
GAZİ ÜNİVERSİTESİ
Gazi
1926

AN EVOLUTIONARY MULTI-OBJECTIVE APPROACH TO UNCOVER COMMUNITIES IN DYNAMIC NETWORKS

Kainat Akhtar TARRAR

MASTER'S DEGREE

DEPARTMENT OF COMPUTER SCIENCE

GAZI UNIVERSITY

INSTITUTE OF INFORMATICS

February 2024

Kainat Akhtar TARRAR tarafından hazırlanan “DİNAMİK AĞLARDAKİ TOPLULUKLARI ORTAYA ÇIKARMAK İÇİN EVRİMSEL ÇOK AMAÇLI YAKLAŞIM” adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Biligisayar bilimleri Ana Bilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Dr.Öğr.Üyesi Yılmaz ATAY
Biligisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi
Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ...……

Başkan: Dr.Öğr.Üyesi Feyza YILDIRIM OKAY
Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi
Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ……...

Üye: Doç.Dr. Özkan İNİK
Bilgisayar Donanımı Ana Bilim Dalı, Tokat Gaziosmanpaşa Üniversitesi
Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ……...

Tez Savunma Tarihi: 29/02/2024

Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

……..…….
Prof. Dr. Aslıhan TÜFEKCİ
Bilişim Enstitüsü Müdürü

ETHICAL STATEMENT

In this thesis study, which I prepared in accordance with Gazi University Informatics Institute Thesis Writing Rules;

- I have obtained the data, information and documents I have presented in the thesis within the framework of academic and ethical rules,
- I present all information, documents, evaluations and results in accordance with scientific ethics and morals,
- I have cited all the works I have benefited from in the thesis by making appropriate references,
- I have not made any changes to the data used,
- The work I have presented in this thesis is original,

Otherwise, I declare that I accept all loss of rights that may arise against me.

Kainat Akhtar TARRAR

29/02/2024

DİNAMİK AĞLARDAKİ TOPLULUKLARI ORTAYA ÇIKARMAK İÇİN EVRİMSEL ÇOK AMAÇLI YAKLAŞIM

(Yüksek Lisans Tezi)

Kainat Akhtar TARRAR

GAZİ ÜNİVERSİTESİ

BİLİŞİM ENSTİTÜSÜ

Şubat 2024

ÖZET

1. GİRİŞ

Bir ağ, bağlı bileşenlerin grafiksel bir temsilini yansıtan, düğümleri (aynı zamanda köşeler olarak da bilinir) birbirine bağlayan çizgilerle gösterilen bir temsil, ağ haritası, düğüm-bağ diyagramı veya basit bir grafik olarak nitelendirilen matematiksel bir yapıdır. Ağ içindeki nesneler genellikle düğümler olarak adlandırılır ve grafikte noktalar ile gösterilir. Bu düğümler arasındaki bağlantılar, iki veya daha fazla düğümü birbirine bağlayan çizgiler olarak adlandırılan kenarlar tarafından temsil edilir. Ağlar, bilgisayar sistemlerinin veya cihazlarının düğümler olarak temsil edildiği ve bunlar arasındaki bağlantıların kenarlar olarak gösterildiği internet gibi çeşitli sistemleri modellemek için kullanılabilir.

Bu çalışma, çeşitli bağlantı türleriyle bağlanmış bireylerden veya gruplardan oluşan sosyal ağlara odaklanmaktadır. Dinamik ağlar, zamana bağlı olarak her bir zaman dilimi için farklı bir grafikle ilişkilidir. Dinamik ağlarla ilgili olan bu alan, Dinamik Ağ Analizi (DNA/Dynamic Network Analysis) olarak adlandırılır. Bu alanda iki önemli yön vardır. Bunlardan ilki, DNA bilgi analizinin istatistiksel bir analizidir. Diğeriyse ağ dinamiklerinin sorunlarını çözmek için simülasyonun kullanılmasıdır. Dinamik ağlar, geleneksel sosyal ağlardan önemli ölçüde farklıdır. Çünkü bunlar dinamik, daha karmaşık, çoklu, çok modlu ağlardır ve aynı zamanda farklı belirsizlik düzeylerine sahip olabilirler. Başka bir önemli fark da DNA'nın sosyal özellikleri ile ilgili yapısal etkileşimlerin ve ağ davranışının hesaba katılmasıdır.

Bir DNA'nın, belirli bir zaman diliminde veri setindeki bir değişkenin anlık incelenmesini sunması gerekmektedir. Bilgisayar simülasyonu perspektifinden bakıldığında, DNA'daki düğümler kuantum teorisinin atomları gibi davranır ve olasılıksal olabilirler. Sosyal Ağ Analizi'nde (SNA/Social Network Analysis), düğümler sabittir. DNA modelinin düğümleri

zaman içindeki özellik değişikliklerine göre öğrenme yeteneğine sahiptir. Örneğin, bir şirketin yüksek mevkide olan bir çalışanı yerine daha çok çalışan ve hak eden başka birini getirdiğimiz örnekte bilgi ve çok çalışmayla yerlerinin değişebildiğini görebiliriz. Başka bir örnekte ise bir terörist yakalandığında ilgili olduğu gruplar arasındaki bağ ve bilgi sayesinde diğer teröristlerin de yakalanmasını sağlanabilir. DNA'daki değişim, ağ içinde bir düğümden diğerine geçer ve bu işlem iteratif olarak devam eder.

Sampson'un Manastır Çalışması, DNA'nın en dikkate değer vakalarından biridir. Sampson'un Manastır Çalışması, Robert J. Sampson tarafından 1960'ların ikinci yarısında ve 1970'lerin ilk yarısında gerçekleştirilen bir araştırma çalışmasıdır. Bu, dinamik ağ analizi (DNA/Dynamic Network Analysis) alanında önemli bir çalışma olarak kabul edilir ve birkaç dergi makalesinde yayımlanmıştır. Bu makaleler, çalışmanın yöntemlerinin, bulgularının ve dinamik sosyal ağların anlaşılmasındaki etkilerinin derinlemesine analizini sunmaktadır.

Konu açıklaması

Bu çalışma; farklı ağların incelenmesi ve topluluk tespiti araştırmasını içerir. Bir topluluk, özelliklerinde yüksek benzerlik gösteren düğüm kümesinin, bir araya gelerek bir bütün oluşturduğu zaman ortaya çıkar. Bir topluluktaki düğümler, diğer topluluklardaki düğümlere göre birbirleriyle daha yoğun etkileşimde bulunurlar. Bu etkileşimler, düğüm katılımcıları arasındaki benzerlik veya mesafe ölçüm yöntemleri kullanılarak ortaya çıkarılabilir. Dinamik sosyal ağlarda topluluk tespiti, ağ yapısının değişmeye devam ettiği bir süreçtir. Burada, ağ yapısı değişmeye devam ettiği için topluluk yapısındaki değişiklikleri etkiler. Düğümler farklı topluluklara katılırken ve ayrılırken, topluluklar zamanla birleşebilir veya bölünebilir. Bu nedenle, dinamik sosyal ağlarda topluluk tespiti, zaman içinde istikrarlı olan toplulukları tanımlamayı içerirken ve aynı zamanda ağ yapısındaki değişiklikleri yakalamayı içerir. Sosyal ağlar, düğümlerin bireyleri, kuruluşları veya diğer varlıkları gösterdiği ve kenarların aralarındaki ilişkiyi temsil ettiği grafikler olarak modellenir. Bu ilişkiler, iş birliği, arkadaşlık veya ilişki gibi çeşitli türlerde olabilir.

Problem tanımı

Toplulukları tanımlamak, ağların mimarisini ortaya çıkarmak ve özellikleri ile eylemlerini daha kapsamlı bir şekilde anlamak için hayati bir yönü temsil eder. Ağ içindeki toplulukları tanımlamak için bir dizi algoritma ve teknik bulunmaktadır. Bunların birçoğu bir veya daha fazla amaç fonksiyonunu optimize etmek için optimizasyon yöntemlerini kullanmaktadır.

Bu fonksiyonlar arasında, Modülerlik ve NMI (Normalized Mutual Information) en yaygın kullanılanlardandır.

Dünya genelinde bireylerin ve toplulukların küresel ölçekte birbirine bağlı olduğu geniş bir köy haline geldiği yaygın bir kabul görmektedir. Sosyal Ağ Analizi alanında, bir ağ içindeki toplulukların bileşimi son derece önemlidir. Araştırmacılar, bir ağ içinde oluşturulan toplulukları ve kümeleri tespit etmek ve gizli bilgileri ortaya çıkarmak için çeşitli yöntemleri keşfetmektedirler. Ancak geleneksel topluluk tespiti yöntemleri, çalışılan ağın tamamen anlaşılmasını gerektirir, bu da çoğu gerçek dünya ağı için yetersiz kalmaktadır. Büyük çevrimiçi sosyal ağların yükselmesi ile araştırmacılar, yalnızca yerel bilgi gerektiren topluluk tespiti yöntemleri geliştirmekte zorlanmaktadır. Topluluk tespiti birçok açıdan hayati öneme sahiptir. Yapısal konumlarına dayanarak düğümleri sınıflandırmayı mümkün kılar. Gerçek ağlar içinde var olan hiyerarşik yapının ortaya çıkmasını sağlar ve ağ verilerinin analiz, işleme ve depolama süreçlerinin verimliliğini arttırmaktadır. Ayrıca, topluluk tespitinin birçok pratik uygulaması vardır. Sosyal ağ çalışmalarında, topluluklar ortak ilgi, hobiler, bağlantılar veya geçmişlere sahip insan kümelerini tasvir edebilir. Alışveriş ağlarında, topluluklar benzer alışveriş alışkanlıklarına sahip tüketicilerin kümelerini simgeliyebilir ve etkili öneri sistemlerinin oluşturulmasına yardımcı olabilir. Tıp biliminde, beyin ağları, birbirine bağlı düğüm topluluklarına sahip olabilir, bu da beynin yapısal bölümlerine dair yerel hesaplamaların yapılmasına ve hastalığın tedavisine dair içgörü sağlayabilir.

Dinamik sosyal ağlardaki topluluk tespiti, araştırmacılar için çeşitli zorluklar ortaya çıkarır. Bu zorluklardan bazıları şunlardır:

- Veri seyrekliği: Dinamik sosyal ağlardaki kenarlar seyrektir. Örneklemek gerekirse bazı düğümler birbirine bağlı olmayabilir. Bu da tamamen bağlı olmayan bir ağa yol açabilir. Seyreklik, topluluk tespitinde karmaşıklığa ve yanlış sonuçlara yol açabilir.
- Ölçeklenebilirlik: Dinamik sosyal ağlar son derece büyük, milyonlarca düğüm ve kenara sahip olabilir. Bu nedenle, topluluk tespitinin son derece büyük verilerle başa çıkabilecek kadar ölçeklenebilir olması çok önemlidir.
- Dinamikler: Ağın yapısı zaman içinde hızla değişebilir, bu da istikrarlı toplulukların belirlenmesinde büyük bir zorluk oluşturur.
- Gürültü: Ağda rastgele olaylardan veya hatalı verilerden kaynaklanan gürültü olabilir. Bu gürültü, gerçek toplulukların tespitini zorlaştırabilir.

Araştırmanın amacı

Bir ağ içindeki toplulukların analiz edilmesi; yapıları, etkileşimleri, paylaşılan özellikleri ve ortak noktaları hakkında değerli bilgiler sağlayabilir. Bu anlayış, belirgin ve net tanımlanmış topluluklar tarafından bağlı bir dünya yaratmada pratik uygulamalara sahip olabilir. Bu topluluklar içindeki etkileşimleri ve ilişkileri inceleyerek, sosyal bağlantıları optimize etmek ve daha büyük işbirliği anlayışını teşvik etmek bu şekilde mümkün hale gelir. Sonuç olarak, bir ağdaki toplulukların analizi, daha bütünlük arz eden ve birbirine bağlı bir küresel topluma katkıda bulunabilir.

Bu araştırmanın temel amacı, dinamik bir zaman içinde benzer ilgi, ilişki veya davranışlara sahip istikrarlı bir düğüm grubunu tanımlamaktır. Sosyal ağlar doğası gereği dinamik ağlardır ve düğümler arasındaki bağlantılar zaman içinde değişir. Bu nedenle, ağın yapısı evrildiğinde bile istikrarlı toplulukları bulmak önemlidir. Bu araştırma, dinamik sosyal ağlardaki topluluk tespiti üzerine odaklanarak, ağ değişikliklerini hesaba katarak bu istikrarlı toplulukları tespit eden bir algoritma geliştirmeyi amaçlamaktadır. Bu algoritmalar, belirli bir zaman diliminde bölünen, birleşen veya ortadan kaybolan toplulukları bulacak ve büyük karmaşık sosyal ağları işleme yeteneğine sahip olacaktır.

Araştırmanın önemi

Dinamik sosyal ağlardaki toplulukların tespiti, hemen hemen tüm ülkelerden milyonlarca insanın çeşitli yollarla bağlı olduğu ve hızla büyüyen küresel toplumda önemli bir yer tutmaktadır. Dinamik sosyal ağlardaki topluluk tespitinin en önemli uygulamalarından biri, sosyal medya analizini içermesidir. Araştırmacılar, benzer ilgi veya inançlara sahip kişileri tespit etmek için topluluk tespit algoritmalarını kullanmaktadır. Örneğin hobiler, dini inançlar veya siyasi bağlılıklar gibi ilişkileri düşünebiliriz. Bu topluluklardan elde edilen bilgiler, kişilere yönelik reklam geliştirmekte ve sosyal medya platformlarındaki bilgi akışını daha iyi anlamakta yardımcı olabilir.

Topluluk tespitinin dinamik sosyal ağlardaki diğer bir önemli uygulaması da sağlık ve tıp alanındaki çalışmalardır. Araştırmacılar, topluluk tespit yöntemlerini aynı sağlık sorunlarına veya risklere sahip grupları bulmak için kullanmaktadır. Bu bilgiler, ilgili sağlık müdahalelerini yapmak veya 2019'da ortaya çıkan Koronavirüs gibi bulaşıcı hastalıkların yayılması konusunda daha iyi bir anlayış sağlamak için kullanılabilir.

Bu uygulamaların yanı sıra, sosyal ağlardaki topluluk tespiti, pazarlama, finans, araştırma ve ulaşım gibi çeşitli alanlarda faydalıdır. Araştırmacılar, ağlardaki istikrarlı toplulukları tespit ederek, insanların veya organizasyonların davranışları, öncelikleri ve eylemleri hakkında derin bir anlayışa sahip olabilir, bu da bu ağlarla ilgili önemli altta yatan bilgilerin ortaya çıkmasına yol açar.

Araştırmanın katkıları

Bu çalışmanın dikkate değer katkıları şunlardır:

- Yenilikçi Algoritma: NSGA-II'nin (Deb et al., 2002) crowding distance mekanizmasında yapılan değişiklik, performansı etkileyici bir şekilde arttırdı. NSGA-II'nin geleneksel crowding distance yönteminin yerini Crowding Distance Elimination (CDE) mekanizması (Liu & Chen, 2019) almıştır. CDE, NSGA-II algoritmasının çeşitliliğini arttırarak etkin bir şekilde non-dominated çözümleri çıkararak bu algoritmanın verimliliğini arttırır.
- Geliştirilmiş Operatörler: NSGA-II'nin crossover operatörü, Cycle Crossover (CX) stratejisi (Puljić & Manger, 2013) kullanılarak geliştirilmiştir. CX stratejisi, çözümdeki yerel optimizasyon sorununu ortadan kaldırmaya yardımcı olmuştur. Ek olarak, Modularity (Girvan & Newman, 2002) ve Normalized Mutual Information (Danon et al., 2005) ile çözümlerin etkin optimizasyonu için üçüncü amaç fonksiyonu olarak Normalized Cut (Dhillon et al., 2004) eklenmiştir. Bu üç önemli amaç fonksiyonunun kullanımı, keşfedilen alt grupların daha doğru ve kapsamlı bir ölçümünü sağlamaktadır.
- Geliştirilmiş Strateji: Transfer tabanlı öğrenmenin uygulanmasıyla, geçmiş nesillerden elde edilen faydalı bilgilerin korunması için önemli ölçüde bir zaman karmaşıklığında azalma elde edilmiştir.

Araştırma varsayımları

Bu araştırma sırasında bazı varsayımlar yapılmıştır. Bu varsayımlar, araştırma çalışması için rehberlik sağlar ve sonuçları anlama çerçevesi sunar. İlgili varsayımlar aşağıda verilmiştir:

- Veri Doğruluğu: Bu araştırma, deneysel amaçlarla kullanılan verinin geçerli olduğunu ve gerçek bir dinamik ağı temsil ettiğini varsayar. Araştırmada kullanılan veri kamuya açık ve diğer araştırmacıların işbirliği ile elde edilmiştir.

- Bilgi Güvenilirliği: Daha önce yapılmış araştırmalardan elde edilen bilgilerin, verilerin ve sonuçların geçerli olduğunu varsayar. Kullanılan algoritmalar ve yöntemler doğru sonuçlar sağlar. Ayrıca, önerilen algoritmayla sonuçların karşılaştırılması için kullanılan algoritmik sonuçların güvenilir olduğu varsayılmıştır.
- Topluluk Stabilitesi: Oluşturulan toplulukların, sosyal ağdaki değişikliklere rağmen zaman içinde nispeten stabil olduğu varsayılmıştır. Algoritma, birçok zaman adımında devam eden stabil toplulukları bulabilmelidir.
- Topluluk Homofiliği: Bir topluluktaki düğümlerin, topluluk dışındaki düğümlere göre birbirine daha bağlı olduğu varsayılmıştır. Bu bağlamda, algoritma homofili konseptine uygun bağlantı desenlerini tespit edebilmelidir.

Araştırma Sınırlamaları

Dinamik sosyal ağlardaki topluluk tespiti üzerine yapılan araştırmalar sırasında ortaya çıkabilecek bazı olası sınırlamalar bulunmaktadır. Bu sınırlamalar, araştırma bilgilerinin geçerliliğini, genelliliğini ve güvenilirliğini etkileyebilir. Bu araştırma çalışmasının olası sınırlamaları aşağıda verilmiştir:

- Veri Kalitesi: Bu araştırmadaki en büyük sınırlamalardan biri kullanılan verinin kalitesidir. Araştırmayı yürütmek için üretilen veri, eksik değerler, gürültü ve yanlış bilgiler içerebilir ve bu durum sonuçların doğruluğunu etkileyebilir.
- Ölçeklenebilirlik: Bu araştırmada topluluk tespiti algoritmalarının ölçeklenebilirliği de bir sınırlamadır. Algoritma, sonuçların analizini yapma yeteneğini sınırlayan sonuçların yanı sıra, sonuçların çarpıtılmasına neden olabilecek çok büyük veri ağları için uygun olmayabilir.
- Yetersiz Gerçek Değer: Topluluk tespiti algoritmasının değerlendirilmesi için net bir gerçek değer veya referans noktasının olmaması durumları olabilir. Gerçek dünya uygulamalarında, ağ bazen her bir düğümün topluluk üyeliğini gösteren önceden tanımlanmış etiketlere sahip olmayabilir. Bu durumlarda, topluluk tespiti algoritmasının değerlendirilmesi için bir gerçek değer kurma konusunda zorluk yaşanabilir. Ayrıca, evrilen ağlar ve örtüşen topluluklar durumunda, gerçek değerin tanımlanması çok zor olabilir.
- Genelliklik: Dinamik sosyal ağlardaki topluluk tespiti araştırmasının sonuçları, diğer topluluk tespiti biçimlerine genellenemeyebilir. Sosyal ağlar; kültür, demografi veya

coğrafi bağlamda çok farklı olabilir. Bu da araştırmanın bulgularının genelliliğine bir sınırlama getirebilir.

Tez organizasyonu

Tez çalışması ayrı bölümler halinde düzenlenmiştir. Bölüm 2, ilgili literatürü ve bu alandaki çalışmaların genel özetini sunmaktadır. Bölüm 3, problem tanımını açıkça sunar ve önerilen yöntemde kullanılan temel kavramların detaylı bir açıklamasını verir. Ayrıca, bu tezde öne sürülen teknik süreçlerin ve yöntemlerin genel taslağını çizer. Bölüm 4, önerilen yöntemin etkinliğini deneysel test ve sonuç sunumu yoluyla değerlendirir. Son olarak, Bölüm 5, tüm tezin bir özetini sunar ve gelecek çalışmalar için kısa bir bakış açısı sağlar.

2. LİTERATÜR TARAMASI

Çok amaçlı evrimsel algoritmalar arasında özellikle Non-dominated Sorting Genetic Algorithm (NSGA-II) yaklaşımı en belirgin olanıdır (Deb et al., 2002). Bu yöntem, geliştirilmiş performansla sonuçlanan çeşitli modifikasyonları içeren NSGA (Bagchi, 1999) üzerine kuruludur. Bu iyileştirmelerden biri, hesaplama karmaşıklığını $O(MN^3)$'ten $O(MN^2)$'ye düşüren elitizm uygulamasıdır.

