya çalışırken kullandığı metoda Açgözlü (Greedy) metot denir. Her adımda, bir düğümden sonraki düğüme ilerlerken Açgözlü yaklaşım kullanılır. Dijkstra algoritmasının davranış biçimi Şekil 3.1'de 5 düğümlü örnek bir çizge üzerinde gösterilmiştir. Bu örnekte, bir çizgenin başlangıç durumundan başlayarak ve sırasıyla birinci, ikinci, üçüncü, dördüncü, beşinci ve son adımdaki davranışı görülebilir. Başlangıç düğümü A olarak kabul edilen örnek çizgedeki ilk alınan aksiyon, A düğümünden direkt gidilebilecek düğümlere ait maliyet ve rota bilgilerinin güncellemektir. Her adımda bir sonraki düğüme gitme işlemi bir

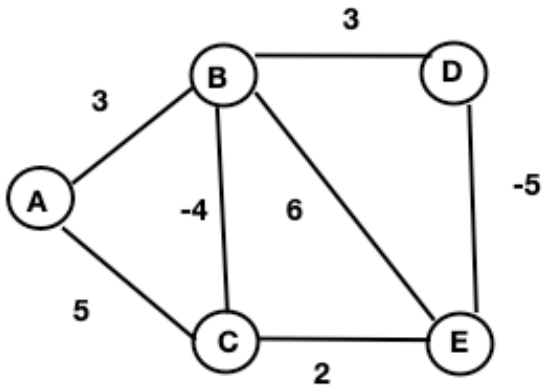
düğümden başka bir düğüme giderken en kısa yolu tercih eden Açgözlü yaklaşımına göre gerçekleştirilir.



Şekil 3.1 Dijkstra algoritmasının davranışı

3.1.2 Bellman ve Ford algoritması

Bellman ve Ford Algoritması, Dijkstra algoritmasının aksine eksi maliyetli çizgelerde de çalışabilmektedir. Dijkstra algoritması kenar maliyetleri sıfır veya artı olan çizgeler için doğru sonuç verirken Dijkstra'dan farklı olarak Bellman ve Ford Algoritmasının kenar maliyetleri eksi olan çizgeler için de çalışmaktadır. Şekil 3.2'de Bellman ve Ford Algoritmasının çalışabileceği örnek bir çizge gösterilmiştir.



Şekil 3.2 Bellman ve Ford algoritmasının uygulanabileceği bir çizge

3.1.3 Floyd-Warshall algoritması

Floyd-Warshall algoritması, en kısa yolları bulmak için kullanılan bir çizge algoritmasıdır. Genellikle ağırlıklı yönlendirilmiş çizge (weighted directed graph) yapısıyla kullanılır. Algoritma, her iki düğüm arasındaki en kısa yolun tüm düğümler üzerinden geçebilecek şekilde güncellenmesini sağlar. Bu algoritma, başlangıçta, tüm düğümler arasındaki mesafeleri ve yolları en sonsuz (∞) olarak kabul eder, ancak her düğüm kendisine olan mesafeyi 0 olarak kabul eder. Daha sonra algoritma, tüm düğüm çiftleri için kendi aralarındaki mesafeyi güncellemeye başlar. Her adımda, bir düğüm çifti için, eğer mevcut yol daha kısa bir yol oluşturuyorsa, yani ihtiyaç duyulan mesafe daha kısa ise, bu yeni mesafeyi günceller. Tüm düğüm çiftleri üzerinde bu işlemi tekrar edilir ve böylece en kısa yolları bulunmuş olur.

3.1.4 Kruskal algoritması

Kruskal algoritması, bir ağırlıklı bağlı olmayan graf içerisindeki en küçük ağırlıklı kesir ağacını (minimum spanning tree) bulmak için kullanılan bir algoritmadır. Başlangıçta, her düğüm kendine ait bir ayrık küme olarak düşünülür ve daha sonra tüm kenarlar ağırlık sırasına göre sıralanır. Eğer seçilen kenar, iki ayrı kümeyi birleştiriyorsa, o zaman bu kenar seçilir ve iki küme birleştirilir. Bu işlemi tüm kenarlar üzerinde tekrar edilir, ta ki tüm düğümler birleşik bir kümede toplanana kadar veya ağaç boyutu n-1 (n düğüm sayısı) olana kadar devam edilir. Sonuç olarak, Kruskal algoritması, verilen graf içerisindeki en küçük ağırlıklı kesir ağacını (minimum spanning tree) bulmamıza yardımcı olur. Bu algoritma, genellikle ağlar arasındaki iletişimi en düşük maliyetle sağlamak için kullanılır.

3.1.5 Johnson’s algoritması

Kenar ağırlıkları belirleniş bir çizgedeki tüm köşe ikilileri arsındaki en kısa yolu bulmaya çalışan Johnson'ın algoritmasında bazı kenar ağırlıkları negatif değerler alabilir fakat negatif ağırlık döngüleri olmayabilir. İlk olarak Bellman-Ford algoritmasını kullanarak tüm negatif ağırlıkları kaldırılır ve çizgenin dönüşümü. Daha sonra dönüştürülmüş çizgede Dijkstra algoritması uygulanır.

Johnson'ın algoritması, çizgeye sıfır ağırlıklı kenarlarla diğer düğümlerin her birine bağlanan yeni bir düğüm ekler. Ardından, yeniden diğerine olan her köşeyi bulmak için Bellmann-Ford algoritmasını kullanır. Bu adımda negatif bir döngü tespit edilirse algoritma sonlandırılır. Ardından başlangıçtaki çizgedeki kenarlar kullanılarak yeniden ağırlıklandırılır.

En son adımda ise en başta çizgeye eklenen düğümün kaldırılma işlemi gerçekleştirilir. Yeniden ağırlıklandırılan çizgedeki Dijkstra algoritması yardımıyla tüm tepe noktalarının en kısa yolları bulunur. Bu işlemin sonundan döndürülen mesafe daha sonra orijinal çizgedeki mesafeye eklenerek her bir mesafe için hesaplanır.

3.1.6 En kısa yol bulma algoritmalarının kıyaslanması

En kısa yol algoritmaları, çizge teorisinde önemli bir yer tutar ve iletişim, ulaşım ve elektronik problemlerinde geniş uygulama alanlarına sahiptir. Bu algoritmalar, çeşitli en kısa yol problemlerini çözmek için kullanılır ve farklı durumlara göre farklı algoritmalar tercih edilir.

Öne çıkan en kısa yol algoritmaları şunlardır:

1. Dijkstra Algoritması: Tek çift, tek kaynak ve tek hedef en kısa yol probleminde kullanılır. Kenar ağırlıklarının pozitif olması gereklidir.

2. Johnson Algoritması: Tüm çiftlerin en kısa yol problemi için kullanılır. Negatif ağırlıklara izin verir ancak negatif ağırlık döngüsü olmamalıdır.

3. Floyd-Warshall Algoritması: Tüm köşe çiftleri arasındaki en kısa yolu hesaplamak için kullanılır. Kenar ağırlıkları pozitif veya negatif olabilir.

4. Bellman-Ford Algoritması: Tek kaynak probleminde kullanılır. Negatif ağırlıklara izin verir ve negatif döngüleri tespit edebilir.

Bu algoritmaların performansı ve uygunluğu, çizgenin yoğunluğuna ve ağırlıklarına bağlı olarak farklılık gösterebilir. Örneğin, Johnson Algoritması seyrek çizgelerde Floyd-Warshall'dan daha hızlı olabilirken, Floyd-Warshall çizgenin yoğun olduğu durumlarda daha etkili olabilir. Dijkstra Algoritması bellek kullanımı ve negatif ağırlıklarla başa çıkma açısından avantajlıdır, ancak negatif ağırlıklı çizgelerde kullanılamaz.

Bu algoritmalardan hangisinin tercih edileceği, problem ve çizge yapısına bağlı olarak değişebilir. Her bir algoritmanın avantajları ve sınırlamaları göz önünde bulundurularak en uygun çözüm seçilmelidir.

Çizelge 3.1 En kısa yol bulma algoritmalarının kıyaslanması

Algoritma	**Avantajları**	**Dezavantajları**	**Alan Maliyeti**	**Zaman Maliyeti**
Dijkstra	- Düşük karmaşıklık - Her zaman en optimal yolu bulması	- Eksi ağırlıklılarda çalışmaması - Eksi döngülerde kesinlikle bir sonuç vermemesi - Çok zaman alması	$O(v^2)$	$O(\|E\| + \|V\|Log\|V\|)$
Bellman-Ford	- Eksi ağırlıkları çözümleyebilmesi - Eksi döngüleri belirleyebilmesi	- Karmaşıklığının Dijkstra algoritmasına göre daha yüksek olması	$O(v^2)$	$O(VE)$
Floyd-Warshall	-Uygulaması daha kolaydır -Büyük çizgelerde daha iyi sonuçlar verebilir -Negatif döngüleri belireyebilir	- Karmaşıklık çok yüksektir - Diğer algoritmalara göre daha yavaş çalışır	$O(n^3)$	$O(n^3)$
Kruskal	- Uygulaması kolaydır - Tüm çizgelerde kullanılabilir - Seyrek çizgelerde daha etkili çalışır	- Yoğun çizgelerde yavaş çalışır - Her zaman en kısa yolu vermeyebilir	$O(\|E\| + \|V\|)$	$O(ELogE)$
Johnson	-Dijkstra algoritmasını negatif ağırlıklar için de çalışmasını sağlar - Seyrek çizgelerde hızlı çalışır	- Çok geniş kullanım alanı yoktur		$O(V^2LogV + VE)$

v: köşe sayısı, u: düğüm sayısı, E: kenar sayısı

3.2 En Kısa Yolu Bulmadaki Zorluklar

Çizge teorisi, modern matematik ve bilgisayar biliminin önemli bir dalıdır ve birçok gerçek dünya probleminin çözümünde kullanılan temel araçlardan biridir. Bu alandaki en temel problemlerden biri, en kısa yol bulma problemdir. İki nokta arasındaki en kısa yolu bulma ihtiyacı, iletişim ağlarından ulaşım sistemlerine kadar birçok alanda ortaya çıkar. Ancak, en kısa yol bulma probleminin altında yatan zorluklar oldukça karmaşıktır ve farklı algoritma tasarımları ve yaklaşımları gerektirir.

En kısa yol bulma probleminin karmaşıklığını anlamak için öncelikle çizge teorisinin temel kavramlarına göz atalım. Bir çizge, düğümler ve bu düğümleri birleştiren kenarlardan oluşan bir yapıdır. Bu düğümler genellikle bir noktanın veya bir varlığın temsil edildiği yerlerdir, kenarlar ise bu noktalar arasındaki ilişkileri ifade eder. En kısa yol bulma problemi, iki düğüm arasında, belirli bir ölçüt (örneğin, uzunluk veya maliyet) doğrultusunda en kısa yolu bulmayı amaçlar.

En kısa yol bulma algoritmalarının başa çıkmak zorunda olduğu temel zorluklardan biri, negatif ağırlıklı kenarların varlığıdır. Negatif ağırlıklar, bazı yolları diğerlerinden daha kısa hale getirebilir, ancak aynı zamanda döngülere yol açabilir. Bu durum, bazı en kısa yol algoritmalarının işlevselliğini sınırlayabilir.

En kısa yol bulma probleminin karmaşıklığı, çizgenin büyüklüğüne göre değişebilir. Büyük çizgelerde en kısa yol bulma işlemi daha uzun sürebilir ve daha fazla bellek gerektirebilir. Bu nedenle, büyük ölçekli ağlarda verimli ve ölçeklenebilir algoritmaların kullanılması önemlidir.

Gerçek dünya problemlerinde genellikle farklı kriterler dikkate alınır. Örneğin, ulaşım ağı tasarımında en kısa yolun yanı sıra trafik yoğunluğu veya maliyet de göz önünde bulundurulmalıdır. Farklı kriterlerin dengelemesi ve optimize edilmesi zor olabilir.

Çizgeler zaman içinde değişebilir. Yolların kapanması, yeni yolların açılması gibi dinamik değişiklikler, en kısa yol hesaplamalarını etkileyebilir. Dinamik değişikliklerin hızlı ve etkili bir şekilde yönetilmesi önemlidir.

En kısa yol bulma probleminin farklı alanlara adapte edilmesi farklı zorluklar doğurabilir. Örneğin, iletişim ağları veya elektronik devre tasarımında en kısa yol bulma, farklı kısıtlama ve gereksinimlere sahip olabilir.

Büyük çizgelerde en kısa yol hesaplamaları oldukça zaman alabilir. Bu nedenle, hesaplamaların paralelleştirilmesi ve dağıtık sistemlerde etkili bir şekilde çalışabilmesi önemlidir.

En kısa yol bulma probleminin bu zorlukları, bilgisayar bilimcileri ve matematikçileri farklı algoritma tasarımları ve yaklaşımları geliştirmeye yönlendirmiştir. Bellman-Ford, Dijkstra, Floyd-Warshall gibi algoritmalar bu zorlukları aşmaya yönelik çeşitli yaklaşımlar sunar. Her bir algoritmanın avantajları ve dezavantajları vardır ve doğru algoritma seçimi problem bağlamına ve gereksinimlere bağlıdır. En kısa yol bulma probleminin bu zorlukları, çizge teorisi alanındaki araştırmaları ve gelişmeleri sürekli olarak motive etmektedir ve daha etkili algoritmaların geliştirilmesi için bir ilham kaynağı olmaya devam etmektedir.

3.3 Gezgin Satıcı Problemi

Gezgin Satıcı Problemi (Traveling Salesman Problem), kısaca GSP ya da TSP olarak adlandırılır. Bilinen düğümlerin (örneğin şehirler veya noktalar) arasındaki mesafelerin belirli olduğu ve her düğümden yalnızca bir kez geçileceği koşuluyla, belirli bir düğümden başlayarak aynı düğüme geri gelmek şartıyla maliyeti en az olan veya en kısa yolu bulmayı hedefleyen bir problemdir. Bu problem, ulaşım rotalarının belirlenmesi, lojistik dağıtım uygulamalarının tasarlanması, uçak rotalarının hesaplanması, elektronik devre tasarımı gibi gerçek hayatta karşılaşılan ve çözüm bekleyen sorunları içermesi nedeniyle sıkça çalışılmaktadır. Ayrıca, sürekli olarak üzerinde sürekli çalışılan bir konu

olduğu için GSP, süreç ve kaynak kullanımında en iyi sonucu elde etmek için ortak ve ideal bir test platformu olmuştur.

Gezgin Satıcı Problemi, Simetrik Gezgin Satıcı Problemi (Symmetric Traveling Salesman Problem - SGSP) ve Asimetrik Gezgin Satıcı Problemi (Asymmetric Traveling Salesman Problem - AGSP) olarak genellikle iki türde ele alınır. Simetrik Gezgin Satıcı Problemi, bir satıcının belirli bir sayıdaki şehri ziyaret etmesi gerektiği ve her şehri sadece bir kez ziyaret etmesi gerektiği bir optimizasyon problemidir. Amacı, başlangıç noktasından başlayarak tüm şehirleri gezerek en kısa yolun (en kısa mesafenin) bulunmasıdır. Bu problemin "simetrik" olarak adlandırılmasının sebebi, herhangi iki şehir arasındaki uzaklığın gidip gelme yönünde eşit olduğu durumu ifade etmesidir.

Asimetrik Gezgin Satıcı Problemi, bir satıcının belirli bir sayıdaki şehri ziyaret etmesi gerektiği ve her şehri sadece bir kez ziyaret etmesi gerektiği bir optimizasyon problemidir. Amacı, başlangıç noktasından başlayarak tüm şehirleri gezerek en kısa yolun (en kısa mesafenin) bulunmasıdır. Ancak, AGSP'nin özelliği, herhangi iki şehir arasındaki uzaklığın gidip gelme yönünde eşit olmayabileceği durumları içermesidir. Yani A şehrinden B şehrine gitmek, B şehrinden A şehrine gitmekten farklı bir mesafe veya maliyet gerektirebilir.

Bu problemin genel çözümü oldukça karmaşıktır, çünkü tüm şehirlerin farklı kombinasyonları ve yolları düşünülmelidir. TSP'nin çözümü, genellikle matematiksel optimizasyon yöntemleri veya sezgisel yaklaşımlar kullanılarak yapılır. Matematiksel yöntemler, dinamik programlama ve tam sayılı programlama gibi teknikleri içerebilir. Sezgisel yaklaşımlar arasında genetik algoritmalar, simüle edilen tavlama, karınca kolonisi optimizasyonu gibi teknikler yer alabilir.

Hangi çözüm yönteminin tercih edileceği, problem boyutu, karmaşıklık düzeyi ve hesaplama kaynaklarının durumuna göre değişebilir. TSP'nin optimal bir çözümünü bulmak pratikte büyük boyutlu problemler için genellikle çok uzun süre ve kaynak gerektirir. Bu nedenle gerçek dünyada genellikle yaklaşık çözümler veya heuristikler kullanılır.

GSP (Gezgin Satıcı Problemi), bir döngü oluşturur ve düğümler tarafından oluşturulan bir kümede belirli bir noktadan başlayıp tekrar aynı noktaya dönmeyi hedefler. Bu döngü oluşturulurken temel hedef, noktalar arasında en az maliyetli yolu seçmektir. Ancak burada en çok dikkat edilmesi gereken kısım en uygun maliyetli yolu seçerken sadece yerel bir arama yapmamaya özen göstermektir. GSP'ye genel bir bakış açısıyla yaklaşılmalı ve sadece komşu noktalar arasındaki maliyetleri göz önünde bulundurmaktansa tüm noktaların birbirleri arasındaki maliyetler toplu bir şekilde hesaplanmalıdır.

Yani, GSP'nin çözümünde sadece yakınlık ve komşuluk ilişkilerine odaklanmak yerine, tüm noktalar arasındaki mesafelerin kapsamlı bir şekilde değerlendirilmesi gerekmektedir. Böylece, sadece yakındaki noktalar arasındaki düşük maliyetli yolları seçmek yerine, tüm noktalar arasındaki kombinasyonları dikkate alarak en az maliyetli döngüyü bulmak mümkün olacaktır.

Bu nedenle, GSP'yi çözerken geniş bir perspektiften bakmak, daha iyi ve daha optimal sonuçlar elde etme şansını artırabilir. Global olarak tüm noktaların maliyetlerini toplu olarak değerlendirmek, problemin karmaşıklığına rağmen daha verimli ve daha iyi çözümler bulmaya yardımcı olabilir.