Feature Transfer-based Multi-Objective Optimization Genetic Algoritması (TMOGA), Zou ve diğerleri tarafından önerilmiştir (Zou et al., 2021). Bu yaklaşım, Baskın Olmayan Sıralama Genetik Algoritması (NSGA-II) algoritmasını temel alır ve performansını artıran modifikasyonlar içerir. TMOGA'nın temel amacı, önceden tanımlanmış topluluk yapılarından değerli özellikler elde etmek ve bu bilgiyi mevcut optimizasyon prosedürüne uygulayarak evrimsel algoritmayı geliştirmektir.

Lin ve ark. dinamik ağlardaki toplulukları keşfetmek için FacetNet (Lin et al., 2008) adlı yeni ve verimli bir yöntem önerdi. Bu yöntem, evrimin zamansal düzgünlüğüne dayalı olarak çıktıyı düzenli hale getirerek, toplulukların tespitini ve evrimini birleştirmesi bakımından benzersizdir. Bu, altta yatan verilere ve onun zamansal gelişimine en uygun toplulukların keşfedilmesiyle sonuçlanır. Yazarlar bu soruna negatif olmayan bir matris çarpanlara ayırma sorunu olarak yaklaştılar ve minimum işlem süresi gereksinimi ile verimli bir çözüme yaklaştılar. Yapılan deneyler, FacetNet'in anlamlı toplulukları etkili bir şekilde ortaya

çıkarabildiğini, ancak elde edilen sonuçlardaki mükemmellik düzeyiyle ilgili iyileştirme alanı olduğunu gösterdi.

Yazarlar Folino ve ark. DYNMOGA (DYNamic Multi-Objective Genetic Algorithms) (Folino & Pizzuti, 2013) olarak adlandırılan dinamik ağlardaki toplulukları bulmak için çok amaçlı bir evrimsel yöntem ortaya koydu. DYNMOGA, topluluk tespiti için çok amaçlı bir problem gibi zamansal pürüzsüzlük kullanır ve her zaman aralığındaki kümelerin kalitesi ile bir aralıktan diğerine ilerleme arasındaki en iyi uzlaşmayı temsil eden çözümler üretmek için genetik algoritma tabanlı bir yöntem kullanır.

Yin ve ark. DYN-MODPSO (Yin et al., 2021) adlı kümelemeye dayalı bir topluluk tespit yöntemi geliştirdi. Yazarlar, kümelemedeki iki ana zorluğu, yani hata birikimine ve sonuçların kaymasına yol açan hata düzeltme eksikliğini ve sonuçların düşük verimliliğine neden olan modülerliğin NP-sertliğini ele aldıklarını iddia ettiler. DYN-MODPSO, bu zorlukların üstesinden gelmek için parçacık sürüsü yöntemini ve evrimsel küme oluşturma işlevini geliştirir. Yazarlar, her döngü sırasında daha çeşitli bireyler oluşturmak için ilk gruplandırma için yeni bir gelecek tahmin tekniği ve popülasyon başlatma prosedüründe tekrarlanmayan bir rastgele yürüyüş tekniği kullanır.

3. MATERYAL VE YÖNTEMLER

Bu bölümde, önerilen yöntemin arka planında kullanılan yaklaşımlar ele alınmaktadır. Önerilen yöntem evrimsel bir algoritmaya dayandığından, evrimsel yaklaşımlar bu bölümde sunulmaktadır.

3.1. Evrimsel Algoritmalar

Evrimsel algoritmalar, doğal süreçlerden ilham alınarak geliştirilen ve canlı organizmaların doğal davranışlarından yararlanılarak çözülen yaklaşım türüdür.

Evrimsel bir algoritmada yer alan ana adımlar aşağıdaki gibidir:

i. Başlatma: Başlatma, bireylerin (bazen değer dizileri veya kromozomlar olarak temsil edilen) rastgele oluşturulduğu bir süreçtir. Her bir çözümdeki birey, verilen bir problemin potansiyel bir çözümünü temsil eder.
ii. Uygunluk değerlendirme: Tüm bireylerin uygunluğu (fitness), bir veya daha fazla amaç fonksiyonu veya fitness ölçü yöntemi kullanılarak kontrol edilir.

Değerlendirme için kullanılan fonksiyon, her bir bireye verilen problemin çözümünde potansiyelini kontrol eder. En iyi fitness değerlerine sahip bireyler daha fazla değerlendirme için seçilir.

iii. Seçim: Bireyler, çoğalma sürecine katılmak üzere fitness değerlerine göre belirlenen bir seçim sürecinden seçilir. En yaygın olarak kullanılan seçim yöntemleri arasında turnuva seçimi, sıra tabanlı seçim ve rulet tekerleği seçimi bulunur. Bu sürecin amacı, en uygun bireyleri seçerek genetik materyallerinin bir sonraki nesile geçme olasılığını arttırmaktır. Bazı seçim süreçleri, uygun olmayan bireylere de bir şans vermek için kullanılır ve evrim sürecine katılmalarına izin verir.

iv. Üreme: Seçim sürecinde seçilen bireyler, soy üretmek için kullanılır. Bu adım genellikle crossover yöntemi olarak bilinir. Bu adımda iki ebeveyn popülasyonun genetik materyali kullanılarak yeni bir popülasyon oluşturulur. Crossover işleminden sonra genellikle mutasyon da uygulanır. Mutasyon uygulama nedeni, çocuk popülasyonuna rastgele değişiklikler yaparak aynı soy popülasyonunun oluşturulmasını önlemektir.

v. Yerine Koyma: Yeni soy, popülasyon içindeki bazı bireylerini değiştirir. Bu genellikle daha düşük bir fitness değerine sahip bireyler üzerinde uygulanır. Bu ek adım, çözümlerin çeşitliliğini koruma ve popülasyon boyutu üzerinde ek bir kontrol sağlama konusunda yardımcı olur.

vi. Sonlandırma: Yukarıda açıklanan süreçler, belirli bir sonlandırma koşulu karşılanana kadar tekrarlanır. Genellikle bu sonlandırma koşulu, maksimum nesil sayısına ulaşmak veya belirli bir eşik değerine ulaşmaktır.

3.1.1. Evrimsel algoritmaların faydaları

Evrimsel algoritmaların çok önemli faydalarından bazıları şunlardır:

- Artan Esneklik: Evrimsel algoritmaların, probleme uygun olarak uyarlanabilme ve değiştirilebilme yeteneği vardır. Bu nedenle, evrimsel algoritmalar, insan yaşamının en karmaşık sorunlarını çözme ve hedeflere başarıyla ulaşma kapasitesine sahiptir.
- Daha İyi Optimizasyon: Birçok çözüm, uygun çözümler olarak kabul edilir ve aralarından en iyisi seçilir. Bu, evrimsel bir algoritmanın belirli bir çözüme sınırlı olmadığını gösterir.

- Sınırsız Çözümler: Klasik algoritmalar bir çözüm bulur ve ardından bu çözümü geliştirmeye odaklanırken, evrimsel algoritmaların birçok çözümü dikkate alınır ve tek bir probleme birden fazla uygun çözüm sunma yeteneğine sahiptir.

3.1.2. Evrimsel algoritmaların uygulamaları

Evrimsel algoritmalar, konu algoritmalar olduğunda prestijli bir yere sahiptir. Evrimsel algoritmaların uyumluluğu ve etkinliği, insan yaşamının muazzam alanlarında yerini almıştır. Evrimsel algoritmaların en yaygın uygulamalarından bazıları aşağıda listelenmiştir:

- Optimizasyon Problemleri.
- Makine Öğrenimi ve Veri Madenciliği.
- Robotik ve Kontrol Sistemleri.
- Görüntü ve Sinyal İşleme.
- Biyoinformatik ve Hesaplamalı Biyoloji.
- Finansal Modelleme ve Portföy Optimizasyonu.
- Oyun Oynama ve Strateji Optimizasyonu.
- Yenilenebilir Enerji Sistemleri.

Yukarıdaki uygulamalar, evrimsel algoritma uygulamalarından birkaçı olarak kabul edilebilir, çünkü çok yönlülükleri ve karmaşık optimizasyon problemlerini çözme yetenekleri, çok sayıda bilimsel hesaplamalı mühendislik görevlerini çözmede önemli bir değere sahip olmalarına izin verir. Bu çalışmada, dinamik sosyal ağlarda topluluk tespiti optimizasyon problemini çözmek için evrimsel algoritmalar kullanılmıştır.

3.1.3. Evrimsel algoritma türleri

Pek çok evrimsel algoritma türü vardır ve hepsinin kendi varyasyonları ve kullanımları vardır. En sık kullanılan evrimsel algoritma türlerinden bazıları aşağıda listelenmiştir:

- Genetik Algoritmalar (GA/Genetic Algorithms)
- Evrimsel Programlama (EP/Evolutionary Programming)
- Evrim Stratejileri (ES/ Evolution Strategies)
- Genetik Programlama (GP/Genetic Programming)
- Diferansiyel Evrim (DE/Differential Evolution)

3.2. Genetik Algoritmalar (GA'lar)

Genetik algoritmalar, doğal seçilim ve genetik kavramlarına dayalı adaptif sezgisel arama algoritmaları türündeki evrimsel algoritmaların bir türüdür. Bu algoritmalar, arama uzayını doğal seçilim ve genetik konseptleri temelinde kullanarak rastgele bir arama uzayını kullanır. Bu algoritmalar, çözüm uzayındaki performansın daha ince bir bölgesine doğru arama uzayının yönlendirilmesi için sağlanan tarihsel verilerin yardımıyla çalışır. Genetik algoritmalar, birçok türde optimizasyon probleminin ve arama probleminin üstün çözümlerini oluşturmak için geniş bir şekilde kullanılmaktadır.

3.2.1. Genetik algoritma kavramı

GA'ların arkasındaki konsept, genetik yapının ve popülasyonun birey kromozomlarının taşıdığı davranışın bir analojisine dayanmaktadır. Genetik algoritmanın unsurları Şekil 3.2'de gösterilmiştir. GA'lar aşağıdaki temel kavramlara dayanmaktadır:

- Popülasyon içindeki bireyler, eş ve kaynak bulmak için birbirleriyle etkileşimde bulunur.
- Başarılı olan yavruların daha fazla üretilmesi için birbirleriyle eşleşen bireylerdir.
- En uygun ebeveynlere ait genler, nesiller boyunca büyümeye devam eder; bu da ebeveynlerin bazen kendi aralarında sahip olmadıkları daha iyi özelliklere sahip yavrular yaratmalarına neden olur.
- Her bir ardışık nesil, ortamlarına daha uygun hale gelir.

3.2.2. Arama alanı

Popülasyonun tüm bireyleri, her bir bireyin mevcut arama uzayındaki problemin bir çözümü olarak hareket ettiği bir arama uzayında bulunur. Programlama dünyasında, bu bireylerin her biri, bileşen kromozomuna benzer bir vektör veya sonlu uzunlukta temsil edilir ve burada bileşenler aslında Genleri oluşturur. Kısacası, farklı genler (değişken bileşenler), bir kromozomu (bir birey) oluşturmak için bir araya gelir.

3.2.3. Genetik algoritmaların aşamaları

Genetik algoritmaların izlediği süreç genel olarak aşağıdaki adımlara ayrılabilir:

İlk nüfus

Genetik algoritmaların süreci başlangıçta bir popülasyon oluşturan bir dizi bireyle başlar. Burada, her birey mevcut soruna bir çözümü temsil eder. Genetik Algoritmalarda başlangıç popülasyonu oluşturmak için kullanılan yaygın yaklaşımlar şu şekilde sıralanabilir:

- Rastgele Başlatma
- Domain-Specific Initialization
- Latin Hypercube Sampling
- Önceden Var Olan Çözümler

Fitness fonksiyonu

Uygunluk işlevi, her bireye bir uygunluk puanı atayarak bir bireyin ne kadar uygun veya uygun olduğunu bulma işine sahiptir. Bir bireyin yaklaşan yeniden üretime katılma olasılığı, taşıdığı uygunluk puanına bağlıdır. Uygunluk fonksiyonunun aşağıdaki kavramlarından bahsetmek önemlidir:

- Uygunluk Fonksiyonunun Oluşturulması: Uygunluk fonksiyonları her zaman optimizasyon probleminin doğasına göre probleme özel tasarlanmalıdır.
- Değerlendirme Süreci: Uygunluk işlevi, popülasyonda bulunan her bir bireyi, optimizasyon hedefleriyle performansına göre değerlendirir.
- Uygunluk Ataması: Değerlendirme aşamasında değerlendirilen uygunluk değerleri popülasyondaki ilgili bireylere atanır.
- Uygunluk Manzarası: Uygunluk manzarası, uygunluk değerlerinin popülasyon içindeki karşılık gelen bireylerle ilişkisini temsil eder.
- Uyumluluk Paylaşımı ve Ölçeklendirme: Bazı problemlerde, uygunluk paylaşımı ve ölçekleme yöntemleri de popülasyondaki çeşitlilik ve keşfetme yeteneğinde iyileştirmeler yapmak için kullanılır.

Seçilim

Bir GA'nın seçim aşamasında amaç, hedeflere ulaşmaya yol açabilecek en uygun bireyleri seçmek ve genlerini sonraki nesillere aktarmaktır.

Seçim aşamasının daha iyi anlaşılması için bu aşama farklı bölümlere ayrılabilir.

- Seçim Yöntemleri: Araştırmacılar, GA'larda kullanılmak üzere birçok seçim yöntemi bulmuşlardır. Bu yöntemlerin her biri problemin parametrelerine, özelliklerine ve

doğasına göre birbirinden farklılık gösterir, bu nedenle her yöntemin kendine göre bazı avantaj ve dezavantajları vardır.

- Seçim Baskısı: Seçim baskısı, daha iyi bireylere seçim için bir ayrıcalık verilme derecesi anlamına gelir.
- Elitizm: Elitizm, bir nesildeki en iyi bireyleri koruma ve ardından onu bir sonraki nesile aktarma sürecidir.
- Ebeveyn Seçimi: Seçim olasılıkları belirlendikten sonra, ebeveynler bu olasılıklara göre seçilir. Değiştirmeli veya değiştirmesiz seçim yapılabilir.
- Çeşitliliğin Korunması: Popülasyondaki çeşitliliğin korunması için seçilim baskısı ile keşif arasında bir dengenin sağlanması çok önemlidir. Daha yüksek seçim baskısı, erken yakınsamaya ve çeşitlilik kaybına neden olabilir, bu da popülasyonun takılıp kaldığı yetersiz bir çözüme yol açar.

Çaprazlama

Çaprazlama işlemi, GA'larda temel bir genetik operatör olan rekombinasyon olarak da adlandırılabilir. Çaprazlama, seçim aşamasında seçilen ebeveynlerden gelen genetik materyalin çiftleşmesini ve rekombinasyonunu uyarır.

- Ebeveyn Seçimi: Çaprazlama işlemi, popülasyondan bireylerin uygunluk değerlerine bağlı olarak ebeveyn bireylerin seçilmesiyle başlar.
- Çaprazlama Operatörler: Pek çok çeşit çapraz operatör vardır ve hepsi ebeveynlerin genetik bilgilerinin kendi kombinasyonlarıyla gelir.
- Çaprazlama Oranı: Çaprazlama oranı, seçilen ebeveynler üzerinden çaprazlama gerçekleştirme olasılığını belirlemek için tanımlanması gereken bir parametredir.
- Mutasyon Oranı: Mutasyon oranı, bir bireyin genetik materyaline mutasyon oranının uygulanma olasılığına karar veren parametreyi temsil eder.
- Yavru Üretimi: Genetik materyal üzerinde belirli bir mutasyon oranında seçilen mutasyon işlemi uygulandıktan sonra oluşan yeni bireye yavru denir.

Mutasyon

Mutasyon süreci, genetik algoritmaların işleyişi açısından önemli bir rol oynar ve bireylerin genetik materyalinde rastgele değişiklikler veya pertürbasyonlar yapar. Bu, popülasyona çeşitlilik ekler ve çözüm uzayındaki yeni bölgelerin keşfinde yardımcı olur. Mutasyon, çözümün erken yakınsamasını önlemede önemli bir rol oynar ve optimal veya optimal

çözümlere ulaşma olasılıklarını artırır. Mutasyonun önemli kavramları aşağıda açıklanmıştır:

- Mutasyonun Amacı: Mutasyon, bireylere rastgele değişiklikler yaparak popülasyondaki genetik çeşitliliği sürdürmek için kullanılan bir mekanizmadır. Yalnızca çaprazlama ile ulaşılamayacak yeni ve çeşitli bölgelerde ileri düzeyde keşif yapma olanağı tanır. Mutasyon, popülasyonun alt-optimal bir çözüme yakınsamasını önler ve daha iyi optimal çözümlerin ortaya çıkmasını destekler.
- Mutasyon Operatörleri: Genetik algoritmalarında birçok mutasyon operatörü keşfedilmiş ve başarıyla uygulanmıştır. Bu operatörler, genetik materyallerde değişiklik yapma konusunda birbirinden farklıdır.
- Mutasyon Oranı: Mutasyon oranı, bir bireyin genetik materyaline uygulanan mutasyon oranının olasılığını belirleyen bir parametreyi temsil eder. Yüksek bir mutasyon oranı, rastgele değişikliklerin olasılığını artırır ve sonunda popülasyon içinde keşif ve daha fazla çeşitliliği teşvik eder. Diğer yandan düşük bir mutasyon oranı, mutasyon olasılığını azaltmaya ve bireylerin genetik materyalini korumaya yardımcı olur.
- Yavruların Oluşturulması: Belirli bir mutasyon oranındaki seçilen mutasyon işleminin uygulanmasının ardından, yeni bireye "yavru" denir. Yeni yavru, çeşitli genetik bilgilerle popülasyona eklenir ve gelecek nesillerde yer alır.

3.3. Çok Amaçlı Optimizasyon

Çoklu amaçlı optimizasyon (ÇAO) alanında amaç, iki veya daha fazla çakışan amacı eş zamanlı olarak optimize etmektir. Tek amaçlı optimizasyona karşı, ÇAO genellikle çatışan çoklu amaç arasında bir uzlaşmayı temsil eden bir çözüm kümesi bulmayı amaçlar. Bu alan, mühendislik, finans, lojistik ve yapay zeka dahil olmak üzere çeşitli alanlarda uygulama bulmuştur. Diyelim ki maksimize edilmesi gereken n adet farklı amaç fonksiyonu bulunsun. Bu işlem matematiksel olarak Denklem 3.1'teki gibi ifade edilebilir.

$$\max\big(f_1(x)\big), \max\big(f_2(c)\big), \dots, \max\big(f_n(x)\big) \tag{3.1}$$

Burada x girdi çözümünü temsil eder ve $(f_1(x), f_2(x), \dots, f_n(x))$ optimize edilmesi gereken farklı amaç fonksiyonlarıdır. Çözüm S_1'in amaç fonksiyonları tarafından elde edilen değerler S_2 çözümü tarafından elde edilen değerlerden üstünse, bir S_1 çözümü S_2 gibi başka bir çözüme göre baskın bir konumda kabul edilir.

Bu sorunları ele almak için bu çalışmada çözümün optimize edilmesinde üç farklı amaç fonksiyonunu kullanan çok amaçlı bir optimizasyon yaklaşımı önerilmiştir. Kullanılan amaç fonksiyonları aşağıda açıklanmıştır.

3.3.1. Temporal smoothness

Çok amaçlı optimizasyonda nihai sonuç olarak kabul edilebilecek tek bir optimal çözüm yoktur. Bunun yerine, diğerlerinden üstün olduğu düşünülen bir dizi çözüm elde edilir. Bunu başarmak için, Denklem 3.2 (Chi et al., 2009) kullanılarak problem istenen bir sonuca dönüştürülür:

$$cost = \alpha SC + (1 - a)\, TC \tag{3.2}$$

Bu yaklaşımda, elde edilen topluluk yapılarının değerlendirmelerini ölçmek için Snapshot Cost (SC) kullanılır. Bu yazıda önerilen yöntem, SC'yi hesaplamak için Modülerliği kullanır. Daha yüksek Modülerliğe ve daha düşük SC'ye sahip bir çözümün daha iyi bir topluluk yapısına sahip olduğu kabul edilir. Temporal Cost (TC), topluluk yapıları arasında var olan benzerlikleri ölçer. Önerilen yöntem, TC'yi hesaplamak için Normalleştirilmiş Karşılıklı Bilgiyi (NMI) kullanır.

3.3.2. Modularity

Q-fonksiyonu olarak da bilinen modülerlik, ağ topluluğu algılama araştırmalarında yaygın olarak kullanılan bir değerlendirme kriteridir. Topluluk yapısının gücünün nicel bir değerlendirmesini sağlayarak, bir grup içindeki gerçek ve tahmin edilen uç sayısı arasındaki tutarsızlığı ölçer. Matematiksel olarak Modülerlik Denklem 3.3'deki gibi yazılabilir.

$$Q(CR^t) = \sum_{i=1}^{CR^t} \left[\frac{l_i}{E} - \left(\frac{d_i}{2E} \right)^2 \right] \tag{3.3}$$

Burada $|E|$, N^t'de bulunan kenarları temsil eder, i. topluluktaki kenarların sayısı l_i olarak anılır ve son olarak d_i derecesi i. topluluğun derecesini temsil eder. Modülerlik metriği, [-1/2, 1] aralığında yer alır ve burada daha yüksek Modülerlik puanı, daha iyi bir topluluğun oluştuğunu gösterir.

3.3.3. Normalized mutual information

Normalleştirilmiş Karşılıklı Bilgi (NMI), iki topluluk bölümü arasındaki benzerliği ölçen köklü bir ölçümdür. NMI, farklı algoritmalardan elde edilen topluluk yapılarının kararlılığının bir ölçüsünü de sağlayabilir. $CR^{t-1} = C_1^{t-1}, C_2^{t-1}, \ldots, C_l^{t-1}$ ve $CR^t =$

$C_1^t, C_2^t, \ldots, C_k^t$ diyelim t zamanında oluşan iki topluluktur, daha sonra bu iki topluluk arasındaki benzerliği elde etmek için kullanılan $NMI(CR^t,\ CR^{t-1})$ Denklem 3.4'te matematiksel olarak tanımlanır.

$$NMI(CR^t, CR^{t-1}) = \frac{-2\sum_{i=1}^{m}\sum_{j=1}^{n} C_{ij}\log\left(\frac{C_{ij}N}{C_i C_j}\right)}{\sum_{i=1}^{m} C_i \log\left(\frac{C_i}{N}\right) + \sum_{j=1}^{n} C_j \log\left(\frac{C_j}{N}\right)} \quad (3.4)$$

3.3.4. Normalized cut

Normalized Cut (NC) [8], bir bölümleme süreciyle elde edilen toplulukların kalitesini değerlendirmek için kullanılan bir ölçüdür. Matematiksel terimlerle, $C^t = C_1, C_2, \ldots, C_k$'yi t anında keşfedilen k topluluk kümesi olarak kabul edelim. NC işlevi, farklı toplulukları birbirine bağlayan toplam kenar ağırlığının bu topluluklar içindeki iç kenar ağırlıklarının toplamına oranını hesaplayarak topluluklar arasındaki benzerliği belirlemek için kullanılır. Belirli bir t zamanındaki topluluk yapısı kalitesi, Denklem 3.5'te verilen normalleştirilmiş kesim kullanılarak belirlenebilir.

$$NC(C^t) = \frac{1}{k}\sum_{i=1}^{k}\frac{cut\ (C_i, \bar{C}_i)}{vol\ (C_i)} \quad (3.5)$$

3.4. Non-dominated Sorting Genetic Algorithm-II

Baskın Olmayan Sıralama Genetik Algoritması-II (NSGA-II) [5] toplulukları tespit etmede oldukça verimli olan çok amaçlı bir optimizasyon algoritmasıdır. Bu algoritma, her çözüm farklı bir topluluk bölümlemesini temsil eden bir dizi Pareto çözümünü bulmak için genetik bir yaklaşım kullanır. NSGA-II algoritmasında çözülen üç ana problem aşağıdaki gibidir.

3.4.1. Yüksek hesaplama karmaşıklığı

Yüksek Hesaplama Karmaşıklığı, NSGA algoritmasında karşılaşılan önemli bir zorluktur. NSGA algoritması, $O(MN^3)$ hesaplama karmaşıklığına sahiptir; burada M, optimize edilmesi gereken hedeflerin miktarını temsil eder ve N, popülasyon büyüklüğünü ifade eder. NSGA-II algoritmasının hesaplama karmaşıklığı, $O(MN^2)$'ye düşürüldü ve bu da onu selefi NSGA'ya kıyasla daha verimli bir algoritma haline getirdi.

3.4.2. Non-elitist yaklaşım

NSGA'da ve diğer birçok GA'da elitist olmayan bir yaklaşım izlenmiştir. Elitizm, mevcut popülasyon ile önceki yinelemede bulunan en iyi baskın olmayan çözüm arasında bir karşılaştırma yaparak yapılır.

3.4.3. Paylaşım parametresi - σ_share

Önceki algoritmalar, her yinelemede farklı bir popülasyon oluşturmak için σ_share olarak belirtilen bir paylaşım parametresi kullanıyordu. Ancak, bu parametrenin ayarı kullanıcının bilgisine ve uzmanlığına bağlıydı ve bu da hatalara yol açabiliyordu. Ek olarak, paylaşım işlevi, her bir çözümün daha önce elde edilen çözümlerle karşılaştırılmasını gerektirdiğinden, $O(MN^2)$ gibi yüksek bir hesaplama maliyetine sahipti. Bu sorunları ele almak için NSGA-II, paylaşım işlevini, paylaşım işleviyle ilgili sorunların etkin bir şekilde üstesinden gelen kalabalık bir işlev yaklaşımıyla değiştirmiştir.

3.4.4. Pareto-rank dayalı değerlendirme

Pareto-rank tabanlı değerlendirme, bir problemin birden fazla çözümünü aynı anda değerlendirmek ve karşılaştırmak için kullanılan çok amaçlı bir optimizasyon yaklaşımıdır. Pareto sıralama yöntemi ilk olarak 1989'da Goldberg (Goldberg, 1989) tarafından önerilmiştir. Pareto sıralaması kavramının kökleri Pareto etkinliğine dayanmaktadır; bu, bir çözümün, bir hedefi geliştirmesine neden olmadan geliştirme yeteneğine sahip başka bir çözüm yoksa etkili kabul edildiğini belirtir. Pareto sıralaması, NSGA-II algoritmasının çok önemli bir bileşenidir. Belirli bir popülasyondaki baskın olmayan bireyler grubu içinde üreme için eşit fırsatlar sağlar. Pareto tabanlı yöntem, çözümleri birden çok aşamaya böler ve popülasyonda en yüksek uygunluk değerlerine sahip bireyler Sıra 1'e atanır. Bu bireyler, diğerleri tarafından domine edilmez ve popülasyondan çıkarılır. Süreç daha sonra, ikinci yinelemede bulunanlar ikinci en iyi uygunluk değerleri ile 2. Sıraya atanarak, bir sonraki domine edilmemiş bireyleri bulmak için tekrarlanır. Bu prosedür, popülasyondaki her birey bir rütbe alana kadar devam eder.

Pareto çözümleri belirlendikten sonra sıra en uygun çözümün seçilmesine gelir. Bunu başarmak için önerilen yöntemde 2. dereceden Topluluk Puanı kullanılmıştır. Topluluk Puanı (CS/Community Score), Denklem 3.6 matematiksel olarak yazılabilen, elde edilen topluluk yapılarının kalitesini belirler:

$$CS(CR^t) = \sum_{i=1}^{k} \left[\frac{\sum_{m \in C_i^t} (\mu_m)^2}{|C_i^t|} \; x \; \sum_{m,n \epsilon C_i^t} A_{mn} \right] \tag{3.6}$$

3.5. CD-INSGA Metodolojisi

Bu tez çalışmasında önerilen metodoloji, Transfer tabanlı Çok amaçlı NSGA-II algoritması olarak adlandırılmaktadır. Bu metodoloji, NSGA-II algoritmasının seçkin özelliklerini,

TMOGA algoritmasından (Zou et al., 2021) esinlenen ek bir transfer tabanlı teknikle kullanır. CD-INSGA'de izlenen genel süreç aşağıda açıklandığı gibidir.

3.5.1. Özellik çıkarımı

Özellik çıkarımı, bir ağdan önemli özelliklerin veya karakteristiklerin tanımlanmasını ve çıkarılmasını içerdiğinden, ağ çalışmalarında çok önemli bir adımdır. Ağ çalışmalarında özellik çıkarımı için topolojik, spektral ve istatistiksel yöntemler dahil olmak üzere çeşitli yöntemler vardır. Topluluk algılama probleminde topluluklar, klikler veya özellikler olarak bilinen daha küçük gruplara ayrılır.

Özellik çıkarma işlemi, arama ağacı genişletmesi için başlangıç noktası olarak bir ilk düğümün seçilmesiyle başlar. Arama ağacı, komşu düğümün indeksi mevcut düğümden büyük olduğu, arama ağacının genel derinliği önceden belirlenmiş maksimum derinlik (Md/Maximum depth) eşiğini aşmadığı ve keşfedilen kliklerin bir Topluluğu olduğu sürece genişlemeye devam eder. Set edilen değer Dahili Yoğunluk (CID/Community Internal Density) eşiğden yüksek ise genişlemeye devam eder. Bu sürecin sonunda, tüm küçük klikler veya özellikler çıkarılmış olacaktır.

3.5.2. Özellik taşıma

Özellik geçişi (taşıma), ağ çalışmalarında bir ağdan çıkarılan özelliklerin başka bir ağa aktarılmasını içeren bir süreçtir. Ağ analizi alanında, her ağdan ayrı ayrı özelliklerin çıkarılmasıyla ilişkili hesaplama süresinin ve karmaşıklığının azaltılmasına yardımcı olan kritik bir tekniktir. Özellik Taşıma işlemi, özelliklerin çıkarıldığı bir başlangıç ağının seçilmesiyle başlar. Ayıklanan özellikler daha sonra bir sonraki ağa aktarılır ve burada daha fazla özellik çıkarımı için bir başlangıç noktası olarak kullanılır. Yeni popülasyondaki çeşitliliği korumak için, Transfer Olasılığı (T_p) adlı bir değişken tanıtılmıştır (Zou et al., 2021). CD-INSGA işleminde (T_p) değeri 0,5 olarak ayarlanmıştır. Özellik taşıma işlemi sırasında, her özelliğin topluluk yapısı, (T_p) değerine göre her popülasyonda bulunan ilk çözüme atanır.

3.5.3. Başlatma

CD-INSGA'de, başlangıç popülasyonunu oluşturmak için Etiket Yayma Algoritması (LPA/Label Propagation Algorithm) (Raghavan et al., 2007) kullanılır ve minimum hesaplama karmaşıklığı ile çok hızlı bir şekilde verimli ve birinci sınıf çözümler sunar.

3.5.4. Seçim

Seçim, bir popülasyondan belirli bireyleri, yeni popülasyonun oluşturulması için çaprazlama ve mutasyon süreçlerine geçirmek üzere seçme aşamasıdır. NSGA-II algoritmasında (Deb et al., 2002) seçim işleminde turnuva yöntemi kullanılmaktadır. Tournament selection'da (TOS) (Hingee & Hutter, 2008), k sayısı bireyleri rastgele seçilir. Ters Pareto sıralama yöntemi, önerilen yöntemde yeniden üretim süreçleri için ağırlıklar atamak ve rastgele bir seçim yapmak için bireylerin Pareto sıralamalarını dikkate alan geleneksel turnuva seçim yönteminin bir modifikasyonudur.

3.5.5. Çaprazlama

Crossover bir genetik algoritmada ana arama operatörü olarak kabul edilir. Ebeveyn popülasyonlarındaki bireyler, çocuk popülasyonları oluşturan yeni bir bireyler dizisi üretmek için belirli bir şekilde birleştirilir. NSGA-II'de (Deb et al., 2002), bu amaçla simüle edilmiş bir ikili geçiş SBX (Simulated Binary Crossover) (Deb & Agrawal, 1995) operatörü kullanılmaktadır. SBX çaprazlamanın dezavantajı, eğer ebeveyn popülasyonları hemen hemen veya tamamen aynıysa çocuk popülasyonunun da aynı olması, farklı bir çocuk popülasyonu elde etme şansını azaltmasıdır. Bunun üstesinden gelmek için, ilk kez NSGA-II'de uygulanan önerilen CD-INSGA yönteminde Cycle Crossover (CX) stratejisi (Puljić & Manger, 2013) kullanılmıştır.

3.5.6. Mutasyon

Mutasyon süreci, genetik algoritmanın amaçlanan amacından saparak rastgele bir arama algoritmasına dönüşmesini önlemek için düşük tutulması gereken Mutation Probability (Mp) (Mp) tarafından yönlendirilir. Mutasyon gerçekleştirmek için yaygın olarak kullanılan bir yönteme polynomial mutation denir. Bu yöntem, aynı zamanda yeni ve potansiyel olarak daha iyi çözümlerin gelişimini teşvik ederken, popülasyonun genetik çeşitliliğini sürdürmeye yardımcı olur. Bu nedenle önerilen algoritmada polynomial mutation uygulanmıştır.

3.5.7. Crowding distance elimination (CDE)

Crowding Distance fonksiyon, bir popülasyon içindeki çeşitliliği korumak ve baskın olmayan çözümlerde bulunan yoğunluğu belirlemek için genetik algoritmalarda yaygın olarak kullanılan bir araçtır. Bu işlev, belirli bir çözümü çevreleyen baskın olmayan çözümlerin göreli yoğunluğunu yaklaşık olarak tahmin eder.

Kalabalık mesafe fonksiyonuna yönelik geleneksel yaklaşımın, yoğun çözümlerin komşu çözümler üzerindeki etkisini dikkate almaması nedeniyle bir kusuru vardır (Liu & Chen, 2019; Tahmasebi et al., 2019). Bu sorunu ele almak için, CD-INSGA algoritmasında Crowding Distance Eliminatio (CDE) (Liu & Chen, 2019) adı verilen geliştirilmiş bir yöntem uygulanmıştır. Bu yöntem, aynı Pareto ön seviyesindeki çözümler için önce dışlama mesafesinin hesaplanmasını ve ardından dışlama mesafesinin en düşük değerine sahip çözümün silinmesini içerir. Önerilen algoritmada CDE uygulanmıştır.

4. SONUÇLAR VE TARTIŞMA

Bu bölüm, önerilen CD-INSGA algoritmasının pratik deneylerini ve sonuçlarını sunar. Algoritma, performansını değerlendirmek için TMOGA (Zou et al., 2021), FacetNet (Lin et al., 2008), DYNMOGA (Folino & Pizzuti, 2013) ve DYNMODPSO (Yin et al., 2021) algoritma ile karşılaştırılır. Normalleştirilmiş Karşılıklı Bilgi (NMI/Normalized Mutual Information), sonuçları analiz etmek ve topluluk bölümlerinin benzerliğini belirlemek için bir kriter olarak belirlenir. NMI değeri 0 ile 1 arasında değişir ve 1'e yakın bir değer, karşılaştırılan topluluklar arasında daha yüksek benzerlik gösterir.

4.1. Parametre Ayarı

CD-INSGA algoritması, yürütülmeden önce yedi parametrenin ayarlanmasını gerektirir. Bu parametreler popülasyon büyüklüğü, nesil sayısı, Topluluk İç Yoğunluğu (CID/Community Internal Density) eşiği, Geçiş olasılığı (Cp/Crossover probability), Mutasyon olasılığı (Mp/Mutation probability), Aktarım olasılığı (Tp/Transfer probability) ve arama ağacının maksimum derinliğidir (Md/Maximum depth). Diğer algoritmalarla karşılaştırıldığında adaleti sağlamak için popülasyon büyüklüğü ve nesil sayısı sırasıyla 200 ve 100 olarak belirlenmiştir. Deneylerde CID eşiği 0,8 olarak ayarlanmıştır. Md parametresi 5 olarak ayarlanmıştır. Tp 0.5 olarak ayarlanmıştır. Genetik algoritmada yeni popülasyon üretimi için Cp ve Mp değerleri sırasıyla 0.8 ve 0.2 olarak sabitlenmiştir.

4.2. Veri Kümeleri

Testleri gerçekleştirmek için sentetik ve büyük ölçekli karmaşık veri kümeleri kullanılmaktadır. Veri kümelerinin açıklaması aşağıda tartışılmaktadır:

4.2.1. Sentetik veri kümesi

Çalışma, Newman ve Girvan (Girvan & Newman, 2002) tarafından öne sürüldüğü gibi, SYN-FIX ve SYN-VAR olmak üzere iki sentetik veri seti kullanmaktadır. Her topluluğa bağlı kenarların sayısını düzenlemek için "Z" adlı bir parametre kullanılır. Aşağıdaki bölümlerde, her iki veri kümesine ilişkin kısa bir genel bakış sağlanacaktır.

- SYN-FIX: SYN-FIX veri kümesinin tanımlayıcı özelliklerinden biri, her bir topluluğu birbirine bağlayan sabit bir kenar sayısına sahip olmasıdır. SYN-FIX veri kümesindeki her bir topluluğu birbirine bağlayan sabit sayıda uç, "Z" adlı bir parametre tarafından kontrol edilir. Z parametresi, topluluk dışında var olan düğümlerle bağlanacak kenarların sayısını yönetir. Yapılan deneylerde Z değeri 3 ve 6 olarak ayarlanmıştır.
- SYN-VAR: Zaman içinde topluluk sayısını değiştiren değişken bir veri kümesidir. Bu, her topluluğu birbirine bağlayan kenarların sayısının topluluğun boyutuyla orantılı olduğu anlamına gelir. SYN-FIX ile aynı şekilde, bu veri setinde ayrıca ayarlanacak Z parametresi vardır. Z'nin değeri Z= 3 ve 6 olarak ayarlanmıştır.

4.2.2. Büyük ölçekli veri kümesi

Mevcut araştırmanın doğruluğu, değerlendirme kriteri olarak Normalleştirilmiş Karşılıklı Bilgi (NMI/Normalized Mutual Information) kullanılarak büyük ölçekli karmaşık veri kümeleri üzerinde araştırılır. Düğümler, her zaman döngüsünde farklı topluluklara rastgele yeniden atandı ve düğümlerin %20'si etkilendi. Dört dinamik ağ, referans (Greene et al., 2010)'te açıklanan araç kullanılarak oluşturuldu. Bu dinamik ağlar, dört farklı türde evrimsel olay geçirdi.

- Doğum ve Ölüm Modeli: "Doğum" terimi ağa yeni düğümler ekleme eylemiyle ilgiliyken, "ölüm" mevcut düğümlerin kaldırılması anlamına gelir. Doğum ve Ölüm Modeli içinde, yeni topluluklar oluşturmak için mevcut bir topluluktan %10 düğümlerden oluşan rastgele bir örneklem seçilirken, topluluğun %10'u çıkarılır.
- Genişleme ve Daralma Modeli: Genişleme ve Daralma veri kümesi modeli, belirli toplulukların zaman içinde boyutunun arttığı (genişlediği) ve diğerlerinin boyutunun azaldığı (daraldığı) bir topluluk evrim senaryosuna atıfta bulunur. Bu modelde, yeni topluluklar oluşturmak için mevcut topluluklardan düğümlerin %10'u seçilir ve mevcut topluluklardan düğümlerin %10'u çıkarılarak yeni bir topluluk seti oluşturulur.

- Aralıklı Topluluklar Modeli: Aralıklı Topluluklar veri kümesi modelinde, dinamik ağın her aralığında, orjinal toplulukların %10'u geçici olarak kaldırılır. Bu topluluklar daha sonra bir sonraki adımda tekrar eklenir ve topluluk yapısında aralıklı davranış görünümü verilir.
- Birleştirme ve Bölme Modeli: Birleştirme ve bölme işlemi her adımda devam ederek dinamik ve gelişen topluluklarla sonuçlanır. Her anlık görüntüde, toplam toplulukların %10'u orjinal topluluklarından ayrılır ve yeni topluluklar oluşturmak için diğer toplulukların %10'u ile birleştirilir.

4.3. Deneysel Sonuçlar

Bu bölüm, CD-INSGA, TMOGA, FacetNet, DYNMOGA ve DYNMODPSO adlı beş farklı algoritmanın kapsamlı bir değerlendirmesini içermektedir. Deneyler, 4.1. bölümde belirtilen parametre ayarlarına özenle dikkat edilerek gerçekleştirilmiştir. Bu deneylerde kullanılan veri setleri, 4.2. bölümünde detaylı bir şekilde açıklanmış olup, algoritmaların değerlendirildiği bağlamla ilgili kapsamlı bir anlayış sunmaktadır. Bu deneyler aracılığıyla, algoritmaların çeşitli veri setleri ve ayarlar üzerindeki bireysel ve kolektif performanslarını aydınlatarak karşılaştırmalı bir analiz sunma hedeflenmektedir.

4.3.1. Sentetik veri setleri kullanılarak elde edilen sonuçlar:

Deney sonuçları tarafından gösterildiği üzere, CD-INSGA algoritmasının FacetNet ve DYNMODPSO algoritmalarını tüm koşullar altında geride bıraktığı açıktır. TMOGA, genellikle başlangıç iterasyonlarında CD-INSGA'yi geride bırakırken, ardışık iterasyonlarda benzer sonuçlar elde etmektedir. DYNMOGA, ağın daha az karmaşık olduğu durumlarda, yani toplulukları birbirine bağlayan kenar sayısı daha az olduğunda, daha iyi performans sergilemektedir. FacetNet'in etkileyici bir zaman karmaşıklığı olmasına rağmen, performansı önemli ölçüde etkilenmiştir. Önerilen algoritmanın, TMOGA, DYNMODPSO ve DYNMOGA'ya kıyasla daha düşük bir zaman karmaşıklığı vardır, bu da onu hem zaman hem de hesaplama karmaşıklığı açısından karmaşık dinamik sentetik ağlar için en iyi seçenek haline getirir.