Bu tez kapsamında çizge içerisinde en kısa yolu bulmak için ve oluşturulan topluluklardaki en kısa yolu bulmak için Christofides algoritmas kullanılmıştır. Bu algoritma Gezgin Satıcı Problemi'nin (GSP) yaklaşık çözümü için kullanılan etkili bir algoritmadır. Bu algoritma, *Nlogn* karmaşıklığı ile GSP'nin %50'ye kadar optimal çözümlerini üretmek için kullanılır. Çok yüksek düğüm sayısına sahip problemler için hızlı ve kabul edilebilir çözümler sunar.

Christofides algoritmasının temel adımlarını aşağıdaki gibidir:

Adım 1: Verilen düğümler arasındaki tüm mesafeleri hesaplanır.

Adım 2: Minimum Geri Dönüş Ağacı (Minimum Spanning Tree, MST) oluşturulur.

a. MST, verilen düğümleri bağlayan ağaçlardan en az maliyetli ağacı temsil eder. MST algoritmaları, genellikle Kruskal veya Prim algoritması gibi kullanılır.

Adım 3: MST'deki düğümlerin derecesini kontrol edin ve çift sayıya getirilir. MST'deki her düğümün derecesi, o düğüme giren ve çıkan kenarların sayısıdır. Tüm düğümlerin derecesini çift sayıya getirmek için eşleştirme yapılır.

Adım 4: Çift sayıda düğüme sahip oluşan ağaçta, minimum maliyetli eşleştirmeyi (Minimum Weight Perfect Matching) bulunur.

a. Bu adımda, çift sayıda düğüme sahip olan ağaçta, her düğümü diğer bir düğümle eşleştiren ve toplam maliyeti minimum olan eşleştirmeyi bulmak için eşleştirme algoritmaları kullanılır. Edmonds' Blossom Algoritması gibi eşleştirme algoritmaları sıklıkla tercih edilir.

Adım 5: İlk adımda oluşturulan MST ile ikinci adımda oluşturulan eşleştirme ağacını birleştirin ve böylece çevreleyen döngü oluşturun.

Adım 6: Çevreleyen döngüdeki tekrar eden düğümleri birleştirerek son çözümü elde edin.

Christofides algoritması, bu adımların tamamlanmasıyla GSP için yaklaşık bir çözüm elde eder. Elde edilen çözüm, GSP'nin %50'sine kadar optimal olacaktır. Christofides algoritması, $nlogn$ karmaşıklığı ile oldukça hızlı bir şekilde çalışır ve GSP'nin büyük ölçekli problemleri için pratik bir seçenek sunar. Ancak çok büyük düğüm sayısına sahip problemler için daha hassas çözümler elde etmek için diğer daha karmaşık algoritmalar da kullanılabilir.

4. TOPLULUK TESPİTİ

Son yıllarda bilgisayar bilimi, bilişim ve sosyal medya şirketleri gibi birçok alanda ağlar üzerine yapılan araştırmalar büyük ilgi görmüştür (J. Scott, P.J. Carrington, 2011). Bu ağlar, ağı daha etkin ve verimli bir şekilde yönetmek için topluluklara bölünebilir. Bir topluluk, çizgeler açısından, aynı çizgedeki diğer topluluklardaki düğümlere sıkı ve gevşek bir şekilde bağlı olan düğümlerin bir alt kümesi olarak tanımlanabilir. Danon ve arkadaşlarının da çalışmalarında belirttiği gibi köşe ve kenarlardan oluşan çizgeler, günlük yaşantımızda birçok sorunu ve uygulamayı görselleştirmek için uygun bir yapıya sahiptirler. (Danon vd. 2005). Bunun günlük hayatımızda en sık kullandığımız kullanım alanı internet ve dünya çapında ağ (world wide web) ağlar olarak gösterilebilir. Bu nedenle, toplulukların tespit edilmesi ve ağlarda topluluk yapılarının değerlendirilmesi, sağladığı faydalar nedeniyle birçok sektör için büyük önem taşımaktadır.

Topluluk tespiti, yapısal özelliklerine bağlı olarak bir ağdaki etkileşimli köşe gruplarını (yani düğümleri) tanımlama sürecini ifade eder (Yang vd, 2013; Kelley vd, 2012). Birçok topluluk tespiti algoritması farklı disiplinlerden metot ve teknikler kullanılarak geliştirilmiştir ve biyoloji, coğrafya, fizik ve bilgisayar bilimleri gibi çeşitli alanlarda kullanılmaktadır (Lancichinetti ve Fortunato, 2009). Bununla birlikte, farklı süreçlerden üretilen çok çeşitli karmaşık ağlar olduğu için tek bir topluluk algılama algoritması tüm ağ türlerinde performans gösteremez (Plantié ve Crampes, 2013; Yang vd., 2016). Ağlar statik veya dinamik olabileceğinden, topluluk algılama algoritmaları büyük ölçüde ağların topolojisine dayanır. Dinamik bir ağ ile kıyaslandığında statik bir ağda topluluk keşfi kolaydır.

Topluluk tespitinde karşılaşılan en büyük zorluklardan biri, toplulukların nesnel olarak tanımlanmasının güçlüğüdür. Topluluklar genellikle belirli bir bağlam içinde anlam ifade eder ve bu bağlamın doğru şekilde tanımlanması ve anlaşılması önemlidir. Örneğin, bir sosyal ağdaki topluluklar belirli ilgi alanlarına, etkileşim tiplerine veya demografik özelliklere göre gruplandırılabilir. Bu nedenle, topluluk tanımının doğru ve anlamlı bir şekilde yapılması, topluluk tespiti yöntemlerinin temel bir bileşenidir. Topluluk yapısının karmaşıklığı da bir diğer zorluktur. Bir ağ içinde topluluklar hiyerarşik bir yapıya sahip

olabilir, örtüşen topluluklar bulunabilir ve ağ dinamik olarak değişebilir. Bu faktörler, topluluk tespitini daha karmaşık hale getirir ve yöntemlerin bu yapılara uygun şekilde adapte edilmesini gerektirir. Veri boyutu ve ölçeklendirme de önemli bir problem olabilir. Büyük veri setleri üzerinde topluluk tespiti yapmak, hesaplama gücü ve bellek yönetimi açısından zorlayıcı olabilir. Bu nedenle, ölçeklenebilir algoritmaların geliştirilmesi ve veri boyutuna uygun çözümlerin bulunması önemlidir. Gizlilik ve güvenlik de topluluk tespitin karşılaştığı zorluklardan biridir. Özellikle sosyal ağlarda, bireyler arasındaki ilişkilerin gizliliği korunmalıdır. Topluluk tespiti yöntemlerinin bu gizlilik gereksinimlerini karşılayacak şekilde tasarlanması ve uygulanması gereklidir.

Gelecekte, daha gelişmiş algoritmaların ve yapay zeka tekniklerinin topluluk tespiti alanında daha etkili sonuçlar elde etmek için kullanılması beklenmektedir. Derin öğrenme ve büyük veri analitiği gibi teknikler, topluluk yapısını daha hassas bir şekilde analiz etme ve anlamlandırmada yardımcı olabilir. Ayrıca, farklı veri türlerinin ve özelliklerinin daha iyi entegre edildiği ve topluluk yapısını daha iyi yansıtan yöntemlerin geliştirilmesi hedeflenmektedir.

Sonuç olarak, bilgisayar biliminde topluluk tespiti, ağ analizi, veri madenciliği ve sosyal bilimlerin kesişim noktasında önemli bir araştırma alanıdır. Topluluk tespiti yöntemleri, ağlardaki gizli yapıları ortaya çıkarmak, içgörüler elde etmek ve karmaşık ağlarda düzeni anlamak için güçlü bir araçtır. Ancak, bu alanda hala çözülmesi gereken zorluklar ve açıklar bulunmaktadır ve gelecekteki çalışmalar bu zorlukları aşmayı hedeflemelidir. Bu şekilde, topluluk tespiti, daha geniş bir bağlamda bilgi çıkarımı ve veri anlayışı için önemli bir kaynak olmayı sürdürecektir.

4.1 Topluluk Tespiti Algoritmaları

Bu bölümde, önce kullanılan kıyaslama ağlarını elde etme yöntemleri açıklanmış ve ardından da kullanılan topluluk algılama algoritmalarını sıralanmıştır.

Topluluk tespiti kavramının ortaya çıkışından beri karmaşık ağların niteliklerini belirlemek ve ayrıştırabilmek için için birçok topluluk algılama algoritması geliştirilmiştir ve geliştirilmeye de devam etmektedir. Ancak bir algoritmanın ne kadar verimli olduğu konusu hala üzerinde çalışılan ve tartışılan bir konudur. Algoritmaların gerçek dünyada test etmek ve ön kabullerinin doğruluğunu anlamak belirli kısıtlamalara sahip olabilmektedir. Çünkü gerçek ağlar genellikle çok büyük ve karmaşık bir yapıya sahiptir. Ayrıca toplulukların hangi özelliklerine göre sınıflandırılacağı, üzerinde çalışılan konuya veya odak noktasına göre değişiklik gösterebilmektedir.