4.3.2. Büyük ölçekli veri setleri kullanılarak elde edilen sonuçlar:

Büyük ölçekli veri setleri üzerinde gerçekleştirilen deneylerin sonuçlarından, önerilen CD-INSGA algoritmasının her bir zaman adımında tüm veri setlerinde, her bir zaman adımında toplulukların değişmesine rağmen, DYNMODPSO ve DYNMOGA'dan daha iyi bir

performans sergilediği gözlemlenmektedir. DYNMODPSO, düzeltme tekniği nedeniyle başlangıçta ilk iki zaman adımında iyi bir performans gösterir, ancak sonraki zaman adımlarında performansı düşer. Sentetik veri setlerinde iyi bir performans göstermiş olan DYNMOGA, büyük ölçekli veri setlerinde zayıf bir performans sergiler. FacetNet, özellikle her veri setinin ilk zaman adımında optimal sonuçlar elde eder, ancak yüksek zaman karmaşıklığına sahip olması, neredeyse CD-INSGA'nin iki katı, büyük ölçekli veri setleri için uygun olmaması önemli bir dezavantajdır. Öte yandan, TMOGA, CD-INSGA ile karşılaştırılabilir sonuçlar elde eder ancak çok daha yüksek bir zaman karmaşıklığına sahiptir. Performans ve zaman karmaşıklığını bir araya alındığında, CD-INSGA en iyi performansı sergiler.

5. SONUÇ VE ÖNERİLER

Ağlar içinde topluluk tespiti alanındaki keşif, özellikle gerçek dünya senaryolarında yaygın olarak bulunan dinamik ağların karmaşıklıklarıyla başa çıkma çabası olarak durmaktadır. Bu tür ağların açık bir şekilde önemli olmasına rağmen, dinamik ağlar, özellikle içsel karmaşıklıkları nedeniyle geçmiş araştırmalarda genellikle göz ardı edilmiştir. Bu çalışma, bu dengesizliği gidermek için dikkatini dinamik yöntemlere yönlendiren bir NSGA-II algoritması üzerine odaklanarak gerçekleştirilen bir çaba olarak ortaya çıkmaktadır. NSGA-II, çoklu çatışan amaçları olan optimizasyon problemlerini çözmek için güçlü bir yöntem olup, motivasyonu bu tür problemlerin ortaya koyduğu zorluklara çözüm getirmektir ve kullanım alanı, karar vericilerin rekabet eden hedefler arasında dengeleme yapmaları gereken çeşitli alanları içermektedir.

Bu çalışma sürecinde NSGA-II'de iyileştirilmesi gereken alanlar, CD-INSGA yönteminin geliştirilmesine yol açtı. Bu yeni yaklaşım, sadece Cycle Crossover (CX) tekniği ile crossover yöntemini iyileştirmekle kalmayıp aynı zamanda Crowding Distance Elimination (CDE) yaklaşımını uygulayarak crowding distance yöntemini de geliştirmektedir. İstenen non-dominant çözümlere ulaşılana kadar en az crowding distance'a sahip çözümlerin iteratif olarak elemesi, önemli bir ilerleme olarak kabul edilmektedir.

Ayrıca, Modülerlik, NMI ve Normalized Cut olmak üzere üç amaç fonksiyonunun optimize edilmesi, edinilen toplulukların güçlü bir değerlendirme çerçevesini oluşturdu ve Performans ölçütü olarak Normalized Mutual Information kullanıldı. Evrimsel algoritma

içinde önceki iterasyonlarda keşfedilen önemli bilgileri kullanma stratejisini içeren bir özellik transfer öğrenme stratejisi, stratejik bir gelişme olarak ortaya çıktı.

Önerilen CD-INSGA yöntemi, sadece gelişmiş topluluk yapıları ve optimal çözümler üretmede değil, aynı zamanda çözüm çeşitliliğini koruma konusunda da dikkate değer bir kapasite sergilemektedir. Sentetik dinamik ağlar ve büyük ölçekli dinamik ağ veri setleri üzerinde yapılan titiz deneyler, önerilen algoritmanın üstün performansını açıkça göstermiş olup, TMOGA, FaceNet, DYNMODPSO ve DYNMOGA gibi köklü yöntemleri aşmıştır. Amaç fonksiyonlarının sayısının arttırılması ve zaman karmaşıklığındaki belirgin gelişmeler, önerilen tekniklerin karmaşık büyük ölçekli dinamik ağlardaki toplulukları tanımlama konusundaki uygunluğunu daha da doğrulamaktadır.

Gelecek yönergeler:

Önümüzdeki süreçte, CD-INSGA yönteminin etkinliğini genişletmek için umut vadeden alanlar araştırılacaktır. Özellikle gelecek çalışmalar önerilen yöntemin üzerinde durarak örtüşen topluluklar için uyarlanmasına öncelik vermelidir. Bu geliştirme, algoritmanın topluluk örtüşme fenomeninin yaygın olduğu gerçek ağlara uygulanabilirliğini önemli ölçüde genişletebilir. Ayrıca, özellik transfer öğrenme stratejisinin daha fazla iyileştirmelerini ve bu stratejinin çeşitli alanlara uygunluğunu incelemek, algoritmanın etkililiğini ilerletmek açısından önemli olacaktır. Bu yöndeki sürekli keşif ve iyileştirme, topluluk tespiti algoritmalarının devam eden evrimine ve karmaşık, dinamik ağ ortamlarında pratik uygulamalarına katkıda bulunacaktır.

Bilim Kodu: 92432

Anahtar Kelimeler: Önemli Alt Gruplar, Dinamik Ağlar, NSGA-II, Çok Amaçlı Optimizasyon, Cycle Crossover (CX), Crowding Distance Elimination (CDE)

Sayfa Adedi: 81

Danışman: Dr.Öğr.Üyesi Yılmaz ATAY

AN EVOLUTIONARY MULTI-OBJECTIVE APPROACH TO UNCOVER COMMUNITIES IN DYNAMIC NETWORKS

(M.Sc. Thesis)

Kainat Akhtar TARRAR

GAZİ UNIVERSITY

INSTITUTE OF INFORMATICS

February 2024

ABSTRACT

The detection of subgroups in networks is crucial for understanding network properties and behaviors. Several methods have been created to identify sub-groups within graphs, but there is limited research on dynamic networks. Recognizing the significance of dynamic structures, the present study proposes an advanced Non-Dominated Sorting Genetic Algorithm, named CD-INSGA, for community detection in social networks. In CD-INSGA, modifications have been carried out on methods of crossover and crowding distance in NSGA-II, with an integration of the transfer-based learning approach. The key contributions of this study are (1) The enhancement of the crossover operator by using the Cycle Crossover (CX) strategy to eliminate the local optimization problem in the solution; (2) The improvement of the crowding distance method by the introduction of Crowding Distance Elimination (CDE) mechanism in NSGA-II which helps to preserve the diversity of NSGA-II algorithm by taking out non-dominated solutions with increased efficiency; (3) Usage of three essential objective functions, naming Modularity, Normalized Mutual Information (NMI) and Normalized Cut (NC) resulting in accurate and qualified measurement of the discovered subgroups; (4) Prominent reduction of time complexity by using transfer based learning to preserve the valuable information found in previous generations and transfer it to the next generations. Experimental trials were conducted on synthetic dynamic networks with established factual information and real-world networks of substantial size. The results indicate that CD-INSGA surpasses its competitors concerning efficiency as well as effectiveness.

Science Code: 92432

Key Words: Significant Subgroups, Dynamic Networks, NSGA-II, Multi-Objective Optimization, Cycle Crossover (CX), Crowding Distance Elimination (CDE).

Page Number: 81

Supervisor: Asst.Prof.Dr. Yılmaz ATAY

PREFACE AND ACKNOWLEDGEMENTS

In embarking on this academic journey, I have encountered challenges, celebrated triumphs, and, most importantly, have been surrounded by unwavering support. As I present this thesis, I extend my deepest gratitude to those who have played instrumental roles in this fulfilling endeavor.

First of all, I take this opportunity to thank my supervisor, Asst.Prof.Dr. Yılmaz ATAY for his unwavering guidance on every aspect of this thesis. His deep knowledge and enthusiasm for exploring and diving deep in the world of social networks and science inspired me to keep up with my research work. His guidance and mentorship have helped me a lot to be motivated. I cannot thank him enough for the efforts and time that he has put in this project.

Foremost, I want to express my heartfelt thanks to my husband, Gökhan Sarıcıoğlu, whose enduring encouragement, patience, and unwavering belief in my abilities have been my pillars of strength. His sacrifices, understanding, and willingness to share both the joys and burdens of this academic pursuit have been my constant motivation. In times of self-doubt, he has been my source of inspiration, reminding me that resilience and perseverance pave the path to success.

This thesis is not just the culmination of my academic efforts, but a shared achievement of a partnership built on love, trust, and mutual support. Gökhan Sarıcıoğlu's contribution to my academic journey goes beyond words, and I am profoundly grateful for his unwavering commitment.

I would also like to express my gratitude to my family and friends who have always supported me. Their guidance, encouragement, and camaraderie have enriched my academic experience and made this journey more memorable. Special thanks to my friends Lionel Alangeh Ngobesing & Cylas Kiganda for always helping and guiding me.

CONTENTS

Page

ÖZET iii

ABSTRACT xxvii

PREFACE AND ACKNOWLEDGEMENTS xxviii

LIST OF TABLES xxxi

LIST OF FIGURES xxxii

SYMBOLS AND ABBREVIATIONS xxxiv

1. INTRODUCTION 1

2. LITERATURE REVIEW 11

3. MATERIALS AND METHODS 23

3.1. Evolutionary Algorithms 23

3.1.1. Benefits of evolutionary algorithms 24

3.1.2. Applications of evolutionary algorithms 25

3.1.3. Types of evolutionary algorithms 27

3.2. Genetic Algorithms (GAs) 28

3.2.1. Concept of genetic algorithms 28

3.2.2. The search space/area 29

3.2.3. The phases of genetic algorithms 29

3.3. Multi-Objective Optimization 38

3.3.1. Temporal smoothness 40

3.3.2. Modularity 40

3.3.3. Normalized mutual information 41

3.3.4. Normalized cut 42

3.4. Non-dominated Sorting Genetic Algorithm-II 42

3.4.1. High computational complexity 43

3.4.2. Non-elitist approach 43

3.4.3. The sharing parameter – $\sigma share$ 43
3.4.4. Pareto-rank based evaluation 44
3.5. CD-INSGA Methodology 46
3.5.1. Feature extraction 46
3.5.2. Feature migration 48
3.5.3. Initialization 49
3.5.4. Selection 49
3.5.5. Crossover 50
3.5.6. Mutation 53
3.5.7. Crowding distance elimination (CDE) 53
4. RESULTS AND DISCUSSION 57
4.1. Parameter Setting 57
4.2. Datasets 58
4.2.1. Synthetic dataset 58
4.2.2. Large scale dataset 59
4.3. Experimental Results 61
4.3.1. Results using synthetic dataset 61
4.3.2. Results using large-scale dataset 66
5. CONCLUSIONS AND SUGGESTIONS 75
REFERENCES 77
CIRRICULUM VITAE 81

LIST OF TABLES

Table **Page**

Table 3.1. Cycle Crossover Algorithm 51

Table 3.2. Crowding Distance Elimination Algorithm 55

Table 4.1. Results with SYN-FIX Z = 3 dataset 62

Table 4.2. Results with SYN-FIX Z = 6 dataset 63

Table 4.3. Results with SYN-VAR Z = 3 dataset 64

Table 4.4. Results with SYN-VAR Z = 6 dataset 65

Table 4.5. Results with Birth and Death Dataset 66

Table 4.6. Results with Expansion and Contraction Dataset 67

Table 4.7. Results with Intermittent Communities Dataset 67

Table 4.8. Results with Merging and Splitting Dataset 68

LIST OF FIGURES

Figure **Page**

Figure 1.1. Graphical representation of nodes 1

Figure 1.2. Edges between the nodes 2

Figure 3.1. General schema of an Evolutionary Algorithm (EA) 23

Figure 3.2. The structure of Genetic algorithms 29

Figure 3.3. Roulette wheel selection 32

Figure 3.4. Common crossover methods 35

Figure 3.5. Common mutation methods 37

Figure 3.6. Systematic diagram of NSGA-II sorting mechanism 45

Figure 3.7. Two cliques connected by an internal link 47

Figure 3.8. Schematic diagram of cycle formation 52

Figure 3.9. Schematic representation of the first stage of crossover cycle. 52

Figure 3.10. Schematic representation of offspring after crossover cycle completion. 53

Figure 3.11. Graphical representation of Crowding Distance mechanism. 54

Figure 3.12. Flowchart of CD-INSGA 56

Figure 4.1 Graphical representation of results using SYN-FIX Z=3 dataset 62

Figure 4.2. Graphical representation of results using SYN-FIX Z=6 dataset 63

Figure 4.3. Graphical representation of results using SYN-VAR Z=3 dataset 64

Figure 4.4. Graphical representation of results using SYN-VAR Z=6 dataset 65

Figure 4.5. Obtained nondominated solutions with CD-INSGA on Birth and Death Dataset 70

Figure 4.6. Obtained nondominated solutions with CD-INSGA on Expansion and Contraction Dataset 70

Figure 4.7. Obtained nondominated solutions with CD-INSGA on Intermittent Communities Dataset 71

Figure 4.8. Obtained nondominated solutions with CD-INSGA on Merging and Splitting Dataset..... 71

SYMBOLS AND ABBREVIATIONS

Symbols and abbreviations used in this study are presented below along with their explanation.

Symbol	Explanation
%	Percent

Abbreviation	Explanation
CDE	Crowding Distance Elimination
CD-INSGA	Community Detection using an Improved NSGA
CID	Community Internal Density
CX	Cycle Crossover
DNA	Dynamic Network Analysis
DYNMOGA	DYNamic Multi-Objective Genetic Algorithm
EA	Evolutionary Algorithm
EDA	Electronic Design Automation
GA	Genetic Algorithm
LPA	Label Propagation Algorithm
Md	Maximum depth
Mp	Mutation probability
NC	Normalized Cut
NMI	Normalized Mutual Information
NSGA	Non-dominated Sorting Genetic Algorithm
PLDLS	Parallel Label Diffusion and Label Selection
SBX	Simulated Binary Crossover
SC	Snapshot Cost
SNA	Social Network Analysis
$\boldsymbol{T_p}$	Transfer probability
TC	Temporal Cost
TMOGA	Feature Transfer-based Multi-Objective Optimization Genetic Algorithm

1. INTRODUCTION

Graph theory is a branch of mathematics that involves the study of graphs, which are mathematical structures used to represent relationships among objects through vertices or nodes and a set of connections among the objects, referred to as edges or arcs. Mathematically, a graph G can be expressed as (V, E), where V represents a set of vertices and E indicates a set of edges connecting those vertices. The edges can be either directed or undirected, depending on the nature of the connection between the vertices.

There are 2 main components of a graph:

- Nodes: In a graph, nodes are used to represent tangible entities such as people, places, and things, as well as intangible entities like topics, concepts, or ideas. Nodes can also be referred to as vertices in graphical terms and are depicted as points in a graph. In Figure 1.1, three nodes can be observed, each representing a person. The text inside the nodes is used to signify their identity.

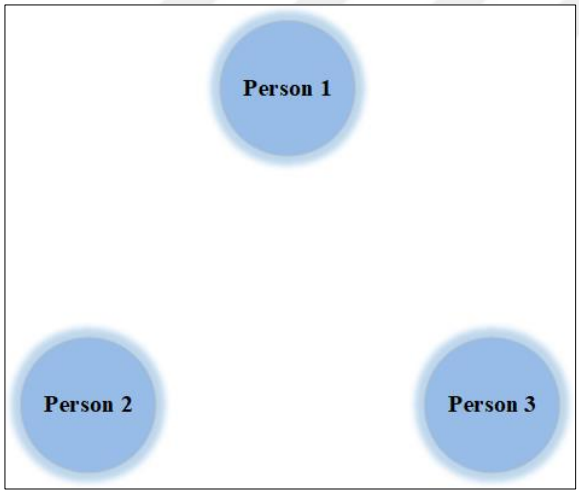

Figure 1.1. Graphical representation of nodes

- Edges: When there is a relation between nodes, an edge is used to define it. The edge represents a specific relationship that explains why the nodes are connected. Inside a graph, the edge can be represented using a line or an arrow, depending on the nature of the existing relationship. In Figure 1.2, we depict a simple relationship among colleagues using edges between the three people.

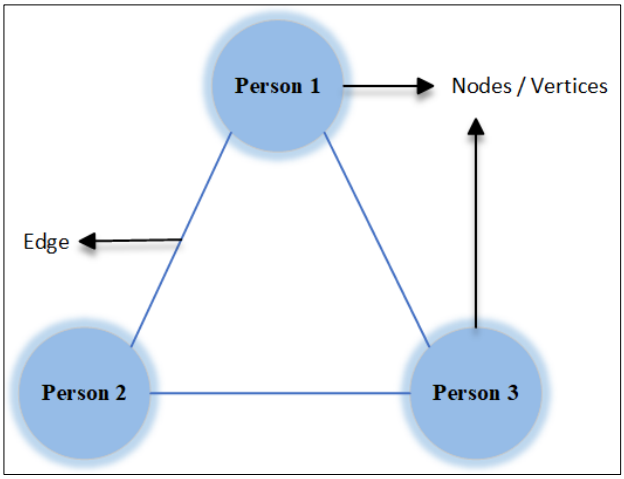

Figure 1.2. Edges between the nodes

The study of graphs includes the investigation of the properties and characteristics of a graph and the development of algorithms to solve related problems. Graph studies involve the examination of various properties existing in graphs, such as:

Connectivity: The connectivity of a graph indicates the degree to which the vertices present in the network are connected. A graph is considered connected if any two or more vertices are connected to each other. If the graph is not connected, it can be divided into connected components.

Degree: The degree of a node or vertex in a graph indicates the number of edges connected to that vertex. In a directed graph, the count of incoming edges represents the indegree of a vertex, while the number of outgoing edges is represented as an outdegree.

Path: A path represents the sequence of vertices connected by edges. The number of edges in a path is known as the length of that path.

Cycle: A cycle is formed in a network when a path starts and ends with the same vertex.

Tree: A connected acyclic graph is known as a tree in graph theory.

Graphs are used in a variety of applications, some of which are discussed below:

Computer Science: In computer science, graphs are used to represent different kinds of networks, such as social networks, the internet, citation networks, and transportation networks. In social networks, graphs depict societal networks, friendship networks, collaboration networks, etc. Studies carried out help understand social behavior and the development of these networks. Various algorithms, including shortest path algorithms,

community detection algorithms, and network flow algorithms, are used to solve and gain a better understanding of social networks. One of the most popular applications, Google Maps, utilizes graphical knowledge, representing various possible destination locations as nodes or vertices and roads as edges. Graph theory is applied to find the shortest path between nodes.

Graphs are also helpful in defining the flow of computation and data organization in computer science. The relationship between interconnected computers in a network uses graph theory principles. Network security also relies on graph theory.

Biology: Graphs are a highly effective way to represent protein-protein interaction networks, gene regulatory networks, and metabolic networks (Atay et al., 2017). Transcriptional regulation networks also use graph theory to understand and analyze these networks. In these networks, genes or proteins are represented by nodes, and edges between them represent interactions—for example, regulatory interactions between transcription factors and their genes. The study of networks in biology helps us understand the flow and behavior of diseases in the human body. Network studies have assisted biosciences in gaining a better understanding of biological processes and producing better drugs.

Electrical Engineering: Graph theory is broadly used in electrical engineering to design, analyze, and optimize electrical systems. It helps model and analyze electrical circuits where circuit elements are represented as nodes and edges between them show connections between the elements. Power grid systems can also be represented with graphs, where generators, transformers, or other electrical components are nodes, and edges between them represent existing transmission lines. Graph theory is also widely used in Electronic Design Automation (EDA) to optimize and design electronic circuits. Overall, graph theory is a powerful way to model complex electrical systems and has become essential for engineers in many applications.

General: Generally, graphs are used to map routes between different cities. Graphs help to represent transportation networks such as railways, roads, airways, etc. Graph theory aids in the optimization of the transportation system and makes better planning and development under transportation policies. With the help of a graph tree, hierarchical information can be ordered, such as a family tree, the flow of work in an organization, the distribution of sub-tasks, etc.

The networks:

A network depicts a graphical representation of connected components, illustrated as lines connecting nodes (also known as vertices). Mathematically, it is referred to as a link chart, network map, node-link diagram, or simply a graph. The objects within the network are typically portrayed as nodes, also known as vertices, depicted as points on the graph. The connections between these nodes are represented by lines, known as edges, linking two or more nodes together. Networks can be used to model various systems in the world, such as the internet, where computer systems or devices are represented as nodes, and the links between them are depicted as edges.

This research focuses on social networks, which consists of individuals or groups linked by various types of connections. Dynamic networks are associated with time, having a different graph for every period. The area of study related to dynamic networks is known as Dynamic Network Analysis (DNA). There are two important aspects related to this field. The first one is the statistical analysis of DNA information. The second one is the employment of simulation to solve the problems of network dynamics. Dynamic networks differ significantly from traditional social networks because they are dynamic, larger, multiplex, and multimode networks, which can also have different levels of uncertainty. Another major difference is that DNA considers the interactions of the conditioning structure of social features and network behavior into account.

DNA must include temporal analysis, defined as the study of a variable in a data set over some time period. From the perspective of computer simulation, the nodes present in DNA behave like atoms of quantum theory, which can be probabilistic. In Social Network Analysis (SNA), the nodes are fixed. The nodes of the DNA model can learn according to the change in properties over time. For example, an employee of a company has the ability to learn and get promoted to a valuable position while replacing someone else, or if one terrorist is captured, it can lead to catching the rest of the terrorists as well. The change transfers from one node to the other inside the network and so on.

Sampson's Monastery Study is one of the most remarkable cases of DNA. Sampson's Monastery Study is a research study conducted by Robert J. Sampson during the latter half of the 1960s and the early part of the 1970s. It is considered a seminal work in the area of dynamic network analysis (DNA) and was published in several journal articles (Sampson &

Laub, 1997). These articles provide an in-depth analysis of the study's methods, findings, and implications for understanding dynamic social networks.

Topic Description

The study of networks encompasses the investigation of communities and community detection. A community is established when a cluster of nodes, which display a high degree of similarities in their properties, are interconnected to form a cohesive group. The nodes within a community have a more intense interaction with each other as opposed to those in other communities. These interactions can be revealed through the utilization of similarity or distance measurement methods between the node participants. Community detection in dynamic social networks is the process of identification of such interactions over time as the structure of the network keeps changing. Here, the network structure keeps changing which impacts the change in community structure. The nodes keep leaving and joining different communities which makes the communities merge or split with the passing time. Therefore, community detection in dynamic social networks includes the identification of communities that are stable over time while capturing the changes involved in the network structure. Social networks are modeled as graphs, in which, the nodes show the individuals, organizations, or other entities while the edges represent some relationship between them. These relationships can be of various types such as collaboration, friendship, or affiliation.

Problem Statement

Identifying communities is a vital aspect in revealing the architecture of networks to obtain a more comprehensive knowledge of their characteristics and actions. Several algorithms and techniques exist to identify communities within a network, many of which utilize optimization methods to optimize one or more objective functions. Among these functions, Modularity and NMI are the most widely used.

It is widely recognized that the world has become a global village, with individuals and communities interconnected on a global scale. In the area of Social Network Analysis, the composition of communities within a network is of utmost importance. Researchers have been exploring various methods for detecting and analyzing the communities and clusters formed within a network to uncover hidden information. However, traditional community detection methods require a complete understanding of the network under study, rendering them inadequate for the majority of real-world networks. With the rise of large online social networks, researchers face a challenge in developing community detection methods that only

require local information. Community detection plays a pivotal role across various contexts. It enables the classification of nodes based on their structural position within their respective communities. It reveals the hierarchical structure existing within actual networks and enhances the efficiency of network data analysis, processing, and storage procedures. Moreover, community detection has numerous practical applications. In social network studies, communities depict clusters of people with common interests, hobbies, connections, or backgrounds. In networks of purchases, communities can symbolize clusters of consumers with similar purchasing habits, which can aid in the establishment of effective recommendation systems. In citation networks, communities can represent groups of papers focused on the same research topic, leading to the identification of scholars and connecting them for collaborative research. In medical sciences, brain networks can have communities of interconnected nodes, which can aid in the calculation of local computations and provide insight into the structural parts of the brain (Alamsyah & Rahardjo, 2014). Additionally, community detection helps in identifying important biological elements, such as cancer driver genes found in cancer patients, which can assist in the treatment of the disease.

Community detection in dynamic social networks introduces various challenges for researchers. A few of these challenges are discussed below:

- Data Sparsity: In a dynamic social network, edges can be sparse, meaning that some nodes might not be connected to each other, resulting in a network that is not fully connected. The sparsity can lead to complexity and false results in community detection.
- Scalability: Dynamic social networks can be extensively big, having millions of nodes and edges. Therefore, the community detection needs to be scalable enough to handle enormously large data.
- Dynamics: The structure of the network can change promptly over time, which presents a great challenge in the identification of stable communities.
- Noise: There can be noise in the network coming from several sources such as random events or erroneous data. This noise can put difficulty in the detection of true communities.

Aim of Research

Analyzing communities within a network can provide invaluable insights into their structure, shared characteristics, and commonalities. This understanding can have practical

applications in creating a world that is interconnected by prominent and well-defined communities. By examining the interactions and relationships within these communities, it becomes possible to optimize social connections and foster greater collaboration and understanding. Ultimately, the analysis of communities in a network can contribute to a more cohesive and interconnected global society (Pizzuti, 2008).

The primary aim of this research is the identification of stable groups of nodes over a dynamic time that have similar interests, relationships, or behaviors. Social networks are dynamic networks in nature where the connection between the nodes keeps changing over time. Therefore, it is significant to find communities that are stable even when the structure of the network evolves. This research on community detection in dynamic social networks aims to develop an algorithm that detects these stable communities with the accountability of network changes. This algorithm will find communities that split, merge, or disappear over some time and it will be scalable with the ability to handle large complex social networks.

Importance of the Research

Detection of communities in dynamic social networks holds significant importance in the rapidly growing global society where millions of people from almost all countries are linked together by some means. One of the most important applications of community detection in dynamic social networks includes social media analysis. Community detection algorithms are being used by researchers to detect groups of people having similar interests or beliefs, such as hobbies, religious beliefs, or political affiliations. The information extracted from these communities can be helpful in the development of targeted advertisements and a better understanding of the flow of information on social media platforms.

Another crucial application of community detection is dynamic social networks in the field of health and medicine. The researchers use community detection methods to find out groups of people having the same health issues or risks. This information can be further used to make relevant health interventions or to provide a better understanding of the spread of infectious diseases like the Coronavirus that broke out recently in December 2019.

Apart from these applications, community detection in social networks is helpful in a variety of fields including marketing, finance, research, and transportation. By detecting the stable communities in the networks, researchers can have a deep understanding of the behavior,

priorities, and actions of people or organizations, which leads to the exposure of important underlying information related to these networks.

Contributions of the Research

The notable contributions of this study are:

- Innovative Algorithm: The modification made to the crowding distance mechanism of NSGA-II (Deb et al., 2002) has impressively increased its performance. The traditional crowding distance method of NSGA-II has been replaced by the Crowding Distance Elimination (CDE) mechanism (Liu & Chen, 2019). CDE helps to preserve the diversity of the NSGA-II algorithm by taking out non-dominated solutions with increased efficiency.
- Improved Operators: The crossover operator of NSGA-II has been enhanced by using the Cycle Crossover (CX) strategy (Puljić & Manger, 2013). The CX strategy helped to eliminate local optimization problems in the solution. Additionally, the Normalized Cut (Dhillon et al., 2004) has been added as a third objective function for efficient optimization of the solutions with Modularity (Girvan & Newman, 2002) and Normalized Mutual Information (Danon et al., 2005). The use of these three important objective functions results in a more accurate and comprehensive measurement of the discovered subgroups.
- Enhanced Strategy: A significant reduction in time complexity has been achieved through the implementation of transfer-based learning, which transfers useful information from previous generations to the next generations for preservation.

Research Assumptions

While conducting this research some assumptions were made, which are considered important to be mentioned. These assumptions hold significant importance as they guide the research work and provide a framework for understanding the results. The assumptions made during this research are given below:

- Data Authenticity: This research assumes that the data used for the experimental purpose is valid and provides an authentic dynamic network. The data used in the research is publicly available and has been made using the collaborated work of other researchers.

- Information Reliability: The assumption that the information, data, and results used from previously done research work are valid. The algorithms and methods used provide accurate results. Moreover, it is assumed that the algorithmic results used for comparison of results with the proposed algorithm are reliable.
- Community Stability: The assumption that the communities being formed are relatively stable over time, despite the changes in the social network. The algorithm should be able to find stable communities that persist over numerous time steps.
- Community Homophily: The assumption that the nodes inside a community are more connected to each other as compared to nodes outside of the community. Given this, the algorithm should be able to find out the patterns of connectivity that are according to the concept of homophily.

Research Limitations

When this research on community detection in dynamic social networks was being carried out, some limitations arose. These limitations may affect the validity, generalizability, and reliability of the research information. The possible limitations of this research work are given below:

- Data Quality: One of the biggest limitations in this research is the quality of the data being used. The data produced to conduct the research may contain missing values, noise, and inaccurate information that may impact the accuracy of the results.
- Scalability: The scalability of community detection algorithms is also a limitation in this research. The algorithm may not be suitable for immensely large data networks. This limitation may result in biased or limited results caused by the incapability to analyze the entire network.
- Lack of Ground Truth: There can be cases where there is an unavailability of clear ground truth or benchmark for the evaluation of the community detection algorithm. In the real world, the network sometimes does not come with pre-defined labels that can show the community membership of each node. In such cases, there is difficulty in establishing a ground truth for the evaluation of the community detection algorithm. Also, in the case of evolving networks and overlapping communities, the definition of ground truth becomes very difficult.
- Generalizability: The results of community detection research in dynamic social networks may not be generalizable to other forms of community detection. The social

networks can be very different in terms of context concerning culture, demography, or geography which can put a limitation on the generalizability of the findings of the performed research.

Organization of Research Document

The paper is organized into individual sections. Section 2 offers a summary of the relevant literature and prior research in the field. Section 3 clearly defines the problem statement and presents a detailed explanation of the fundamental concepts used in the proposed method by outlining the technical processes and methods put forth. Section 4 evaluates the effectiveness of the proposed method through experimental testing and the presentation of results. Finally, Section 5 offers a conclusion of the entire paper and provides a brief outlook for future work.

2. LITERATURE REVIEW

This study focuses on methods of community detection with multiple objectives in social networks. The field of community detection has gained increasing attention from researchers, with various algorithms being modified to improve their performance and complexity. The utilization of multi-objective functions in community detection has proven to be an effective and efficient approach. This section provides a review of recent research advancements in the area of multi-objective community detection.

The work of Deb et al. on multi-objective evolutionary algorithms is noteworthy, particularly their Non-dominated Sorting Genetic Algorithm (NSGA-II) (Deb et al., 2002). This method builds upon NSGA (Bagchi, 1999), incorporating several modifications that result in improved performance. One of these improvements is the implementation of elitism, which reduces computational complexity from $O(MN^3)$ to $O(MN^2)$. Another key feature of NSGA-II is the elimination of the sharing parameter used in NSGA, and its replacement with the crowded-comparison approach. This strategy entails arranging populations by the ascending order of magnitude of each objective function value. Additionally, the algorithm employs a selection operator that merges the parent and offspring populations to form a mating pool, and then chooses the top N solutions while preserving diversity in the solutions. Experimental results of NSGA-II show that it is capable of finding superior solutions and converging near the Pareto-optimal front.

Feature Transfer-based Multi-Objective Optimization Genetic Algorithm (TMOGA) has been proposed by Zou et al. in their research (Zou et al., 2021). This approach builds upon the Non-dominated Sorting Genetic Algorithm (NSGA-II) algorithm and includes modifications that enhance its performance. The central aim of TMOGA is to obtain valuable features from previously identified community structures and apply this knowledge to the current optimization procedure, thus enhancing the evolutionary algorithm. The authors also established a theoretical framework to evaluate the community detection problem, which was used to show the validity of the suggested method. The empirical output indicates that TMOGA outperforms state-of-the-art dynamic network community detection algorithms.

The authors, Zhao et al., put forward a solution to the problem of community detection in large networks using graph compression (Zhao et al., 2021). The approach involves repeatedly combining nodes with either 1 or 2 edges into neighboring vertices of higher degree to obtain a compressed graph. Subsequently, the density and worthiness of all nodes

are assessed to identify community seeds. Using these measures, the initial community seeds and several communities are established. The community structure is then derived from the compressed network through seed expansion and transferred back to the original network. The outcome of tests performed on big social networks confirms the effectiveness of the suggested method.

Roghani et al. introduced a parallel label diffusion and label selection (PLDLS) algorithm (Roghani et al., 2021). PLDLS is built upon Spark and leverages an improved and efficient version of the Label Propagation Algorithm (LPA) (Raghavan et al., 2007) that eliminates the need for random parameter tuning. The algorithm consists of two phases of label diffusion among core nodes and employs multiple factors to compute the importance of nodes. A new method is then used to select the important nodes. The first phase leverages parallel label diffusion to disperse labels into two levels, recognizing that the nodes that create triangular structures are probable to be part of the same community. The second phase involves a parallel and iterative process to assign labels to remaining nodes, followed by a rapid, parallel merging stage to generate more precise and dense communities. Experimental results indicate that PLDLS outperforms other algorithms regarding both effectiveness and efficiency.

Lin et al. proposed a novel and efficient method called FacetNet (Lin et al., 2008) for discovering communities in dynamic networks. This method is unique in that it unifies the detection and evolution of communities, regularizing the output based on the temporal smoothness of the evolution. This results in the discovery of communities that are best suited to the underlying data and its temporal evolution. The authors approached this problem as a non-negative matrix factorization issue and converged to an efficient solution with a minimal processing time requirement. The experiments conducted showed that FacetNet can effectively uncover meaningful communities, albeit with room for improvement regarding the level of excellence in the obtained results.

The authors Folino et al. put forth a multi-objective evolutionary method for finding communities in dynamic networks called DYNMOGA (DYNamic Multi-Objective Genetic Algorithms) (Folino & Pizzuti, 2013). DYNMOGA employs temporal smoothness like a multi-objective problem for community detection and uses a genetic algorithm-based method to generate solutions that represent the best compromise between the quality of the clusters at each time interval and the progression from one interval to the next. The authors suggest that DYNMOGA can function as a comprehensive blueprint for evolutionary

clustering. Experimental tests indicate that DYNMOGA surpasses the current leading algorithms in comparison.

Yin et al. came up with a community detection method which is based on clustering called DYN-MODPSO (Yin et al., 2021). The authors claimed to address the two main challenges in clustering, namely the lack of error correction leading to error accumulation and result-drifting problems, and the NP-hardness of modularity resulting in low efficiency of results. To resolve these challenges, DYN-MODPSO enhances the particle swarm method and the evolutionary cluster formation function. The authors employ a recent future prediction technique for the initial grouping and a non-repetitive random walk technique in the population initialization procedure to generate more diverse individuals during each cycle. Additionally, the method integrates a modified interference operator and a multi-individual crossover to eliminate the local optimization problem. Experimental results demonstrate that DYN-MODPSO outperforms all other algorithms in comparison.

Besharatnia et al. found a metaheuristic algorithm for multi-objective detection of communities in dynamic networks (Besharatnia et al., 2021). The authors put forth the MOGWO-LP (Multi-Objective GWO LP) algorithm, which integrates an improved version of the Grey Wolf Optimizer (GWO) algorithm (Mirjalili et al., 2014) combined with an upgraded version of the classic Label Propagation (LP) algorithm (ZhuЃ & GhahramaniЃн, 2002) to identify communities within the network. The algorithm considers two objectives to detect dynamic communities: (1) community structure quality at any period, which is measured using Modularity (Girvan & Newman, 2002) as a fitness function, and (2) the temporal cost (TC) for evaluating the similarity of communities formed in different snapshots, which is measured using normalized mutual information (NMI) (Danon et al., 2005). Although the MOGWO-LP algorithm demonstrates efficient performance, it has some disadvantages. Like other metaheuristic algorithms, it may get trapped in local optimal solutions, and its performance is poor for unstable and dense communities, as it is an incremental algorithm. Thus, the method is not suitable for unstable datasets.

In a detailed project put forward by Panchal A. an enhanced metric is introduced which will allow the use of a new measurement method (Panchal, 2021). This method involves the consideration of shared communities between the nodes that are connected by a new edge in the network. Additionally, as a preprocessing step, good seed nodes are discovered by an offline non-overlapping community structure present inside a subgraph. The proposed method is an improved version of Community Detection via seed-set expansion on graph

streams (COEUS) (Liakos et al., 2017). The algorithm has two input parameters, a stream of edges and the seed nodes. During the working of the algorithm, three parameters are being tracked throughout the run time. These three parameters include the degree of nodes, the community degree of nodes, and the communities being formed. The proposed algorithm is being evaluated using extensive datasets to prove its efficiency. Experiments are performed using a new dataset of web graphs and other extensively used datasets. In the experiments, f1 scores have been used for the datasets having ground truth. As we examine the results, it can be noticed that the proposed algorithm performs 17% better than the traditionally used common methods. In the case of the new graph dataset; the method of overlapping modularity has been used. These experiments also yield a high score of 0.71 modularity which makes an increase in performance by 10%. As a suggestion for future work, the author states that the algorithm can be further improved in cases of ground truth and simple cases where the proposed algorithm doesn't perform better as compared to COEUS. The author suggests using a combination of different metrics, for example, community participation with the common communities method to increase accuracy.

A motif-based embedding label propagation algorithm (MELPA) has been proposed by Li. Et al. The authors believe that the network motifs are building blocks of a network that can be very helpful in exposing the deep structure of the networks (Li et al., 2022). Label propagation algorithm works in almost linear time complexity, however, the label updating method of LPA remains an unsolved issue. Hence, this paper comes out with an improved method of community detection which uses LPA and an improved technology for the label updating procedure. In MELPA, first of all, the topological structure of the network is reconstructed by integration of higher-order topology and lower-order connectivity features. Here, higher-order topology is gained by exploring the network motifs. In the second phase of the algorithm, a label propagation model is designed in accordance with the nodes. At this time, a new label updating rule gets generated which is according to the reconstructed network. This rule takes into consideration the frequency present among the neighbor labels, the effect of nodes, propagation attributes, and the compactness of nodes to update the node label. By using this method, the random labeling of LPA is overcome and provides a better identification of community structure with stability. At last, the authors perform many experiments on synthetic as well as real-world networks to show the efficiency of MELPA in detecting communities in complex networks.

A graph segmentation merged with a node labeling approach has been proposed by Zhou, H., & Zhang, Y. In this algorithm, first of all, the graph is segmented by labeling the positive and negative sample nodes (Zhou & Zhang, 2022). After this, the enhanced graph network model is used to get the local community which has the starting node inside. In the last stage, local community optimization is performed by adjusting the membership degree threshold to an increased focus on nodes that exist outside of the community and get efficient community detection results. The database sets used for experimental purposes are Flixter, Yelp, and Douban datasets. The results show that the algorithm provides fair values of modularity and is very robust.

In a research (Shang et al., 2022), the high-order details and information of the network are considered to perform local community detection. In local community detection, the ultimate goal is to find a local community that contains the seed. The authors mention the researcher's ignorance of high-order information, also called network motif, which plays a significant role in the construction of a network. Most of the methods only consider edge information for the extension of the local community. Some methods concentrate on network motifs, but this time, the edges connected sparsely have been ignored because of this, some edge points fail to extend. Therefore, the authors come up with a new method that considers higher-order structure and edge information (HSEI). First of all, different methods are applied to select the node that joins the local community by the motif degree of the seed. Secondly, an enhanced modularity function is developed that is based on motifs. This new method helps in the extension of the local community and makes sure that the connections are firm. Also, a newly modified community central node is introduced which is also motif-based helping in the extension of the central region of the local community. In the HSEI method, edge information is being used to find connection strength existing between the nodes and communities. This helps the algorithm to find more outright local community members. In the end, the algorithm is tested using artificial networks having different parameters and real networks. The results are compared with five other algorithms which prove that HSEI gives superior results as compared to all other algorithms.

Tharwat et al. also put their focus on finding communities in dynamic social networks using a searching algorithm based on a non-dominated approach called the non-dominated sorting-based community detection with a dynamical awareness (NDS-CD-DA) algorithm (Tharwat et al., 2021). In state-of-the-art methods that utilize meta-heuristic search with multi-objective optimization generally, NSGA-II is employed. In NSGA-II, a collection of non-

dominated solutions is found utilizing the non-dominated sorting and crowding distance methods. Non-dominated sorting is used to exploit the solutions and crowding distance is used to enable exploration. In the proposed algorithm, multi-objective optimization is used with NMI and Modularity as the objective functions. These objective functions explore the community detection quality of static and dynamic aspects to get a smooth transition over time. Despite the improved outcomes in terms of the excellence of the communities detected, there is a limitation of this approach. The proposed method uses a large number of iterations before convergence.