Topluluk belirleme algoritmalarını karşılaştırırken, gerçek veya topluluk yapısı zaten bilinen, genellikle temel gerçek olarak adlandırılan yapay ağlar kullanılmaktadır. Gerçek ağlar için ünlü Zachary'nin Karate Kulübü veya Amerikan Kolej Futbol Takımları ağı yaygın olarak kullanılmıştır. (Zachary 1977, Girvan M. ve Newman M., 2002) Yapay ağlar arasında ise, daha yaygın olarak kullanılanlar GN ve LFR kriterleridir (Girvan M. ve Newman M., 2002, Lancichinetti, A. ve Fortunato, S. 2009). Bununla birlikte, temel bir gerçeğin ilişkilendirilebileceği gerçek ağlar elde etmek yalnızca zor değil, aynı zamanda ekonomik açıdan ve zaman açısından da maliyetlidir. Veri toplamanın karmaşıklığı ve maliyetleri nedeniyle, gerçek dünya kıyaslamaları genellikle küçük boyutlu ağlardan oluşur. Dahası, algoritmalar gerçek bir ağın tüm özelliklerini tamamen temsil edebilecek bir veri grubu olmadığından sınırları çizilmiş ve belirli özelliklere sahip durumlarda test edilebilir. Gerçek dünya örneklerinin nesnel bir şekilde tanımlanamaması ya da çok az görülen bir şekilde benzersiz bir topluluk oluşturması topluluk tespiti için uygun bir durum oluşturmaz. Ancak, yapay yollarla elde edilen ağlar gerçek dünyada karşımıza çıkan bu sınırlamaların ortadan kaldırılmasına yardımcı olabilir.

4.1.1 Kenar aralığı algoritması (Edge Betweenness)

Girvan ve Newman tarafından geliştirilen bu algoritma topluluk tespiti için yaygın olarak kullanılan bir yöntemdir. Bu algoritma, bir ağdaki kenarların (edges) önemini ölçerek topluluk yapılarını tespit etmeye çalışır. Kenar aralığı, bir kenarın ağ içindeki en kısa yol hesaplamalarında ne kadar sık kullanıldığını temel alır.

Algoritma, aşağıdaki adımlarla çalışır:

1. Başlangıçta, her kenara ait kenar aralığı değeri sıfıra ayarlanır.
2. Tüm düğüm çiftleri üzerinde en kısa yol hesaplamaları yapılır. Bu hesaplamalar sırasında, her kenarın ne sıklıkla kullanıldığı kaydedilir.
3. Bir düğüm çiftinin en kısa yolu birden fazla yol içeriyorsa, bu yolların sayısı eşit olarak dağıtılır ve her yolun üzerinden geçen kenarlar için kenar aralığı değerleri arttırılır.
4. Kenarların kenar aralığı değerleri güncellendikten sonra, en yüksek kenar aralığı değerine sahip olan kenar ağdan çıkarılır. Bu, ağın yapısal olarak bölünmesine neden olabilir ve topluluklar ortaya çıkabilir.
5. Kalan kenarlar üzerinde tekrar en kısa yol hesaplamaları yapılır ve kenar aralığı değerleri güncellenir.
6. Bu işlem, topluluklar netleşene kadar tekrarlanır.

Kenar aralığı algoritması, ağın kenarlarının önemini ve bağlantılarını belirleyerek toplulukları tespit eder. Kenarlar üzerinde yapılan bu hesaplamalar, kenarların hangi topluluklar arasında önemli bağlantılar kurduğunu gösterir. Yüksek kenar aralığı değerine sahip kenarlar, ağdaki toplulukları birbirine bağlayan kritik geçiş noktalarını temsil eder. Bu kenarların çıkarılması, toplulukları açığa çıkarabilir.

Kenar aralığı algoritması, ağ yapısını ve toplulukları anlamak için kullanışlı bir araçtır. Ancak büyük ağlarda hesaplama maliyeti yüksek olabilir ve bazen küçük toplulukları atlayabilir. Bu nedenle, farklı topluluk tespiti yöntemleriyle birleştirilerek kullanılabilir veya daha büyük ağlarda daha verimli algoritmalar tercih edilebilir.

4.1.2 Fastgreedy algoritması

Clauset ve arkadaşları tarafından geliştirilenr Fast Greedy algoritması topluluk tespiti için kullanılan etkili bir yöntemdir. Bu algoritma, ağdaki toplulukları tespit etmek için

hızlı ve basit bir yaklaşım sunar. Fast Greedy algoritması, ağın yapısını kullanarak toplulukların belirlenmesini amaçlar.

Fast Greedy algoritmasının çalışma prensibi şu adımlarda özetlenebilir:

1. Başlangıçta, her düğümü kendi topluluğunda tek bir üye olarak düşünün.
2. Her adımda, iki farklı topluluğu birleştirerek birleşik topluluk oluşturun. Bu birleştirme sırasında, toplulukların birleşiminden elde edilen kazancı ölçen bir değer kullanılır. Genellikle bu kazanç, modülerite olarak adlandırılan bir ölçü ile ifade edilir. Modülerite, topluluk içindeki bağlantı yoğunluğunun beklenen bağlantı yoğunluğuna göre ne kadar yüksek olduğunu belirten bir değerdir.
3. En yüksek kazancı sağlayan topluluk birleşimi gerçekleştirilir ve bu iki topluluk tek bir topluluk olarak ele alınır.
4. Adımlar 2-3 tekrar edilir ve her adımda en yüksek kazanca sahip topluluk birleşimleri gerçekleştirilir.
5. Algoritma, artık topluluk birleşimi sağlayamayana kadar devam eder.

Fast Greedy algoritması, topluluk yapısını hızlı bir şekilde tespit etmek için kullanışlı bir yöntemdir. Ancak algoritma bazen büyük toplulukları daha küçük topluluklara bölme eğiliminde olabilir ve bunun sonucunda bazı topluluklar göz ardı edilebilir. Bu nedenle, sonuçların yorumlanması ve gerekirse başka yöntemlerle doğrulanması önemlidir.

Ayrıca, Fast Greedy algoritması modülerite gibi belirli bir hedef fonksiyonunu optimize ederek çalışırken, gerçek dünya ağlarında topluluk yapısı genellikle daha karmaşıktır ve birden fazla kriteri içerebilir.

4.1.3 Infomap algoritması

Rosvall ve diğerleri tarafından geliştirilen InfoMap algoritması topluluk tespiti için kullanılan bir yöntemdir. Bu algoritma, ağı optimize etmek ve topluluk yapısını belirlemek için bilgi teorisine dayanan bir yaklaşımı benimser. InfoMap, ağın yapısındaki bilgi akışını kullanarak toplulukları tespit eder.

InfoMap algoritmasının çalışma prensibi şu adımlarda özetlenebilir:

1. Başlangıçta, her düğümü kendi topluluğunda tek bir üye olarak düşünün.
2. Her adımda, ağ yapısındaki bilgi akışını temsil eden bir optimizasyon hedefi belirlenir. Bu hedef, ağın topolojisindeki düğümlerin ve kenarların ne kadar iyi bir şekilde örgütlendiğini ölçer. Bu hedef, ağın kodlanması gereken minimum bilgi miktarını temsil eder.
3. Algoritma, ağı bu optimizasyon hedefine göre yeniden düzenleyerek topluluklar oluşturmaya çalışır. Daha iyi bir kodlama elde etmek için topluluklar arasındaki bilgi akışını artırır.
4. Toplulukların sıkışıklığı arttıkça, InfoMap algoritması ağı daha iyi bir şekilde kodlar ve böylece topluluklar daha belirgin hale gelir.
5. Adımlar 2-4 tekrar edilir ve ağdaki bilgi akışı en aza indirildiğinde, topluluklar istenen düzeyde tanımlanmış olur.

InfoMap algoritması, ağ yapısındaki bilgi akışını temel alan bir topluluk tespit yöntemidir. Bu yaklaşım, ağın içsel yapısını anlamak ve toplulukları daha iyi tanımlamak için kullanışlıdır. Bununla birlikte, algoritmanın parametrelerinin ve optimizasyon hedefinin doğru şekilde ayarlanması önemlidir. Ayrıca, InfoMap'in bazı durumlarda ağ yapısını aşırı özelleştirebileceği ve bu nedenle gerçek dünya ağlarda dengeli sonuçlar elde edilmesini zorlaştırabileceği unutulmamalıdır.

InfoMap algoritması, topluluk tespiti için farklı bir perspektif sunar ve ağ yapısındaki bilgi akışını kullanarak toplulukları belirlemeye çalışır.

4.1.4 Etiket yayma (Label Propagation)

Raghavan ve diğerleri tarafından geliştirilen Etiket Yayma algoritması topluluk tespiti için kullanılan bir yöntemdir. Bu algoritma, etiketlerin yayılması yoluyla toplulukları tespit etmeye çalışır. Başlangıçta rastgele veya belirli bir kritere göre düğümlere etiketler atanır ve bu etiketler komşu düğümlere yayılır.

Etiket Yayma algoritmasının çalışma prensibi şu adımlarda özetlenebilir:

1. Başlangıçta, her düğüme bir etiket atanır. Bu etiketler, düğümleri farklı topluluklarda temsil eder.
2. Her adımda, her düğüm kendi etiketini güncellemek üzere komşularından aldığı etiketleri değerlendirir. Bu adımda komşuların sahip olduğu etiketlerin ağırlıklı ortalaması hesaplanır ve düğümün yeni etiketi olarak atanır.
3. Adım 2 tekrar edilirken, etiketler komşu düğümler arasında yayılarak topluluklar oluşur. Bu yayılma işlemi ağın tüm düğümlerine yayılıncaya kadar devam eder.
4. Adımlar 2 ve 3 tekrar edilir ve etiketler istikrarlı hale gelene kadar güncellenir.