An improved Ant Colony Optimization (ACO) is presented in (Mu et al., 2019) built on the principle of decomposition in which, Ratio Cut and Negative Ratio Association (NRA) are minimized for community detection problems. The proposed ACO algorithm based on the idea of decomposition (MOACO/D-Net) breaks down the solution into several single-objective problems, where every individual problem gets inspected by an ant. Division of the ant colony takes place into several groups. When non-dominant solutions are updated in a group, the pheromone of that group also gets updated. The ants upgrade their current solution based on a better neighbor which can be one of their neighboring groups. As a result, the last external archive is obtained having the non-dominated solutions, where each solution portrays a discrete division of the network. Despite the efficiency of the suggested method, it requires further improvement in terms of efficiency and application on the overlapping detection problems.

Interesting research was done by Shaik et al. in which they proposed two modifications of a three-objective methodology by enhancing the NSGA-III algorithm to explore the structures of communities present inside a network (Shaik et al., 2021). The first variant proposed in this paper is called NSGA-III-KRM, in which they have used three objective functions naming Kernel k means, Modularity, and Ratio cut. The second method that has been proposed is called NSGA-III-CCM in which Community Score (CS), Modularity, and Community Fitness are considered objective optimization functions. By the experimental results, the authors prove that the usage of the third objective function doesn't make the final results worse than that of two objective functions. They have also proposed a method for ranking of Pareto solutions using the ratio of Inverted Generational Distance (IGD) and hyper-volume. In this method, the ratio is directly proportional to the Pareto set, that is, a higher ratio indicates a better Pareto set.

A new mechanism of community detection in complex networks has been proposed in which an improved version of the Cuckoo Search Algorithm (SCO) has been used in combination with a Genetic Algorithm (GA) to perform community detection (Shishavan & Gharehchopogh, 2022). The limitations of CSO including delayed convergence, premature convergence, and being stuck in the local trap have been resolved in the improved algorithm. The use of GA has been preferred to give strength to the community detection procedure in terms of exploitation and exploration. Along with an increased speed, the GA algorithm helped in the improvement of the accuracy of results obtained from the CSO algorithm. In the proposed method, the CSO algorithm explores the search space efficiently while the GA improves the performance in terms of accuracy and speed. The authors have used modularity and NMI as the objective functions. The experiments were carried out on six different types of real complex networks. The results of the proposed algorithm were compared with the results obtained using prominent algorithms including GA, Grey Wolf Optimizer (GWO), Artificial Bee Colony (ABC) algorithm, and CSO over several iterations. The results of modularity and NMI show that the proposed algorithm performs better than the other algorithms in comparison. The overall performance of the proposed algorithm was 54% better in terms of modularity and 88% in terms of NMI as compared to the other algorithms.

A detailed search has been done in which the author discusses six metaheuristic algorithms proposed in recent years to solve community detection problems (Koc, 2022). The six algorithms discussed include the Archimedes Optimization Algorithm (AOA), Coot Bird Natural Life Model (COOT), Atom Search Optimization (ASO), Harris Hawks Optimization (HHO), Arithmetic Optimization Algorithm (AROA), and the Slime Mould Algorithm (SMA). Archimedes Optimization Algorithm (AOA) is a metaheuristic algorithm which has immersed objects as a population. Similar to other metaheuristic algorithms, AOA starts the search process with the generation of an initial population of objects containing random volumes, accelerations, and densities.

The Coot Bird Natural Life Model (COOT) algorithm was proposed in 2021 by observing the behavior of coot birds. Coots perform regular or a sequence of irregular movements on the surface of the water. A group of coots goes towards the food with a few coots in the front row, which are considered the leaders. In COOT, four basic motions have been considered which include the side-to-side movement, chain movement, the adjustment of position by the leader movement, and the movement directed by the group leaders to adjust the position.

Atom Search Optimization (ASO) was developed in the year 2019 inspired by the motion of atoms. The motions of atoms considered in the algorithm are interaction force, potential function, and geometric constraint force. ASO obtains the initial population being called atoms. In the algorithm, two vectors are being used for each atom which are called position and velocity. The location of atoms shows a solution calculated with the mass of the related atom, a heavier mass represents a better solution, and vice versa.

Harris's Hawks Optimization (HHO) algorithm is an optimization algorithm based on Harris's hawks chasing collaboratively. In this method, a group of hawks collaborate with each other and attack from several sides to dodge and confuse the prey. The hawks have a unique strategy of chasing and attacking the prey according to the scenario. The HHO algorithm uses the same strategies to compensate for surprise attacks, the nature of the scenario, and the attacking method of the hawks to explore the solution space. Along with these, the HHO algorithm has an incredible strategy for making the transition from exploration to exploitation just like Harris's hawks.

Arithmetic Optimization Algorithm (AROA) is an optimization-based algorithm that uses arithmetical operations to solve optimization problems instead of computing derivatives. AROA is inspired by arithmetic operations including basic operations like division, multiplication, addition, and subtraction to find out the optimal solution in a set of candidate solutions. As identical to other optimization algorithms, AOA also contains the stages of exploration and exploitation. Here, exploration is the process of searching for a set of candidate optimal solutions in a search space and exploitation is the stage of finding out the optimal solution and converging to it in a range of promising candidate solutions.

Slime Mould Algorithm (SMA) has been proposed in 2020. It is based on the way a slime mould is released. The SMA works on the principle of food path along with a positive-negative feedback system. Slime mould are capable of adjusting their search path on the basis of fitness of the available food. SMA consists of three phases in terms of a mathematical model which includes approaching the food, wrapping of food, and grabbling the food. During the approach to the food stage, the smell present in the air engages the slime moulds towards it. In the stage of wrapping the food, mathematical calculations of getting shrink for the food are calculated. At last, in the grabbling food stage, the cytoplasmic flow that is present in the vessels is altered by the biological oscillator. The slime mould depends mainly on the wave that is produced to search for a food position having a better concentration.

The author has proposed a CommunityID approach that has been used as an objective function to minimize the time cost during the calculation of fitness values of all the algorithms mentioned above. Experiments are performed to compare these six algorithms to find out the best algorithm in terms of computation and time complexity. According to the findings of this research, COOT comes out to be the best algorithm due to the good quality of results and least time cost. The experiments also prove that the algorithms perform faster when the CommunityID function is used, hence the author recommends using this objective function while finding communities in large networks.

Community detection is crucial not only for purposes such as commercialization, recommendations, and mutual friend discovery on social networks but also holds great significance in real-world human networks. The rapid spread of COVID-19 serves as a reminder of the importance of understanding the structure of communities formed by humans. This knowledge could have potentially mitigated the spread of the virus. Consequently, community detection is of vital importance not only in social media but in the field of health sciences as well.

Amidst the COVID-19 pandemic, numerous scholars and researchers have proposed innovative approaches and preventive measures against the virus. This text will highlight some of the recent studies and findings in this field.

In a research conducted in 2021, the authors analyzed the epidemic spread by examining social networks according to information gathered in the initial quarter of 2020 when the COVID-19 epidemic first emerged globally (Wickramasinghe & Muthukumarana, 2021). The world is treated as a network and visual observations are conducted to comprehend the spread of the coronavirus in different nations and the effect of foreign nations on the transmission of the disease specifically in Canada. The authors employed degree centrality to discover processes that drive links between countries. Community detection algorithms, including Label Propagation, Infomap, Luvain, and Spinglass, were used, with NMI and Adjusted Rand Index serving as metrics for performance evaluation. The in-degree and out-degree spread of the coronavirus were calculated for the overall world and Canada. The results for the world network were found to be more promising compared to the Canada network, with the Spinglass and Luvain algorithms performing better than the other algorithms. The research could be further improved by using community detection-based algorithms on individual contact tracing data to detect the transmission of COVID-19 in local communities.

In the same year, Wickramasingh (Wickramasinghe, 2021) presents her thesis on the application of community detection to COVID-19 data in social networks. The author conducted a thorough analysis of the COVID-19 pandemic's spread by analyzing social network data from the outbreak of the disease globally and in Canada. The spread of the virus was evaluated in terms of communities, where the virus was transmitted from one community to another. To do this, the author leveraged community detection algorithms that applied to directed graphs, including Spinglass, Label propagation, Infomap, and Louvain algorithms. Simulation studies were performed to evaluate the efficiency of the algorithms and to assess the effect of network density through the use of a randomly partitioned graph generator and a combination of Gaussian distributions.

Prabhu et al. introduced the Social Infection Analysis Model (SIAM) to control the COVID-19 epidemic through community detection methods. Their methodology comprises three primary components (Prabhu et al., 2021): first, the modeling of societies using social network technology and analyzing them at a large scale; second, the detection of communities in the network; and third, the examination of the transmission of the illness within a local population over time. To identify consistent communities within the network, the Girvan-Neumann method was favored by the authors, while for complex networks, they used the label propagation method.

In their experiments, they considered a population of 100 individuals. The communities, in which people can move freely, were treated as random networks, with two common probabilities, 0.05 and 0.2, found. A network with a high probability is a dense network where individuals have numerous random interactions, increasing the probability of disease spread due to a low average path between nodes. The authors used simulations in NetLogo to observe network properties and analyzed the transmission between communities using the NK problem, while the behavior of vulnerable nodes was uncovered through agent-based models. The study puts forth a substantial effort to mitigate the spread of COVID-19 and suggests that further research in this area could significantly resolve the viral disease's transmission. Azad et al. conducted a study on COVID-19 transmission through different communities (Azad & Devi, 2020). The authors utilized the Newman-Girvan Modularity (Girvan & Newman, 2002) communities by systematically removing edges within the network while prioritizing edges between communities. The study employed the Louvain method to calculate modularity, a two-step iteration process that combines vertices to form communities that optimize modularity by evaluating the node connection density within the

community versus a random network. The procedure is repeated until the highest modularity value is achieved. The authors utilized travel histories of COVID-19 cases as the data, analyzing 1386 cases in total, with 373 being from international contacts and 1013 from domestic contacts.

Observing the research in this field, it has been noted that similar fitness functions have been utilized in all studies, with the most commonly used being Modularity, Negative Ratio Association (NRA), community fitness, Ratio Cut, and community score. Despite their crucial role in community detection, these fitness functions have received limited attention. There is a potential for advancements in the field through the development of a better fitness function. In this interest, a novel and flexible fitness function was introduced that facilitates the identification of communities with distinct characteristics (de França & Coelho, 2015). The immune-inspired algorithm, referred to as cob-aiNet[C], was utilized to identify both disjoint and overlapping communities across different networks. The results were promising, yielding partitions that were more aligned with the ground truth compared to those obtained through Modularity.

Zhang et al. also proposed a fitness function that is based on the Local Fitness Method (LFM) (Zhang & Yan, 2019). They enhanced the LFM by recognizing that it generates unnecessary nodes due to its backtracking step, leading to the introduction of an extended version, called LFMEX. The LFMEX uses an improved fitness measure of nodes to determine whether the adjacent node has to be incorporated into the community or not. The method takes into consideration the exclusive, direct, and indirect relationships between nodes and communities, eliminating the need for backtracking. After locally optimizing the fitness function, duplicate communities may still be present, reducing the value of Modularity. To address this, the authors employed an enhanced community similarity criterion based on probability theory and random graph concepts to identify similarities between communities. Upon merging similar communities, the modularity value improved. The experimental results indicated that the new fitness function improved both the modularity and NMI values in community detection.

Recently, Bitcoin, a decentralized, peer-to-peer cryptocurrency, has received widespread attention from media, researchers, and the general public. Studies have demonstrated that the Bitcoin transaction network can be effectively analyzed through the use of complex network analysis techniques, by using community detection for the identification of multiple addresses that belong to the same user. Despite its potential, there has been limited research

in this field with regard to real-world applications. This lack of research presents a compelling opportunity for further exploration and investigation into community detection techniques that can be applied to address real-world problems.

The use of community detection in relation to Bitcoin transactions has the capability to yield a significant understanding of the behavior of users and the nature of the network as a whole. Through the identification of communities within the network, researchers can gain a deeper understanding of the relationships between different addresses and users, providing valuable information for various purposes, including anti-money laundering efforts, tracking of illicit activities, and the development of new financial products and services. Given the increasing popularity and importance of cryptocurrencies, such as Bitcoin, the need for a more comprehensive and sophisticated understanding of these networks is paramount, making the research in community detection an exciting and promising area for future investigation.

3. MATERIALS AND METHODS

In this section, the methods used behind the proposed approach will be discussed in detail. As the proposed method is based on an evolutionary algorithm, first of all, evolutionary algorithms will be highlighted in this chapter.

3.1. Evolutionary Algorithms

Evolutionary algorithms are the type of algorithms that have been developed by taking inspiration from natural processes and solving by using the natural behaviors of living organisms.

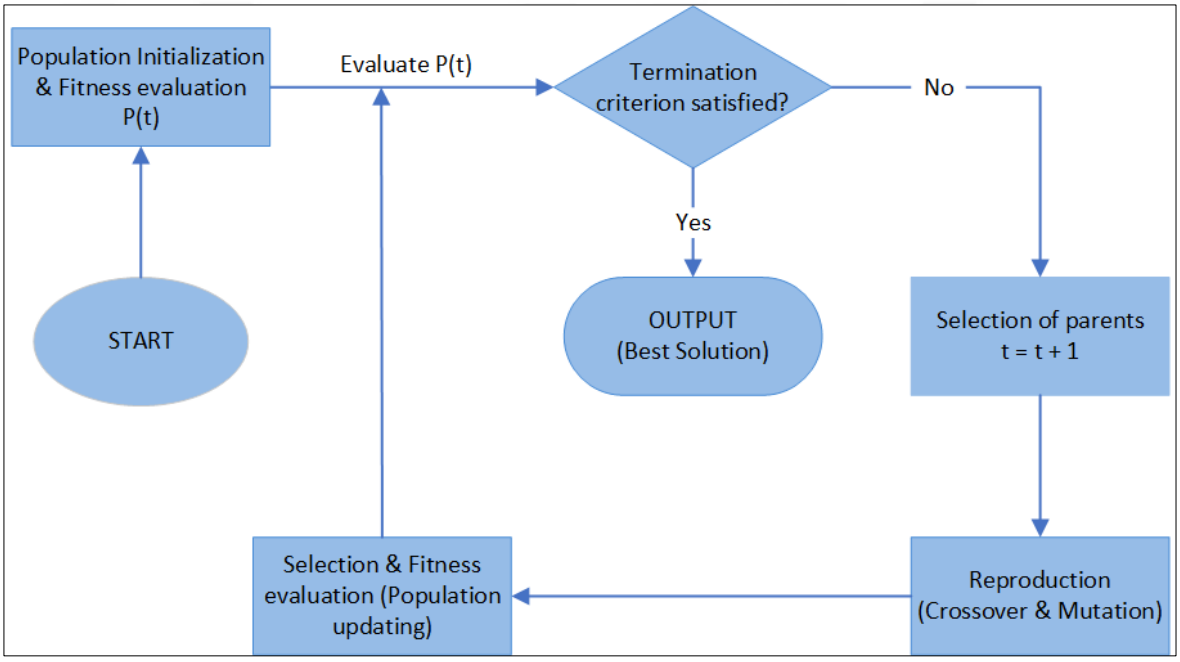

Figure 3.1. General schema of an Evolutionary Algorithm (EA)

The evolutionary algorithms follow the rules of Darwinian Evolution. In evolutionary algorithms, the solutions are considered as individual living organisms in a real population. The main concept of evolutionary algorithms is to follow the principles of biological evolution processes, for example, selection, reproduction, mutation, etc. These processes are run iteratively to make improvements in the population of candidate solutions to a problem. The population starts with a set of randomly formulated individuals and then is run through the process of evolutionary operators to find the fittest solution (Figure 3.1).

The main steps involved in an evolutionary algorithm are as follows:

i. Initialization: Initialization is a process where a population of individuals, which are sometimes represented as strings of values or chromosomes, is generated randomly. Each individual in the solution represents a potential solution to a problem.
ii. Evaluation: The fitness of all individuals is checked by using one or more objective functions or fitness measure methods. The function used for assessment checks the potential of each individual to solve the given problem. The individuals with top fitness values are considered for further evaluation.
iii. Selection: The individuals are selected from the population to take part in the reproduction process which is decided according to the fitness values they carry. The most commonly used selection methods include tournament selection, rank-based selection, and roulette wheel selection. The goal of this process is to select the fittest individuals to increase the chance of their genetic materials to pass onto the next generations. Some selection process allows unfit individuals as well to give them a chance to participate in the evolution process.
iv. Reproduction: The individuals selected in the selection process are used to reproduce offspring. This step is usually known as the crossover method. In this step, the genetic material of two-parent populations is used to generate a whole new population. After the process of crossover, the mutation is also applied usually. The reason to apply mutation is to make random changes to the child population to avoid the generation of the same offspring population.
v. Replacement: The new offspring replaces some of its individuals in the population. This is usually applied to individuals having a lower fitness value. This additional step helps maintain the diversity of solutions with additional control on the population size.
vi. Termination: The processes explained above are run in an iterative form until a specific termination condition is met. Usually, this termination condition is reaching a maximum number of generations or gaining a specific threshold value.

3.1.1. Benefits of evolutionary algorithms

Some of the very important benefits of evolutionary algorithms include:

- Increased Flexibility: The evolutionary algorithms have the ability to be adapted and modified according to the problem. Hence, evolutionary algorithms hold a great

capability to solve the most complicated problems in human life and successfully meet the goals.

- Better Optimization: A great number of solutions are considered feasible solutions from which, the best one is selected. This shows that an evolutionary algorithm is never limited to a particular solution.
- Unlimited Solutions: The classic algorithms find a solution and then keep working on improving one particular solution, however, in the case of evolutionary algorithms, a vast number of solutions are taken into account and they hold the ability to provide multiple suitable solutions to a single problem.

3.1.2. Applications of evolutionary algorithms

Evolutionary algorithms hold a prestige place when it comes to algorithms. The compatibility and efficiency of evolutionary algorithms have taken their place in enormous fields of human life. Some of the most common applications of evolutionary algorithms are listed below:

- Optimization Problems: Evolutionary algorithms are excellent at solving optimization problems in a vast number of fields including engineering design, scheduling, logistics, resource allocation, community detection, network routing, etc. Evolutionary algorithms provide an opportunity to find optimal or near-optimal solutions to problems in complex systems or high-dimensional spaces where classic optimization methods would struggle a lot.
- Machine Learning and Data Mining: Evolutionary algorithms have been very helpful in solving feature selection, model optimization, and parameter tuning in the machine learning field, and data mining tasks. These algorithms hold the ability to automatically discover feature subsets, optimize hyperparameters, and model evolution that can then be used in a variety of tasks like regression, classification, and clustering.
- Robotics and Control Systems: Evolutionary algorithms have been proven to be very significant when it comes to the field of robotics and control systems. These algorithms are employed for robot behavior learning, robot path planning, design of the robot morphologies, and parameter optimization in control systems. Hence, evolutionary algorithms can efficiently be used to create brilliant and adaptable robots to navigate complicated environments and perform smart tasks.

- Image and Signal Processing: Evolutionary algorithms have proved their efficiency in image and signal processing tasks including image reconstruction, signal denoising, image enhancement, and feature extraction. The evolutionary algorithms can optimize filter parameters, evolve image processing algorithms, and analyze patterns to modify image quality and bring out meaningful information.
- Bioinformatics and Computational Biology: Evolutionary algorithms have a number of applications in bioinformatics, and computational biology, for instance, protein structure prediction, DNA sequence alignment, gene expression analysis and genome assembly. In this field, they assist the researchers in having a better understanding of biological systems, identification of optimal sequences, modeling genetic interactions and even finding the disease carriers to cure serious diseases like cancer.
- Financial Modelling and Portfolio Optimization: Evolutionary algorithms are being used to model financial programs, algorithmic trading, and portfolio optimization tasks. They not only help in the optimization of investment portfolios but are also used for asset allocation, prediction making of stock market trends, and risk management. Such algorithms help investors and traders to make smart decisions in the trading markets.
- Game Playing and Strategy Optimization: Evolutionary algorithms have helped the human race to make strategies and intelligent agents in games. They can make up game-playing agents in several games like, poker, chess, monopoly, and several other strategic games in order to make improvements and adapt strategies according to the opponents' moves.
- Renewable Energy Systems: The evolutionary algorithms have been valuable in the optimization of renewable energy systems, for example, solar panel arrays and wind farms. They can be used to optimize the configuration and placement of energy resources, make improvements in energy efficiency, and make intelligent decisions in the operation and design of renewable energy systems.

The above applications can be considered a few of the evolutionary algorithm applications, because, their versatility and capability of tackling complex optimization problems allow them to hold a significant value in solving a vast number of scientific, computational, and

engineering tasks. In this work, evolutionary algorithms have been used to solve the optimization problem of community detection in dynamic social networks.