Etiket Yayma algoritması, etiketlerin komşu düğümler arasında hızla yayılmasına dayandığı için hızlı bir şekilde çalışabilir. Ancak bu hızlı yayılma nedeniyle bazı durumlarda aşırı özelleşebilir ve gerçek dünya ağlarda dengesiz sonuçlar elde edebilir.

Etiket Yayma algoritması, genellikle büyük ağlarda kullanılan ve kolay uygulanan bir topluluk tespiti yöntemidir. Ancak, etiketlerin başlangıçta nasıl atanacağı ve nasıl yayılacağı gibi parametrelerin doğru şekilde ayarlanması önemlidir. Ayrıca, algoritmanın bazı durumlarda aşırı özelleşebileceği ve çıkarımların dikkatli bir şekilde değerlendirilmesi gerektiği unutulmamalıdır.

Etiket Yayma hızlı ve basit bir topluluk tespiti yöntemi olarak kullanılabilir. Ancak daha karmaşık ağ yapıları ve topluluk yapıları için daha spesifik yöntemlerle birleştirilerek daha kesin sonuçlar elde edilebilir.

4.1.5 Leading Eigenvector algoritması

Newman tarafından geliştirilen Leading Eigenvector algoritması, topluluk tespiti için kullanılan bir yöntemdir ve genellikle büyük ağlarda etkili sonuçlar elde etmek için tercih edilir. Bu algoritma, ağın yapısını temsil eden matrislerin özdeğer ve özvektör analizini kullanarak toplulukları tespit etmeye çalışır.

Algoritmanın çalışma prensibi şu adımlarda özetlenebilir:

1. Ağ yapısını temsil eden bir matris oluşturulur. Genellikle bu matris, düğümlerin ve kenarların ilişkisini ifade eden bir komşuluk matrisi veya benzeri bir matrisdir.
2. Matrisin en büyük özdeğerine (leading eigenvalue) ve buna karşılık gelen özvektöre (leading eigenvector) odaklanılır. Bu özdeğer ve özvektör, ağın yapısındaki önemli topluluk yapılarını yansıtabilir.
3. Özvektörün bileşenleri, düğümlerin topluluk üyeliklerini temsil eder. Örneğin, özvektörün i. bileşeni, i. düğümün hangi topluluğa ait olduğunu gösterebilir.
4. Elde edilen özvektör bazen normalleştirilir ve ardından kümeleme algoritmaları kullanılarak topluluklar belirlenir. Bu adım, her düğümün en büyük özvektör bileşenlerine göre hangi topluluğa ait olduğunu belirlemeye yardımcı olur.

Leading Eigenvector algoritması, özellikle büyük ağlarda hızlı bir şekilde çalışabilir ve belirli topluluk yapılarını tespit etme yeteneğine sahiptir. Bu algoritma, ağın yapısındaki düzenlilikleri ve topluluk yapılarını ortaya çıkarabilir. Ancak, tüm topluluk yapılarını tespit etmek yerine, en büyük özdeğeri ve buna karşılık gelen özvektörü kullanarak genellikle ana toplulukları belirler.

Leading Eigenvector algoritması, diğer topluluk tespiti yöntemleriyle birlikte kullanıldığında daha kesin sonuçlar elde etmek için kullanılabilir. Ayrıca, topluluk tespitinde farklı yaklaşımların birleştirilerek daha geniş bir perspektiften ağ yapısının analiz edilmesi mümkündür.

4.1.6 Multilevel algoritması

Blondel ve arkadaşları tarafından geliştirilen Multilevel algoritması, büyük ağlarda topluluk tespiti yapmak için kullanılan etkili bir yöntemdir. Bu algoritma, ağ yapısını hiyerarşik olarak böler ve daha küçük alt-ağlarda toplulukları tespit ederek daha sonra bu alt-ağları birleştirir. Bu sayede, büyük ağları daha küçük ve daha yönetilebilir parçalara bölerken toplulukları daha hassas bir şekilde tespit etme imkanı sağlar.

Algoritmanın çalışma prensibi şu adımlarda özetlenebilir:

1. Başlangıçta, ağ yapısı rastgele veya başka bir yöntemle biraz küçültülür ve daha basit bir temsile dönüştürülür. Bu, büyük ağ yapısının karmaşıklığını azaltmayı amaçlar.
2. Daha basit temsil üzerinde bir topluluk tespiti algoritması uygulanır. Bu, küçük alt-ağlarda toplulukları tespit etmeye yardımcı olur.
3. Elde edilen topluluklar, orijinal ağ yapısına yansıtılır ve her bir topluluğu bir düğüm olarak ele alarak daha büyük bir ağ yapısı oluşturulur.
4. Oluşturulan yeni ağ yapısı üzerinde tekrar topluluk tespiti algoritması uygulanır. Bu adım, daha geniş topluluk yapılarını tespit etmeye yardımcı olur.
5. Adım 3 ve 4, birkaç seviye boyunca tekrar edilir. Her seviye, daha geniş topluluk yapılarına odaklanarak daha geniş bir perspektif sağlar.
6. En son seviyede elde edilen topluluklar, tüm ağ yapısına yansıtılır ve son topluluk yapıları elde edilir.

Multilevel algoritması, büyük ağların analizinde zaman ve kaynak açısından daha verimli sonuçlar elde etmeye yardımcı olabilir. Alt-ağlarda toplulukları tespit ederek başlamak,

daha karmaşık ve geniş topluluk yapılarını belirlemeyi kolaylaştırabilir. Bu yöntem, özellikle büyük ölçekli sosyal ağlar gibi karmaşık yapıları olan ağlarda kullanıldığında etkili sonuçlar sağlayabilir.

Multilevel algoritması, diğer topluluk tespiti yöntemleriyle birlikte kullanılabileceği gibi, tek başına da kullanılabilir. Her seviyede topluluk tespiti işlemi daha küçük ve daha anlamlı parçalarda gerçekleştirildiği için, sonuçlar genellikle daha doğru ve anlamlı olabilir. Bu nedenle, büyük ağ yapılarını analiz etmek ve toplulukları tespit etmek için Multilevel algoritması etkili bir seçenek olabilir.

4.1.7 Walktrapp algoritması

Pon ve Latapy tarafından geliştirilen Walktrap algoritması, topluluk tespiti için kullanılan bir yöntemdir. Bu algoritma, ağ yapısında benzer düğümleri bir araya getirerek toplulukları tespit etmeye çalışır. Walktrap algoritması, düğümler arasındaki yürüyüşlerin uzunluğunu kullanarak toplulukları tanımlar.

Walktrap algoritmasının çalışma prensibi şu adımlarda özetlenebilir:

1. Başlangıçta, her düğümün bir topluluk üyesi olarak başladığı varsayılır. Yani her düğüm, kendi topluluğuna aittir.
2. Ardışık adımlarda, belirli bir uzunluktaki yürüyüşlerin (walks) olasılıkları hesaplanır. Bu yürüyüşler, düğümler arasında rastgele gezintileri temsil eder.
3. Yürüyüşlerin olasılıkları, iki düğümün benzerliğini ölçmek için kullanılır. İki düğüm arasındaki yürüyüş olasılıkları ne kadar benzerse, bu düğümler aynı topluluğa ait olma olasılığı o kadar yüksektir.
4. Her adımda, düğümler arasındaki benzerlikler güncellenir ve topluluklar yeniden düzenlenir. Daha benzer düğümler aynı topluluğa dahil edilirken, daha az benzer düğümler farklı topluluklara ayrılabilir.

5. Adımlar tekrar edildiğinde, topluluklar giderek daha kesin bir şekilde tanımlanır. Her adımda topluluklar daha belirgin hale gelir ve düğümler arasındaki ilişkiler daha netleşir.

Walktrap algoritması, topluluk tespiti için etkili bir yöntemdir çünkü düğümler arasındaki yürüyüşlerin benzerliklerini kullanarak toplulukları tespit etmeye çalışır. Bu sayede, düğümler arasındaki yapısal ilişkileri ve bağlantıları dikkate alarak anlamlı topluluklar belirlenebilir.

Algoritma, büyük ağ yapısında da kullanılabilecek kadar ölçeklenebilir ve genellikle iyi sonuçlar verir. Walktrap, ağ yapısının lokal özelliklerine odaklandığı için, küçük ölçekli toplulukları da tespit etme yeteneğine sahiptir.

Walktrap algoritması, diğer topluluk tespiti yöntemleriyle birlikte kullanılabileceği gibi, tek başına da etkili sonuçlar sağlayabilir. Toplulukların tespiti, ağ analizinde önemli bir rol oynadığından, Walktrap gibi algoritmaların kullanılması, ağ yapısının daha iyi anlaşılmasına ve analiz edilmesine yardımcı olabilir.

4.2 Böl ve Fethet Metodu (Divide and Conquer)

Gerek bilgisayar bilimlerinde gerek diğer alanlarda çok büyük bir kümeye sahip olan problemleri daha basite indirgemek ve yönetimini kolaylaştırmak için kullanılan en popüler metotlardan biri olan Böl ve Fethet Metodu birçok problemin çözümünde önemli bir rol oynamaktadır. Böl ve Fethet algoritması basitçe üç bölümden oluşmaktadır:

- Problemi daha basit parçalara ayır
- Alt problemleri çöz
- Alt problemlerin çözümünü birleştirerek esas problemin çözümüne ulaş

Bu tez konusunun kapsamında olan ve çok fazla düğümlerden oluşan çizgelerde de hem problemim çözüm süresini kısaltmak için hem de en kısa yolu daha etkili bir şekilde

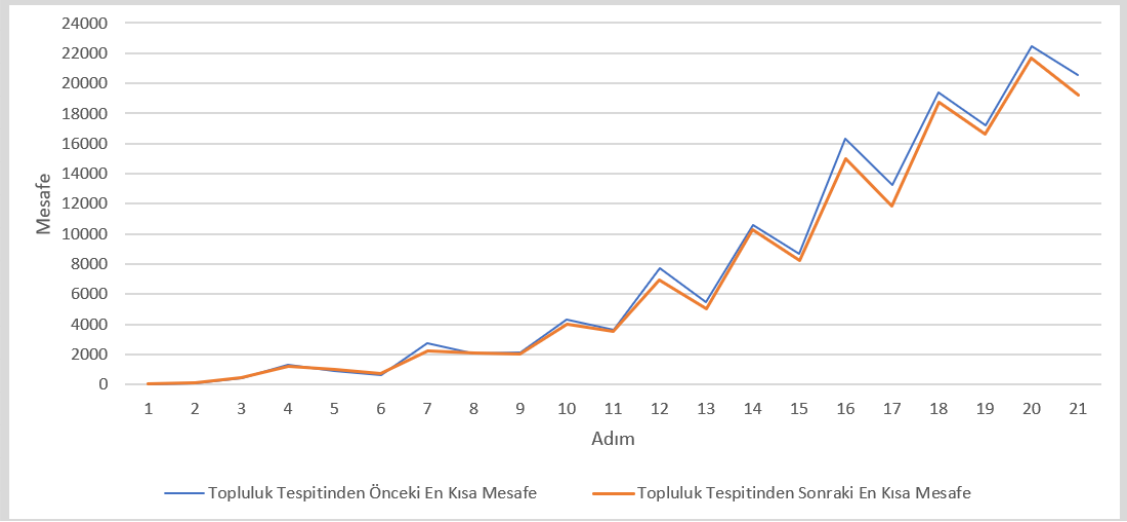
bulmak için bu metot uygulanmıştır. Verilen düğümleri önce topluluk tespiti yapıp alt topluluklar ayrılmış ve devamında da bu alt topluluklara en kısa yol bulma algoritmaları uygulanmıştır. Eğer böl ve fethet metodu yapılmadan önce bu algoritmalar ayrı ayrı uygulanırsa hem algoritmanın toplamdaki çözüm süresi hem de karmaşıklık seviyesi artacaktır. Bunun önüne geçmek için toplulukları alt topluluklara bölmek önemli bir adımdır.

5. ARAŞTIRMA BULGULARI

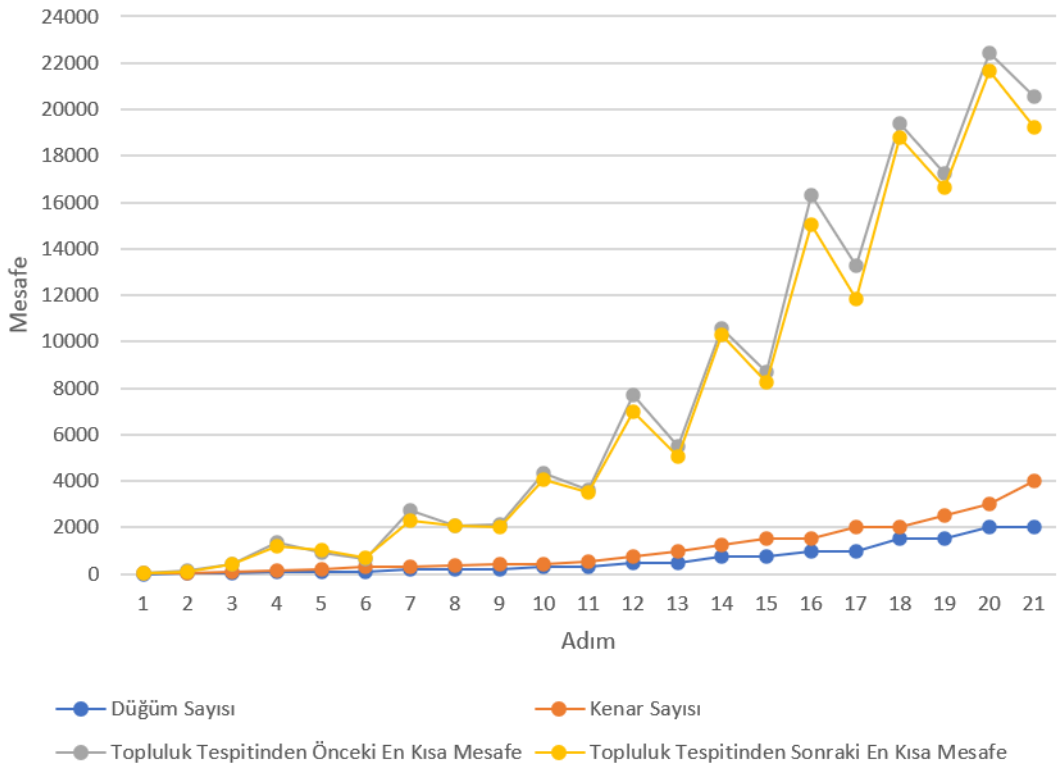
Tez çalışması kapsamında geliştirilen algoritma Python dili kullanılarak geliştirilmiştir. Kenar ve köşe sayısı belli olan bir çizge oluşturmak için Pyhton'ın 'Random' kütüphanesi kullanılmıştır. İlk olarak, elde edilen çizgedeki tüm düğümleri ziyaret ederek çizge içerisindeki en kısa yolu bulmak için Gezgin Satıcı Probleminin çözümünde kullanılan Christofides Algoritması çizge topluluklara ayrılmadan en kısa yol bulunmuştur. Daha sonra oluşturulan çizgeyi Louvain algoritması kullanılarak topluluklara ayırma işlemi gerçekleştirilmiştir. Oluşan topluluklar birer alt çizge olarak düşünülebilir. Bu çizgelerin her birine tekrardan Christofides Algoritması uygulanmış ve sonuç çizgenin ilk haliyle elde edilen sonuçla kıyaslanmıştır. Sonuçların görselleştirilmesi için ise Python dilinin matplotlib kütüphanesi kullanılmıştır. Bu tez kapsamında gerçekleştirilen işlemler Intel Core i5-6300 işlemcisi ve 8 GB ram ile Windows işletim sistemi üzerinde gerçekleştirilmiştir. Algoritmaların çalışma prensiplerinin farklı olmasından ve düğüm sayısının artması algoritmaların çalışmasına direkt etkisi olmasından dolayı düğüm sayısı ve kenar arttırılarak algoritmanın en iyi sonucu bulma performansı Çizelge 5.1'de gösterilmiştir. Çizelgelerin kolonları soldan sağa Düğüm Sayısı, Kenar Sayısı, Topluluk Tespiti Yapılmadan Önceki En Kısa Yol ve Topluluk Tespiti Yapıldıktan Sonraki En Kısa Yol olarak sıralanmıştır.

Çizelge 5.1 Çalışma Kapsamında Elde Edilen Sonuçlar

Düğüm Sayısı	Kenar Sayısı	Topluluk Tespitinden Önceki En Kısa Mesafe	Topluluk Tespitinden Sonraki En Kısa Mesafe
10	20	34	44
20	40	123	120
50	100	411	451
100	150	1344	1194
100	200	923	1024
100	300	653	714
200	300	2731	2273
200	350	2095	2083
200	400	2117	2016
300	450	4331	4039
300	550	3640	3524
500	750	7721	6963
500	1000	5493	5073
750	1250	10571	10311
750	1500	8695	8249
1000	1500	16298	15019
1000	2000	13284	11853
1500	2000	19421	18769
1500	2500	17235	16623
2000	3000	22452	21678
2000	4000	20532	19215



Şekil 5.1 Topluluk tespitinden önceki sonraki en kısa yol grafiği



Şekil 5.2 Bütün bileşenlerin grafiği

6. TARTIŞMA VE SONUÇ

Tez çalışması kapsamında çizgelere parçala ve fethet yöntemi mantığıyla topluluk tespiti uygulanarak katedilecek toplam mesafenin daha kısa bir hale getirilip getirilemeyeceği üzerine çalışmıştır. İncelemeler kapsamında düğüm sayısı ve kenar sayısının orantılı değişiminin sonuçları etkilediği görülmüştür. Öncelikli olarak düğüm sayısı ve kenar sayısı az olduğunda topluluk tespiti uygulayarak en kısa yolu bulmanın bir artı etkisi olmadığı görülmüştür. Ancak düğüm sayısı ve kenar sayısının artış oranı sabit tutularak düğüm sayısı ve kenar sayısı arttırıldıkça topluluk tespiti uygulamanın önemli avantajlar getirdiği görülmüştür. Düğüm sayısı ne kadar artarsa en kısa yol bulmak algoritmasının da aynı oranda artmasa bile topluluk tespiti uygulanmadan önceki haline göre daha verimli olduğu görülmüştür. Diğer yandan düğüm sayısı sabit tutulup kenar sayısının arttırıldığı durumlarda ise artan kenar sayısının çizgede topluluk tespiti uygulanmadan da gidilecek yolu kısalttığı görülmüştür. Bunun nedeni, kenar sayısının artmasının gidilecek yol sayısını da arttırdığı için daha kısa ve alternatif yolların ortaya çıkmasına neden olduğu söylenebilir. Ayrıca kenar sayısı arttırılıp topluluk tespiti uygulandığında da çizgenin ilk haline göre yine daha kısa yollar bulunduğu görülmüştür. Genel olarak değerlendirildiğinde düğüm sayısı ve kenar sayısı, grafiklerdeki (ağlardaki) en kısa yolu bulmaya önemli ölçüde etki eder. Düğüm sayısı ve kenar sayısının en kısa yolu bulmaya olan etkisi şu şekilde açıklanabilir:

Düğüm Sayısı Etkisi:

- Düşük düğüm sayısı: Düşük düğüm sayısına sahip bir grafik, en kısa yolu bulma işlemini daha hızlı ve daha verimli hale getirir. Çünkü daha az düğüm üzerinde arama yapmak, daha az adım ve hesaplama gerektirir. Ancak çok az düğüm olduğunda, en kısa yolun bulunmasında birçok yol seçeneği olmayabilir ve bu da en kısa yolun benzersiz olmama olasılığını artırabilir.