3.1.3. Types of evolutionary algorithms

There are many types of evolutionary algorithms and all come with their own variations and uses. Some of the most commonly used types of evolutionary algorithms are listed below:

- Genetic Algorithms (GA): Genetic algorithms are one of the most popular and widely used evolutionary algorithms. They work on the principles of natural and genetic selection. GAs mostly use binary encoding or some other representation to encode solutions and make selection decisions.
- Evolutionary Programming (EP): Evolutionary programming puts its focus on the optimization of computer programs or structures that represent computer programs. It sometimes has direct manipulation of program parameters and code where it employs self-adaptive rules of mutation. EP has its applications in fields like control systems, evolutionary robotics, and neural network design.
- Evolution Strategies (ES): The primary use of evolution strategies is continuous optimization problems. They vary from traditional genetic algorithms by the use of self-adaptive mutation rates, generally by using covariance matrixes. ES has been used effectively in different areas including robotics, function optimization, and engineering design.
- Genetic Programming (GP): Genetic programming includes the evolution of populations of computer models or programs which are then represented as trees. The population comprises program structures and evolution operators making an impact on the tree structure using crossover and mutation. GP has successfully been applied in automatic program synthesis, symbolic regression, and machine learning jobs.
- Differential Evolution (DE): Differential evolution has its basis in population-based evolutionary algorithms that work with real-valued vectors. The DE utilizes crossover, mutation, and selection processes for the exploration of search space. DE has been used in many optimization problems, typically in engineering pattern designing, function optimization, and parameter estimation.

These are just a few of the important evolutionary algorithm types, many other variants and hybrid approaches do exist that use features from other types. The choice of an evolutionary

algorithm to be used is dependent on the problem at hand, the nature of the solution space, and the requirements of the optimization problem. The evolutionary algorithm being used in this paper is GAs because community detection is an optimization problem that has a large solution space and many solutions can be made out of one problem.

3.2. Genetic Algorithms (GAs)

Genetic algorithms are the kind of evolutionary algorithms that are adaptive heuristic search algorithms based on the concepts of natural selection and genetics. These algorithms exploit the random search space with the help of historical data provided for the direction of the search space towards the region of finer performance in the solution space. The GAs are widely used for the generation of superior solutions for many kinds of optimization problems and also search problems.

The genetic algorithms use the method of natural selection which means that the species having an adaptable nature towards their environment, are the survivors and get a chance to reproduce and move on to the upcoming generation. Each generation contains a population of individuals and all of the individuals behave as a pointer in the search space and the possible feasible solution. In the programming world, each of these individuals is represented as a string of integer/float/character/bits. This string is also known as a Chromosome.

3.2.1. Concept of genetic algorithms

The concept behind genetic algorithms is completely on an analogy of genetic structure and the behavior that the individual chromosomes of the population carry. The elements of a genetic algorithm are shown in Figure 3.2. The genetic algorithms are based on the following basic concepts:

- The individuals inside a population participate to find mates and resources.
- The successful individuals who turn out to be the fittest then mate with each other for the creation of further offspring.
- The genes that belong to the fittest parent keep growing throughout the generation, which depicts that there are times when parents create offspring that come out to have better properties than either of them.
- Each of the successive generations proves to be more suitable and fit for their environment.

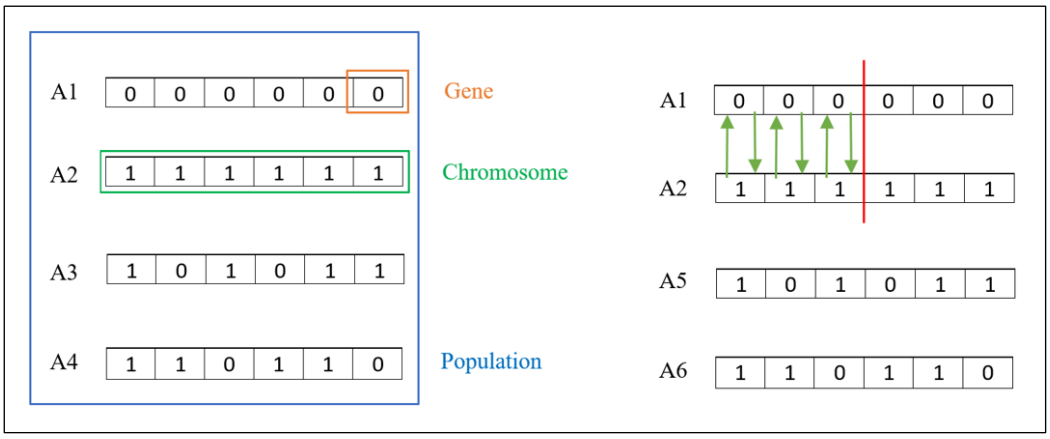

Figure 3.2. The structure of Genetic algorithms

3.2.2. The search space/area

All the individuals of the population exist within the search space where each acts as a solution to the problem in the current search space. In the programming world, each of these individuals is represented as a vector or finite length which is analogous to the chromosome of components, whereas the components are the genes. In short, different genes (variable components) join up to form a chromosome (an individual).

3.2.3. The phases of genetic algorithms

The process that the genetic algorithms follow can be generally divided into the following steps:

Initial population

The process of genetic algorithms initially starts with a set of individuals that make up a population. Here, each individual represents a solution to the current problem. Each of these individuals is made of a set of parameters (variables) which are also called genes. These genes are then joined into a string of specific length which represents a chromosome (solution).

The creation of this initial population is a very crucial step because it sets a foundation for the whole process. The initial population behaves as a starting point from here the exploration and improvement of candidate solutions begins. The common approach used for generating the initial population in Genetic Algorithms can be listed below:

- Random Initialization: The individuals inside the population are created in a random fashion. The individuals are generated by giving values to genes or elements of chromosomes on the basis of a uniform random distribution. This is the most commonly used way of generating the initial population in genetic algorithms as it introduces diversity in the population and allows the exploration of a bigger range of solutions.
- Domain-Specific Initialization: Domain knowledge or the heuristics specific to the given problem are used to generate the initial population. Depending on the nature of the problem, it is sometimes recommended to use specific initial solutions. The method of creating an initial population using domain-specific initial methods can be able to provide a fitter starting point with respect to feasibility and proximity to better solutions.
- Latin Hypercube Sampling: The Latin Hypercube Sampling (LHS) is a hierarchical sampling technique to generate initial populations. In this method, the search space is divided into several intervals from which a single sample is selected randomly from each interval to create an individual. LHS is effective in creating an even search space and also assists in the reduction of redundancy in the initially created population.
- Pre-existing Solutions: If there is a previously found solution or data in hand, then it can used as a starting point to create an initial population. Such solutions may be found from prior optimization runs, existing solutions from resembling problem instances, or from well-known good solutions. By the incorporation of already existing solutions, the convergence of the algorithm can be accelerated by providing a better starting point which has a higher chance of being closer to the optimal solution.

Fitness function

The fitness function has the job of finding how fit or suitable an individual is, which is done by assigning a fitness score to each individual. The probability of an individual participating in the upcoming reproduction depends on the fitness score it carries. The formation of fitness function holds immense importance in the working of GAs as it decides how well an individual can perform towards solving the problem. The following concepts of a fitness function are important to be mentioned:

- Formation of Fitness Function: Fitness functions should always be designed specific to the problem based on the nature of the optimization problem. It is dependent on the optimization constraints, objectives, and performance criteria of the problem. The fitness function may be a single objective function where a single criterion is evaluated or it can be a multi-objective function in which several conflicting objective functions are considered simultaneously.
- Evaluation Process: The fitness function assesses each individual present in the population with respect to its performance with the optimization objectives. Generally, an aggregate value is calculated on the basis of a set of pre-defined calculations or metrics. This evaluation process can have different phases of simulations, comparisons, or mathematical models against some known metrics which are dependent on the nature of the problem at hand.
- Fitness Assignment: The fitness values evaluated during the evaluation phase are assigned to the respective individuals inside the population. The higher the fitness value, the better the solution is. This shows a high compatibility or optimality of the individuals; therefore, the fitness values show the suitability of the individuals for selection in the reproduction phase of the selection process.
- Fitness Landscape: The fitness landscape represents the relationship of fitness values with the corresponding individuals inside the population. It determines how the fitness values can make a difference in the solution space and helps in providing a better understanding of the difficulty of the optimization problem. The shape of a fitness landscape has a great impact on the working of the GA which includes the convergence speed, search dynamics, and the probability of finding out the optimal solutions.
- Fitness Sharing and Scaling: In some problems, the fitness sharing and scaling methods are also used to make improvements in the diversity and exploring capability of the population. Fitness sharing allows individuals residing inside crowded areas of the search space to provide more diverse solutions. Fitness scaling, on the other hand, adjusts the fitness values to make a balance in the selection process and avoid premature convergence.

To design an effective fitness function, a deep understanding of the problem and the optimization objectives is required. Therefore, a fitness function should always be created carefully to give a reliable and meaningful measure of the solution quality. With the help of

an efficient fitness function, GAs are capable of exploring the search space and converging towards an optimal solution more effectively.

Selection

In the selection phase of a GA, the aim is to select the fittest individuals which can lead to meeting the goals and sending their genes to the succeeding generations. A pair of individuals which are then called parents, are selected out with respect to high fitness scores.

The selection phase plays a very important role in GAs, as it determines which individuals should be moved on to the next phases for reproduction. The main goal of the selection phase is the promotion of survival and propagation of fitter individuals while keeping the diversity in the population maintained. The selection phase can be divided into the following parts to get a better understanding.

- Selection Methods: The researchers have come up with many selection methods to be used in GAs. Each of these methods varies each other according to the problem parameters, characteristics, and nature, therefore, all methods come up with some advantages and disadvantages of their own. Some of the most commonly used methods are listed below:
    i. Roulette Wheel Selection: The probability of selection for each individual is directly proportional to the fitness value it has. The reason for being called roulette wheel selection is the natural phenomenon of spinning a roulette wheel, where the probability of getting on a particular individual is found by its fitness value. This phenomenon can be easily explained with the help of the Figure 3.3:

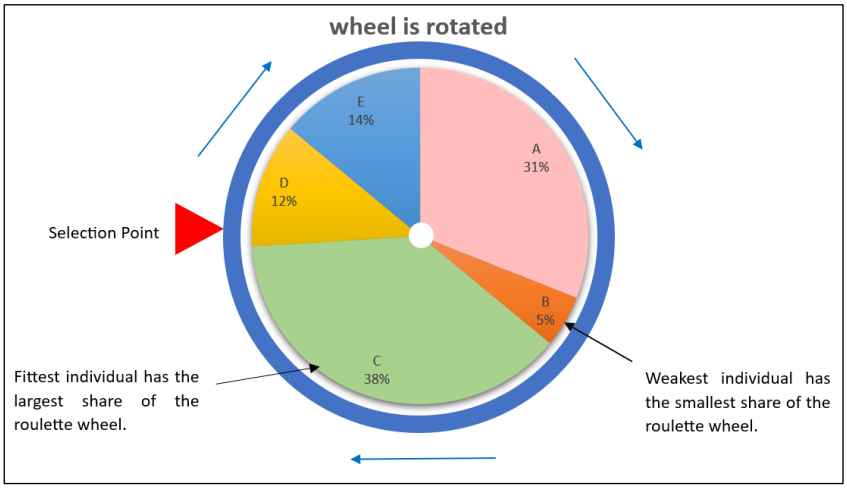

Figure 3.3. Roulette wheel selection

ii. Tournament Selection: In the tournament selection method, a random subset of individuals, which is called the tournament size, is selected and the individual with the highest fitness value is chosen as a parent for the next generations. This whole process is repeated over time to select multiple parents.

iii. Rank-Based Selection: In this method, the individuals are first sorted out on the basis of the fitness value they carry and then their ranks are used to give them a selection probability. The higher the rank, the more the probability of getting selected.

iv. Stochastic Universal Sampling: In the Stochastic Universal Sampling method, multiple individuals are selected simultaneously with the help of a selection pointer which moves in a roulette wheel and chooses individuals at evenly spaced intervals.

v. Boltzmann Selection: This method is based on the Boltzmann distribution where a temperature parameter is used to control the selection pressure. At the start, the temperature is high, which shows that the selection pressure is less. The temperature gets lowered with time, resulting in a gradual increase in pressure. This allows the GA to focus more on the best part of the search space while maintaining diversity in the search space.

- Selection Pressure: Selection pressure means the degree to which better individuals are given a favor for selection. Higher selection pressure means the convergence is faster but it can result in a reduction of diversity and it can lead to premature convergence to sub-optimal solutions. On the other hand, low selection pressure gives an increased opportunity for exploration, but it can make the convergence process slower. The selection method choice and the parameters can have a great impact on the selection pressure and therefore, the working of GA.
- Elitism: Elitism is a process of preserving the best individuals from one generation and then transferring it to the next one. Elitism makes sure that the best solutions reside in the solution, putting a stop to the loss of good solutions. Normally, a specific number of the fittest individuals are copied to the next generation without a need to

use genetic operators. The method proposed in this thesis also makes use of the elitism strategy.

- Parent Selection: After the selection probabilities are identified, parents are chosen on the basis of these probabilities. Selection can be made with replacement or without replacement. When selection is made with replacement, the same individual can be selected a multiple number of times which increases its chance of becoming a parent. The process of selection is repeated several times until a required number of parents are selected for the process of reproduction.
- Diversity Preservation: To preserve diversity in the population, it is very important to maintain a balance between selection pressure and exploration. Higher selection pressure can result in premature convergence and a loss of diversity, which then leads to a suboptimal solution where the population gets stuck. Methods like fitness sharing and crowding can be used to introduce more diversity in the population. In the method introduced in this thesis, the crowding distance method has been used to get a higher diversity.

The choice of the right method of selection is crucial when it comes to the working fashion of GAs. A selection method should always make sure that the fitter individuals get a higher chance of making a contribution of their genetic material to the upcoming generations. A careful selection of parents based on the fitness value they carry; genetic algorithms make their focus on the search process for the most promising regions of solution space and also preserve diversity for exploration.

Crossover

The process of crossover can also be called recombination, which is a fundamental genetic operator in GAs. Crossover stimulates the mating and recombination of the genetic material from the parents selected in the selection phase. The Crossover method includes the exchange of genetic information across parents to create new offspring that would have combined traits. This process helps in the exploration of search space and the potential production of better solutions. Here's a comprehensive explanation of a crossover procedure taking place in a genetic algorithm.

- Parent Selection: The process of crossover starts with a selection of parent individuals from the population which depends on the fitness values of the individuals. Usually, two parents are selected for a crossover procedure, however,

the number of parents can be different depending on the implementation of a GA and the requirements specific to the problem. The process of selection can be done by using different methods for example roulette wheel selection, rank-based selection, or tournament selection.

- Crossover Operators: There are many kinds of crossover operators, and all come with their way of combining of genetic information of the parents. Some of the most commonly used crossover operators are shown in Figure 3.4, while some are defined below:
    - One-Point Crossover: A random crossover point is selected in the chromosome, and the genetic materials after that specific point are interchanged between the parents in order to form a new offspring.
    - Two-Point Crossover: Two random crossover points are chosen in the chromosome, and the genetic information between these two points is exchanged between the selected parents.
    - Uniform Crossover: Each gene inside the chromosome has its independent probability of having an exchange between the parents.
    - Multi-Point Crossover: A multiple number of crossover points are selected to exchange the genetic material between these points in the parents.
    - Whole Arithmetic Crossover: The new offspring comes into being by taking a weighted average of the genetic information from the parents.
    - Heuristic Crossover: In this method, the crossover operators are based on a specific problem heuristics or knowledge.

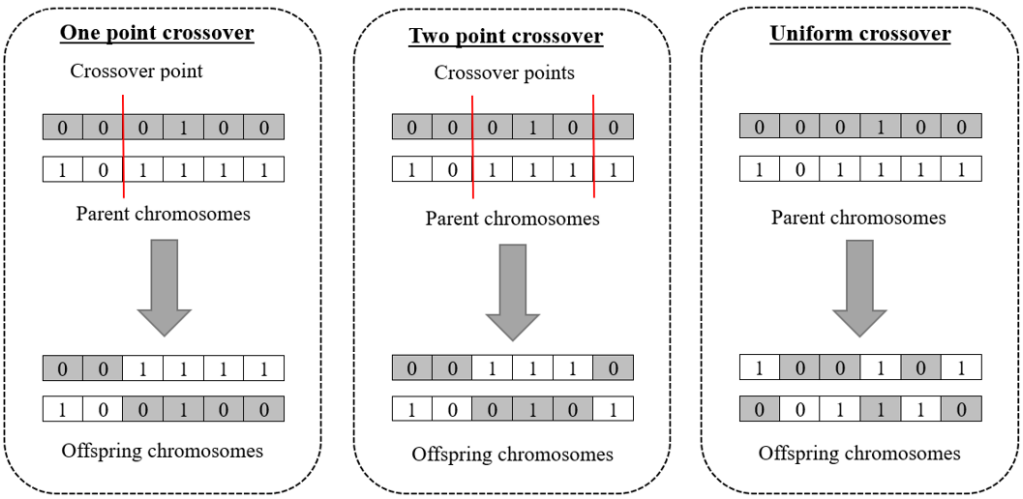

Figure 3.4. Common crossover methods

- Crossover Procedure: After the selection of parents and the suitable crossover operator, the process of crossover is taken into place to produce new offspring. The crossover is only applied to the selected regions of the chromosome which depends on the crossover operator and the parameters being selected. The genetic materials of two parents are exchanged or recombined to form new offspring having different genetic information.
- Crossover Rate: The crossover rate is a parameter that has to be defined to determine the probability of performing crossover over the selected parents. A high value of crossover rate results in greater chances of exploration which makes an increase in the likelihood of the genetic combination. This promotes diversity in the population. On the other hand, a lower value of the crossover rate reduces the process's exploration capability and conserves the parents' genetic material. This leads to a slower convergence but with a more significant potential for the maintenance of good solutions.

The process of crossover behaves as a key to the genetic operator when it comes to the performance of the genetic algorithms. This is because they promote the recombination and exchange of genetic information between superiorly selected parents. A combination of traits from several individual crossovers helps in exploration inside the solution space, introduces diversity in the population, and produces new offspring having improved characteristics. The crossover operator, rate, and related parameters must be chosen after a careful observation of the problem domain and genetic algorithm working. In the method introduced in this thesis, a recently introduced, cycle crossover method has been applied after a careful observation of the problem given.

Mutation

The process of mutation holds significant importance when it comes to the working of genetic algorithms that make random perturbations or changes in the genetic material of individuals. It introduces diversity to the population and aids in the exploration of new regions inside the solution space. Mutations play an important role in the prevention of premature convergence of solutions and improve the chances of getting the optimal or near-optimal solutions. The important concepts of mutation are explained below:

- Purpose of Mutation: Mutation is a mechanism that is used to maintain genetic diversity in the population by making random changes to the individuals. It allows

to make advanced exploration in the new and diverse regions inside the solution space which might not be reachable by crossover alone. Mutation also causes prevention of the population from converging to a suboptimal solution and also supports the promotion of disclosure of better optimal solutions.

- Mutation Operators: There have been many mutation operators discovered and applied successfully in genetic algorithms. All of these operators differ from each other in making changes to the genetic materials. Commonly used mutation methods are shown in Figure 3.5. Some of the most commonly used mutation operators being used can be listed below:

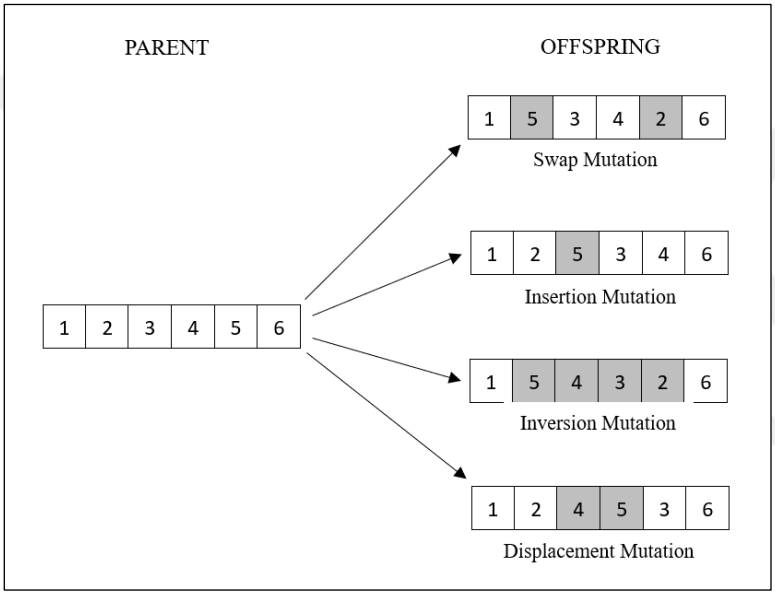

Figure 3.5. Common mutation methods

  - Bit-Flip Mutation: A mutation method that selects one or more bits of the chromosomes randomly flips the values that include the variation of bits, making 0s as 1s and 1s as 0s.
  - Swap Mutation: It randomly chooses two genes of the chromosome to swap their positions changing the order of their genetic information resulting in a new offspring (Figure 3.5).
  - Insertion mutation: A mutation method in which a new gene or a segment of genes is added to the genetic information of the chromosome at a random position in it (Figure 3.5).
  - Inversion Mutation: A mutation process in which a subset of genes is selected randomly from the chromosome and their order is reversed which rearranges the present genetic information of the chromosome (Figure 3.5).