- Yüksek düğüm sayısı: Yüksek düğüm sayısına sahip bir grafik, en kısa yolu bulma işlemini daha karmaşık hale getirir. Çünkü daha fazla düğüm üzerinde arama yapmak, daha fazla adım ve hesaplama gerektirir. Bu nedenle, yüksek düğüm sayısına sahip grafiklerde en kısa yolu bulma süresi ve işlem gücü artabilir.

Kenar Sayısı Etkisi:

- Düşük kenar sayısı: Az kenar sayısına sahip bir grafik, en kısa yolu bulma işlemini daha hızlı ve daha verimli hale getirir. Daha az kenar, daha az bağlantı ve daha az yol seçeneği anlamına gelir. Ancak, düşük kenar sayısında, bazı düğümler arasında direkt bağlantılar olmaması durumunda en kısa yolun bulunmasında zorluklar yaşanabilir.

- Yüksek kenar sayısı: Yüksek kenar sayısına sahip bir grafik, en kısa yolu bulma işlemini daha karmaşık hale getirir. Daha fazla kenar, daha fazla bağlantı ve daha fazla yol seçeneği anlamına gelir. Bu durumda, en kısa yolu bulma süresi ve işlem gücü artabilir.

Sonuç olarak, düğüm sayısı ve kenar sayısı, grafiklerdeki en kısa yolu bulmaya önemli ölçüde etki eder. Daha büyük ve karmaşık grafiklerde, en kısa yolu bulma işlemi daha zaman alıcı ve hesaplama gücü gerektiren bir süreç olabilir.

Kenar sayısı ve düğüm sayısının ayrı ayrı sonuca etkisine ek olarak düğüm sayısının kenar sayısına oranı, çizgelerdeki en kısa yolu bulmaya etki eden önemli bir faktördür. Bu oran, grafiklerin yapısını ve bağlantı yoğunluğunu belirleyen bir ölçüdür. Bu etkiyi şu şekilde inceleyebiliriz:

Düğüm Sayısı ve Kenar Sayısı Oranı = (Düğüm Sayısı) / (Kenar Sayısı)

1. Düşük Oranlı Grafikler:

- Düğüm sayısı, kenar sayısına göre daha az: Bu durumda, grafiklerde düğümler arasında göreceli olarak daha az bağlantı vardır. Bu tür düşük oranlı grafiklerde, en kısa yolu bulma işlemi genellikle daha hızlı ve daha verimli olur. Çünkü daha az bağlantı, daha az seçenek ve yol oluşturur, bu da daha az hesaplama ve arama işlemi gerektirir.

2. Yüksek Oranlı Grafikler:

- Düğüm sayısı, kenar sayısına göre daha fazla: Bu durumda, grafiklerde düğümler arasında göreceli olarak daha fazla bağlantı vardır. Yüksek oranlı grafikler, daha yoğun

bir yapıya sahiptir ve bağlantılar arasında daha fazla seçenek ve yol bulunur. Bu tür grafiklerde, en kısa yolu bulma işlemi daha karmaşık hale gelir ve daha fazla hesaplama ve arama işlemi gerektirebilir.

3. Denge Oranlı Grafikler:

- Düğüm sayısı ve kenar sayısı dengeli: Bu durumda, grafiklerde düğüm sayısı ve kenar sayısı arasında bir denge vardır. Denge oranlı grafikler, en kısa yolu bulma işleminde orta seviyede karmaşıklığa sahip olabilir.

Genel olarak, düğüm sayısının kenar sayısına oranı, grafiklerdeki en kısa yolu bulmaya etki ederken, ağın yapısını ve bağlantı yoğunluğunu belirler. Düşük oranlı grafiklerde daha basit ve hızlı çözümler elde etmek daha olasıdır, ancak yüksek oranlı grafiklerde daha karmaşık ve zaman alıcı işlemler gerekebilir. Bu nedenle, en kısa yolu bulma algoritması seçerken ağın yapısını ve özelliklerini dikkate almak önemlidir. Ayrıca, büyük ve karmaşık grafiklerde daha verimli algoritmalar ve hesaplama yöntemleri kullanmak, en kısa yolu bulma sürecini iyileştirebilir.

Tez kapsamında kullanılan Christofides algoritmasının düğüm ve kenar sayısının artmasıyla daha iyi sonuçlar verdiği görülmüştür. Christofides algoritmasının çok tercih edilmesinin nedenlerinden biri de optimum çözüm uzunluğunun 3/2 faktörü içinde olacağını neredeyse garanti etmesi olduğu söylenebilir.

Gelecekte yapılabilecek çalışmalar için farklı Gezgin Satıcı Problemi algoritmaları kullanılarak problemi farklı bir şekilde ele almak mümkündür. Ayrıca, bu tez kapsamında kullanılan topluluk tespiti algoritmasının da değiştirilmesi farklı çalışma konularının önünü açabilir. Her geçen yıl bu alanda geliştirilen algoritmaların iyileştirilmesi ve yeni yöntemlerin bulunması olası olduğu için bu çalışma kapsamında elde edilen sonuçların iyileştirilmesi olası olarak düşünülebilir.

Sonuç olarak bu tez çalışması kapsamında düğüm ve kenar sayısı bilinen biz çizge uygulanarak Christofides algoritması uygulanmış ve en kısa yol bulunmuştur. Daha sonra

Louvain algoritması kullanılarak topluluk tespiti yapılmış ve oluşan alt çizgelere de aynı yöntem uygulanarak en kısa yol bulunmuş ve sonuçlar kıyaslanmıştır. Köşe sayısının ve kenar sayısının artışı ile topluluk tespiti yapılarak bulunan sonuçların daha kısa olduğu bulunmuş ve köşe sayısının kenar sayısına oranının da bu sonuçlarda önemli bir etkisi olduğu görülmüştür.

KAYNAKLAR

A. Bar-Noy, S. Khuller, and B. Schieber. The complexity of finding most vital arcs and nodes. Technical Report CS-TR-3539, Institute for Advanced Studies, University of Maryland, College Park, MD, 1995.

A. Brander and M. Sinclair. A comparative study of *K*-shortest path algorithms. In *Proc. of 11th UK Performance Engineering Workshop*, pages 370–379, 1995.

B. Fortz and M. Thorup. Internet traffic engineering by optimizing OSPF weights. In *INFOCOM*, pages 519–528, 2000.

Bang-Jensen, Jørgen; Gutin, Gregory (2018), "Basic Terminology, Notation and Results", in Bang-Jensen, Jørgen; Gutin, Gregory (eds.), Classes of Directed Graphs, Springer Monographs in Mathematics, Springer International Publishing, pp. 1–34, doi:10.1007/978-3-319-71840-8_1; see p. 17

Beck, M.; Blado, D.; Crawford, J.; Jean-Louis, T.; Young, M. (2013), "On weak chromatic polynomials of mixed graphs", Graphs and Combinatorics, arXiv:1210.4634, doi:10.1007/s00373-013-1381-1

Bisen, S.K. Application of Graph Theory in Transportation Networks. Int. J. Sci. Res. Manag. 2017, 5, 10–12.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R. & Lefebvre, E. Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment 2008, P10008 (2008).

Campbell,W.M.; Dagli, C.K.;Weinstein, C.J. Social network analysis with content and graphs. Linc. Lab. J. 2013, 20, 61–81.

Chen, Wai-Kai (1997). Graph Theory and its Engineering Applications. World Scientific. pp. 29. ISBN 978-981-02-1859-1.

Chrysochoos, A.; Louche, H. An infrared image processing to analyse the calorific e_ects accompanying strain localisation. Int. J. Eng. Sci. 2000, 16, 1759–1788.

Clauset, A., Newman, M. E. & Moore, C. Finding community structure in very large networks. Physical Review E 70, 066111 (2004).

Cordero, P.; Enciso, M.; Mora, A.; Ojeda-aciego, M.; Rossi, C. Knowledge-Based Systems Knowledge discovery in social networks by using a logic-based treatment of implications. Knowl.-Based Syst. 2015, 87, 16–25.

D. Eppstein. Finding the *k* shortest paths. *SIAM J. Computing*, 28(2):652–673, 1998.

Danon, L., Díaz-Guilera, A. & Arenas, A. The effect of size heterogeneity on community identification in complex networks. Journal of Statistical Mechanics: Theory and Experiment 2006, P11010 (2006).

Danon, L.; Díaz-Guilera, A.; Duch, J.; Arenas, A. Comparing community structure identification. J. Stat. Mech. Theory Exp. 2005, 2005, P09008.

De Klerk, E. Exploiting special structure in semidefinite programming: A survey of theory and applications. Eur. J. Oper. Res. 2010, 201, 1–10.

Demange, M.; Ekim, T.; de Werra, D. Discrete Optimization A tutorial on the use of graph coloring for some problems in robotics. Eur. J. Oper. Res. 2009, 192, 41–55

Derrible, S.; Kennedy, C. Network Analysis of World Subway Systems Using Updated Graph Theory. Transp. Res. Rec. 2009, 2112, 17–25.

Durand, G.; Belacel, N.; Laplante, F. Graph theory based model for learning path recommendation. Inf. Sci. 2013, 251, 10–21.

E. Nardelli, G. Proietti, and P. Widmayer. Finding the most vital node of a shortest path. In *Proc. COCOON*, 2001.

Fletcher, Peter; Hoyle, Hughes; Patty, C. Wayne (1991). Foundations of Discrete Mathematics (International student ed.). Boston: PWS-KENT Pub. Co. p. 463. ISBN 978-0-53492-373-0. A weighted graph is a graph in which a number w(e), called its weight, is assigned to each edge e.

Fortunato, S.: Community detection in graphs. J. Phys. Rep. 486(3–5), 75–147 (2010) Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. J. Phys. Rev. E 70(6), 066111 (2004)

Freeman, L. C. Centrality in social networks conceptual clarification. Social Networks 1, 215–239 (1979)

Ganzha, M.; Maciaszek, L. Position Papers of the 2019 Federated Conference on Computer Science and Information Systems; Springer: Leipzig, Germany, 2019; p. 19.

Gärtner, T.; Flach, P.; Wrobel, S. On graph kernels: Hardness results and e_cient alternatives. Lect. Notes Artif. Intell. (Subser. Lect. Notes Comput. Sci. 2003, 2777, 129–143.

Girvan, M. & Newman, M. E. Community structure in social and biological networks. Proceedings of the National Academy of Sciences 99, 7821–7826, 2002.

Girvan, M., Newman, M.: Community structure in social and biological networks. PNAS 99, 7821–7826 (2002)

Iturria-medina, Y.; Sotero, R.C.; Canales-rodríguez, E.J.; Alemán-gómez, Y.; Melie-garcía, L. Studying the human brain anatomical network via di_usion-weighted MRI and Graph Theory. Neuroimage 2008, 40, 1064–1076.

J. Hershberger and S. Suri. Vickrey prices and shortest paths: What is an edge worth? In *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 252–259, 2001.

J. Hershberger, M. Maxel, and S. Suri. Finding the *k* Shortest Simple Paths: A New Algorithm and its Implementation. To appear in *Proc. of ALENEX*, 2003.

J. Scott, P.J. Carrington, The SAGE Handbook of Social Network Analysis, SAGE publications, 2011.

J. Xie, B.K. Szymanski, X. Liu, Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process, in: Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on, IEEE, 2011, pp. 344e349.

J. Y. Yen. Finding the *K* shortest loopless paths in a network. *Management Science*, 17:712–716, 1971.

Jaromczyk, J.W.; Lonc, Z.; Truszczy, M. Constructions of asymptotically shortest k-radius sequences. J. Comb. Theory Ser. A 2012, 119, 731–746.

John Hershberger, Subhash Suri, and Amit Bhosle, On the Difficulty of Some Shortest Path Problems, January, 2013

Journal, I.; Core, O.; Ijcem, M. A study of Vertex—Edge Coloring Techniques with Application. Int. J. Core Eng. Manag. 2014, 1, 27–32.

K. Malik, A. K. Mittal, and S. K. Gupta. The *k* most vital arcs in the shortest path problem. *Oper. Res. Letters*, 8:223–227, 1989.

Kaundal, K. Applications of Graph Theory in Everyday Life and Technology. Imp. J. Interdiscip. Res. 2017, 3, 892–894.

Kocay, W.; Kreher, D.L. Graphs, Algorithms, Optimization; Chapman & Hall/CRC Press: Boca Raton, FL, USA, 2017; pp. 1–483.

Lancichinetti, A. & Fortunato, S. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. Physical Review E 80, 016118, 2009.

Lee, J. Kinematic Analysis of Tendon-Driven Robotic Mechanisms Using Graph Theory. ASME J. Mech. Trans. Automat. DXes. 1989, 111, 59–65.

M. O. Ball, B. L. Golden, and R. V. Vohra. Finding the most vital arcs in a network. *Oper. Res. Letters*, 8:73–76, 1989.

M. Potamias, F. Bonchi, C. Castillo, A. Gionis, Fast shortest path distance estimation in large networks, in: Proceedings of the 18th ACM Conference on Information and Knowledge Management, ACM, 2009, pp. 867e876.

Maoguo Gong, Guanjun Li, Zhao Wang, Lijia Ma, Dayong Tian , An efficient shortest path approach for social networks based on community structure 2016

Minor, E.S.; Urban, D.L. A Graph-Theory Framework for Evaluating Landscape Connectivity and Conservation Planning. Conserv. Biol. 2008, 22, 297–307.

Mondal, B.; De, K. Overview Applications of Graph Theory in Real Field. Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol. 2017, 2, 751–759.

Mukherjee, A., Choudhury, M., Peruani, F., Ganguly, N. & Mitra, B. Dynamics On and Of Complex Networks, Volume 2: Applications to Time-Varying Dynamical Systems (Springer Science & Business Media, 2013).

Nagendram, N.V. A Note on Su_cient Cindition on Hamiltonian Path to Complete Graphs (SC-HPCG). IJMA 2011, 2, 1–6.

Newman, M. E. Finding community structure in networks using the eigenvectors of matrices. Physical Review E 74, 036104 (2006).

Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. J. Phys. Rev. E 69(2), 026113 (2004)

Plummer, M.D. Some covering concepts in graphs. J. Comb. Theory 1970, 8, 91–98.

Polak, M.; Roma, U. On the applications of Extremal Graph Theory to Coding Theory and Cryptography. Electron. Notes Discret. Math. 2013, 43, 329–342.

Pons, P. & Latapy, M. Computing communities in large networks using random walks. In Computer and Information Sciences-ISCIS 2005, 284–293 (Springer, 2005).

Qiantt, Y.; Suent, C.Y.; M, Q.H.G. Clustering Combination Method. In Proceedings of the 15th International Conference on Pattern Recognition, Barcelona, Spain, 3–7 September 2000; pp. 732–735.

Raghavan, U. N., Albert, R. & Kumara, S. Near linear time algorithm to detect community structures in large-scale networks. Physical Review E 76, 036106 (2007).

Robertson, N.; Seymour, P.; Thomas, R. Quickly excluding a planar graph. J. Comb. Theory Ser. B 1994, 62, 323–348.

Rosvall, M. & Bergstrom, C. T. An information-theoretic framework for resolving community structure in complex networks. Proceedings of the National Academy of Sciences 104, 7327–7331 (2007).

Rosvall, M., Axelsson, D. & Bergstrom, C. T. The map equation. The European Physical Journal Special Topics 178, 13–23 (2010).

Saerens, M.; Fouss, F.; Yen, L.; Dupont, P. The Principal Components Analysis of a Graph, and Its Relationships to Spectral Clustering. In Proceedings of the European conference on machine learning, Pisa, Italy, 20–24 September 2004; pp. 371–383.

Salembier, P.; Garrido, L. Binary Partition Tree as an E_cient Representation for Image Processing, Segmentation, and Information Retrieval. IEEE Trans. Image Process. 2000, 9, 561–576.

Sarma, S.V.M. Applications of Graph Theory in Human Life. Int. J. Comput. Appl. 2012, 1, 21–30.

Sciences, D.M. A Survey on some Applications of Graph Theory in Cryptography. J. Discret. Math. Sci. Cryptogr. 2015, 18, 209–217.

Shuman, D.I.; Narang, S.K.; Frossard, P.; Ortega, A.; Vandergheynst, P. The Emerging Field of Signal Processing. IEEE Signal Process. Mag. 2013, 30, 83–98.

Siqueira, S.; Eduardo, C.; Junior, B.; Comfort,W.E.; Rohde, L.A.; Sato, J.R. Abnormal Functional Resting-State Networks in

Tyagi, S.S. Statical Analysis of Social Network; JUIT (Jaypee university of information technology): Himachal Pradesh, India, 2014; pp. 1–99.

Voloshin, V.I. Introduction to Graph Theory; Nova Science Publishers: New York, NY, USA, 2009; pp. 1–144.

Xie, J. & Szymanski, B. K. Community detection using a neighborhood strength driven label propagation algorithm. In Network Science Workshop 188–195 (IEEE, 2011).

Yuan, M.; Chen, L.; Yu, P.S.; Mei, H. Privacy preserving graph publication in a distributed environment. World Wide Web 2015, 18, 1481–1517.

Zachary, W. W. An information flow model for conflict and fission in small groups. Journal of Anthropological Research 452–473, 1977.