  - Displacement Mutation: It is a method in which a segment of a chromosome is selected and is displaced to a new random position inside the same chromosome (Figure 3.5).
  - Gaussian Mutation: This method makes changes in the numeric values of genes by the addition of small random values taken from a Gaussian distribution method.
  - Uniform Mutation: In this method, random changes are made in numeric values of genes that reside in a specific range by the addition of a random value taken from a uniform distribution algorithm.
  - Heuristic Mutation: Mutation operators that are based on the nature of the problem in hand, having specific knowledge or heuristics.
- Mutation Rate: Mutation rate represents the parameter that decides the probability on which the rate of mutation is applied to the genetic material of an individual. A high rate of mutation increases the chances of random changes and eventually promotes exploration and more diversity inside the population. Whereas, a lower rate of mutation helps in the reduction of the probability of mutation and preserves the genetic material of the individuals. This leads to faster convergence but there is a risk of being trapped inside a local-optima.
- Offspring Generation: After the application of the selected mutation operation on a specific mutation rate on the genetic material, the new individual is called an offspring. The new offspring with diverse genetic information is added to the population to take part in the next generations.

The mutation process behaves as a vital component when it comes to the working of genetic algorithms. It not only produces random changes in the genetic material of individuals but also maintains diversity and explores new spaces in the solution. By combination of mutation with the other genetic operators for example crossover, genetic algorithms can be used to get optimal or near-optimal solutions for complex optimization problems.

3.3. Multi-Objective Optimization

Multi-objective optimization (MOO) refers to a field within optimization where the goal is to simultaneously optimize two or more conflicting objectives. In contrast to single-objective optimization, MOO seeks to find a set of solutions that represents a compromise among multiple, often conflicting, objectives. This field has found applications in various domains,

including engineering, finance, logistics, and artificial intelligence. Suppose we have n distinct objective functions that need to be maximized, it can be expressed mathematically as in Equation 3.1.

$$\max(f_1(x)), \max(f_2(c)), \dots, \max(f_n(x)) \quad (3.1)$$

Where x represents the input solution and $(f_1(x), f_2(x), \dots, f_n(x))$ are different objective functions that have to be optimized. A solution S_1 is considered in a dominant position over another solution say S_2 if the values obtained by objective functions of the solution S_1 are superior to the values obtained by the solution S_2.

Key concepts of MOO:

Objective Functions: In MOO, each objective is associated with a function that quantifies a particular aspect of the system being optimized. Objectives are often in conflict; improving one may deteriorate another.

Pareto Dominance: A solution is considered better than another if it is at least as good in all objectives and better in at least one. Solutions that are not dominated by any other solutions are part of the Pareto-optimal set.

Pareto Front: The set of all Pareto-optimal solutions forms the Pareto front. The ideal case is to have a diverse and well-distributed Pareto front, offering a range of trade-offs.

Multiple studies have demonstrated the efficacy of multi-objective optimization compared to bi-objective or single-objective optimization approaches. For example, a comprehensive study (Hasanoglu & Dolen, 2020) shows that multi-objective optimization outperforms other methods in achieving results without the need for weighting the objective functions beforehand. The multi-objective approach can provide results for multiple single-objective functions in a single iteration, making it computationally more efficient. Additionally, multi-objective community detection yields more robust outcomes as each objective function uncovers a distinct aspect of the community structure.

During the conducted research, it was observed that most community detection methods employ either single-objective optimization or bi-objective optimization. Single-objective optimization frameworks generally use Modularity. However, the quality of the community structure may be compromised when two objective functions are employed, potentially leading to solutions with high community density and low inter-linkage.

For example, in a network N, a solution consisting of a single community comprising all nodes in the network might have maximum intra-linkage and no inter-linkage, but this may not represent the optimal structure. This is because such a solution would result in a Modularity value of 0, which doesn't align with the optimization goals of the problem.

To address these issues, this paper proposes a many-objective optimization approach that utilizes three objective functions to optimize the solution. The objective functions being used are described below:

3.3.1. Temporal smoothness

There is no single optimal solution in multi-objective optimization that can be considered as the final result. Instead, a set of solutions that are deemed to be superior to others is obtained. To achieve this, the problem is transformed into a desired outcome using Equation 3.2 (Chi et al., 2009):

$$cost = \alpha SC + (1 - a)\, TC \tag{3.2}$$

In this approach, the Snapshot Cost (SC) is utilized to quantify the assessments of the obtained community structures. The method proposed in this paper uses Modularity to calculate the SC. A solution with a higher Modularity and a lower SC is considered to have a better community structure. The Temporal Cost (TC) quantifies the similarities existing between the community structures. The proposed method employs the Normalized Mutual Information (NMI) in order to calculate TC. The higher the NMI, the lower the TC and the more similar the obtained communities are. To strike a balance between SC and TC, a parameter alpha is used, which ranges from [0,1]. The community detection algorithm finds an optimal community using the described algorithm. However, the uncertainty in the value of alpha can be a limiting factor as it needs to be specified as an input to maintain a balance between the SC and TC functions, which can be a challenging task.

3.3.2. Modularity

Modularity, also known as the Q-function, is a widely utilized evaluation criterion in network community detection research. It measures the discrepancy between the actual and predicted number of edges inside a group, providing a quantitative assessment of the strength of community structure. Modularity score represents a fraction of edges within a group compared to the edge number that would occur if the edges in the network were distributed

in a random fashion. This metric is popular among researchers due to its simplicity and effectiveness in identifying significant community structures. It was first introduced in a seminal paper by Newman and Girvan (Girvan & Newman, 2002) in 2004. The use of modularity in network analysis has since been further developed and extended to address various challenges, such as resolution limit and the detection of overlapping communities. Despite its limitations, modularity remains a widely used measure for network community detection and continues to be an important tool for understanding complex network structures. Mathematically, Modularity can be written as in Equation 3.3.

$$Q(CR^t) = \sum_{i=1}^{CR^t} \left[\frac{l_i}{E} - \left(\frac{d_i}{2E} \right)^2 \right] \tag{3.3}$$

Here, $|E|$ represents the edges present at N^t, the number of edges in the i-th community is referred to as l_i and lastly the degree d_i represents the degree of i-th community. The modularity metric falls within a range of [-1/2, 1], in which, the greater Modularity score shows that a better community is formed.

3.3.3. Normalized mutual information

Normalized Mutual Information (NMI) is a well-established metric that measures the similarity between two community partitions. NMI was first introduced in the field of computer science as a technique used to assess the effectiveness of community detection algorithms (Danon et al., 2005). NMI is a widely used metric in community detection research because of its ability to provide a normalized measure of similarity comparison of the communities generated by an algorithm to the actual, ground-truth community structures. NMI can also provide a measure of the stability of community structures obtained from different algorithms. Let us say, $CR^{t-1} = C_1^{t-1}, C_2^{t-1}, \dots, C_l^{t-1}$ and $CR^t = C_1^t, C_2^t, \dots, C_k^t$ are two communities formed at time t, then $NMI(CR^t, CR^{t-1})$ used to get the similarity between these two communities is defined mathematically in Equation 3.4.

$$NMI(CR^t, CR^{t-1}) = \frac{-2 \sum_{i=1}^{m} \sum_{j=1}^{n} C_{ij} \log\left(\frac{C_{ij} N}{C_i C_j} \right)}{\sum_{i=1}^{m} C_i \log\left(\frac{C_i}{N} \right) + \sum_{j=1}^{n} C_j \log\left(\frac{C_j}{N} \right)} \tag{3.4}$$

Here, C represents a confusion matrix, C_{ij} shows the number of nodes in $C_i^t \in CR^t$ and $C_j^{t-1} \in CR^{t-1}$. The sum of all elements in the ith row of C is written as C_i which is the total nodes C_i^t and CR^{t-1} have in common, in other words, the nodes present inside the

community C_i^t. C_j shows the nodes present inside the community C_j^{t-1} which is calculated by summing up all the elements located in column j of the community C. The total number of nodes is referred to as N. In the expression, the symbols m and n describe the number of overall communities present in CR^t and CR^{t-1} communities respectively. The value of NMI is between [0,1]. The closer the value is to 1, the more similar compared communities are to each other.

3.3.4. Normalized cut

Normalized Cut (NC) (Dhillon et al., 2004) is a measure used to assess the quality of communities obtained through a partitioning process. In mathematical terms, let us consider $C^t = C_1, C_2, \dots, C_k$ to be a set of k communities discovered at time t. The NC function is employed to determine the similarity between communities by calculating the proportion of the total edge weight connecting different communities to the sum of the internal edge weights within these communities. NC is a widely used evaluation criterion in various community detection algorithms and has been proven to provide reliable results in many cases. The community structure quality at a specific time t can be determined by using normalized cut written as given in Equation 3.5.

$$NC(C^t) = \frac{1}{k} \sum_{i=1}^{k} \frac{cut\ (C_i, \bar{C}_i)}{vol\ (C_i)} \tag{3.5}$$

Here, $cut\ (C_i, \bar{C}_i)$ shows the edges existing among the two communities C_i. $\bar{C}_i = \frac{G}{C_i}, vol\ (C_i)$ represents the vertices present in the community C_i and finally the quantity of communities is represented by the letter k.

3.4. Non-dominated Sorting Genetic Algorithm-II

The Non-dominated Sorting Genetic Algorithm-II (NSGA-II) (Deb et al., 2002) is a multi-objective optimization algorithm that is highly efficient in detecting communities. This algorithm employs a genetic approach to locate a set of Pareto solutions, each solution representing a different community partitioning. The decision maker then selects the best Pareto solution based on the desired criteria. One of the notable features of NSGA-II is its ability to consider only the local information of the network vertices during the crossover and mutation operations. This makes NSGA-II well-suited for large and complex networks where global information might not always be accessible. The three main problems being solved in the NSGA-II algorithm are as below:

3.4.1. High computational complexity

The High Computational Complexity is a significant challenge faced in the NSGA algorithm. NSGA algorithm has a computational complexity of $O(MN^3)$, where M represents the number of objectives that need to be optimized, and N refers to the population size. This computational complexity can make the NSGA algorithm highly costly in terms of computation, hence the need for a solution. Fortunately, this issue has been addressed in the NSGA-II algorithm. The computational complexity of the NSGA-II algorithm has been reduced to $O(MN^2)$, making it a more efficient algorithm compared to its predecessor, the NSGA. This reduction in computational complexity has made the NSGA-II algorithm a more practical solution for community detection in large-scale networks.

3.4.2. Non-elitist approach

In NSGA and many other GAs, non-elitist approach has been followed. It is found by many researchers (Groşan et al., 2003; Ishibuchi et al., 2008; Purshouse & Fleming, 2002) that an elitist approach is much better in performance than a non-elitist approach in Genetic algorithms. Elitism is done by making a comparison between the current population and the best non-dominated solution found in a previous iteration. This approach helps in preserving the better solutions being found over different iterations.

3.4.3. The sharing parameter - σ_{share}

Previous algorithms utilized a sharing parameter, denoted σ_{share} to generate a diverse population in each iteration. This parameter was used to specify the intensity of sharing in the sharing function and represented the maximum distance between two solutions that shared fitness values. However, the setting of this parameter was dependent on the user's knowledge and expertise, which could lead to errors. Additionally, the sharing function had a high computational cost of $O(MN^2)$, as it required comparison of each solution with previously obtained solutions. To address these issues, NSGA-II has replaced the sharing function with a crowded function approach, which effectively overcomes the problems associated with the sharing function.

It has been demonstrated through empirical analysis of time and computational complexity, as well as community partition results, that NSGA-II outperforms all other algorithms in both synthetic and real-world network datasets.

3.4.4. Pareto-rank based evaluation

Pareto-rank based evaluation is a multi-objective optimization approach used to evaluate and compare multiple solutions to a problem simultaneously. Pareto rank method was first proposed by Goldberg (Goldberg, 1989). The concept of Pareto-rank is rooted in Pareto efficiency, which states that a solution is considered efficient if no other solution exists that has the ability to enhance one objective while not causing degradation in another. In Pareto-rank based evaluation, solutions are ordered according to their Pareto dominance, which refers to the degree to which a solution dominates other solutions within an objective space.

The process of Pareto-rank based evaluation begins with the identification of a set of objectives to be optimized. Next, multiple solutions to the problem are generated and evaluated according to each objective. These solutions are then compared to determine which solutions are non-dominated, meaning they cannot be improved in one objective without worsening another. These non-dominated solutions are considered as a part of Pareto front, representing a collection of optimal solutions for the problem. Pareto-rank based evaluation provides a powerful framework for evaluating and comparing multiple solutions to a problem by considering trade-offs between multiple objectives. It offers a rigorous and comprehensive methodology for multi-objective optimization and is widely used in a variety of fields.

Pareto ranking is a crucial component of the NSGA-II algorithm. It enables equal opportunities for reproduction within the group of non-dominated individuals in a given population. The Pareto-based method divides solutions into multiple ranks, with the individuals having the highest fitness values in the population assigned to Rank 1. These individuals are non-dominated by all others and are removed from the population. The process then repeats to find the next non-dominated individuals, with those found in the second iteration assigned to Rank 2 with the second-best fitness values.

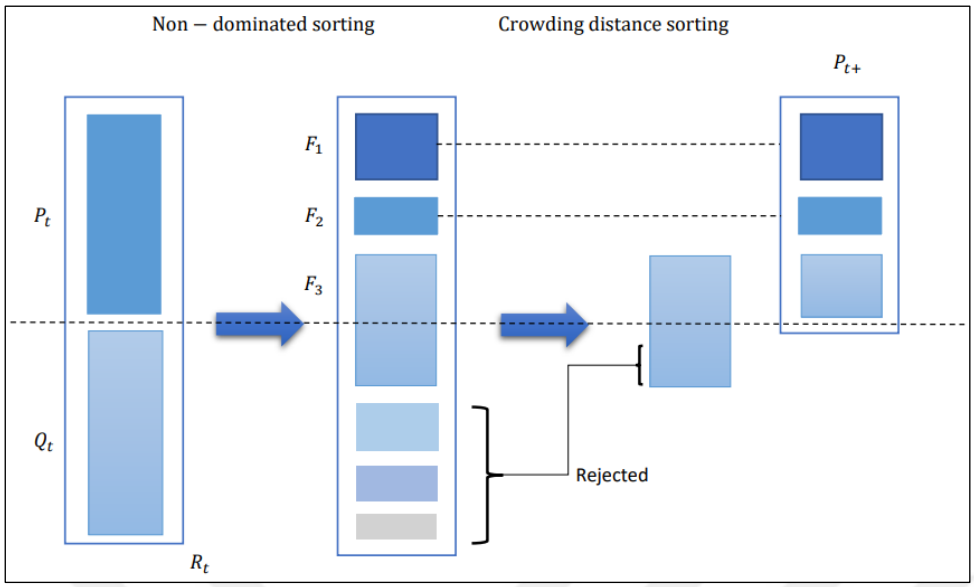

Figure 3.6. Systematic diagram of NSGA-II sorting mechanism

This procedure persists until every individual in the population has received a rank. To address the challenge of maintaining multiple optimal solutions, fitness-sharing techniques are often employed. This method calculates the distance between a specific individual and all other individuals in the same rank. In NSGA-II, the technique of crowding distance solves the fitness-sharing problem for individuals, which is discussed in depth in section 4.3 of the document. A systematic diagram of how Pareto ranking and crowding distance sorting is performed in NSGA-II is presented in Figure 3.6.

After identifying the Pareto solutions, the subsequent step is to choose the optimal solution. To achieve this, a Community Score with an order 2 has been utilized in the proposed method. Community Score (CS) determines the quality of the obtained community structures which can be written mathematically as in Equation 3.6 (Zou et al., 2021):

$$CS(CR^t) = \sum_{i=1}^{k} \left[\frac{\sum_{m \in C_i^t} (\mu_m)^2}{|C_i^t|} \; x \; \sum_{m,n \epsilon C_i^t} A_{mn} \right] \tag{3.6}$$

Here, μ_m represents the expression $\frac{1}{|C_i^t|} \sum_{n \in C_i^t} A_{mn}$. In this expression, $|C_i^t|$ is the existing total nodes inside a community i at a given time t and the adjacency matrix is being represented by A. The higher value of CS represents a good community partition, this is the reason for selecting a solution having the highest CS value to be the best possible solution out of all other Pareto solutions.

3.5. CD-INSGA Methodology

The methodology proposed in this thesis work is called the Transfer-based Multi-objective NSGA-II algorithm. This methodology incorporates the elite properties of the NSGA-II algorithm, along with an additional transfer-based technique inspired by the TMOGA algorithm (Zou et al., 2021). This technique transfers useful information found in previous generations and passes it on to the next generations to enhance the performance and time complexity of the NSGA-II algorithm. In CD-INSGA (Community Detection using an Improved NSGA) algorithm, the optimization of the multi-objective problem is performed using evolutionary algorithms, which are based on inspiration from natural processes. The procedure involves selecting individuals to be included in the next population and generating new populations through crossover and mutation processes. This cycle is repeated until the specified fitness functions (Modularity, NMI, and Normalized cut) are optimized. The improvement in the crossover function and the modification of the crowding distance function in CD-INSGA will be discussed in subsequent sections. The overall process being followed in CD-INSGA is as described below.

3.5.1. Feature extraction

Feature extraction is a crucial step in network studies, involving the identification and extraction of important features or characteristics from a network. These features represent the network in a compact, informative manner, making it easier to analyze, compare, and understand the network's properties. Feature extraction is pivotal for network analysis, enabling network comparison and evaluation, and supporting the development of new network models and algorithms. Various methods for feature extraction in network studies include topological, spectral, and statistical approaches.

Topological methods extract features based on the network structure, such as the number of nodes, edges, degrees, triangles, and shortest paths. Spectral methods extract features from the eigenvalues and eigenvectors of the network's adjacency matrix, such as eigen centrality and eigenvector centrality. Statistical methods extract features based on the distribution of network properties, including centrality measures, clustering coefficient, and degree distribution.

Regardless of the method used, feature extraction is essential in network studies, providing valuable information about the network structure, properties, and relationships between

nodes. Choosing the right feature extraction method is crucial according to the goals and objectives of the network analysis, and to validate the extracted features for reliability.

The feature extraction method reduces the dimensionality of the analysis and prediction processes, facilitating an efficient learning process with minimal computational complexity and the development of generalized models that require reduced storage capacity (Arowolo et al., 2017; Rostami et al., 2021).

In the community detection problem, communities are subdivided into smaller groups known as cliques or features. The division of a community into cliques is determined by the Community Internal Density (CID) threshold. When the CID threshold is set to 0, each community is considered as a tiny clique. Conversely, when the threshold is set to 1, only sub-graphs that are fully connected within the community are considered as features.

The best solution is selected in an iteration, and its small cliques (features) are extracted. In Figure 3.7, if we consider it as a community, then the cliques formed inside the community are (A, B, C, D), (E, F, G, H), and (I, J, K). To reduce the time consumption and computational difficulty of feature extraction, TMOGA utilizes orderly search, greedy pruning, and depth limit methods (Zou et al., 2021). Greedy pruning reduces complexity by halting the search in a sub-tree if it has a CID less than the set threshold, avoiding unnecessary search processes. Orderly search is employed because the nodes in sub-graphs are unordered, allowing the search to continue only if the neighbor of a node has an index greater than the current node. This reduces the number of subtrees at each level in a tree and their depth by approximately 50%.

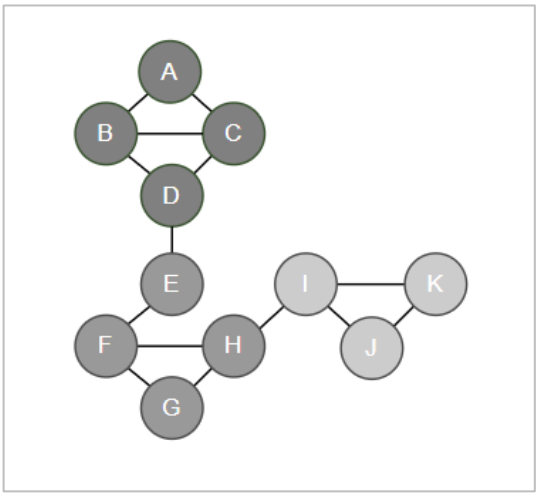

Figure 3.7. Three cliques connected by an internal link

To prevent overfitting, the authors of TMOGA implemented a depth limit approach to restrict the size of the transferred cliques. This method establishes a maximum depth (Md) threshold to limit the size of transferred cliques, with cliques larger than the threshold not being transferred.

The feature extraction process begins by selecting an initial node as the starting point for the expansion of the search tree. The search tree continues to expand as long as the index of the neighbor node is greater than that of the current node. The overall depth of the search tree should not exceed the predetermined maximum depth (Md) threshold, and the cliques discovered must have a Community Internal Density (CID) value that exceeds the set threshold. At the conclusion of this process, all small cliques or features will have been extracted.

3.5.2. Feature migration

Feature Migration is a process in network studies that involves transferring features extracted from one network to another. It is a critical technique in the area of network analysis that helps reduce the computational time and complexity associated with extracting features from each network individually. The process of Feature Migration can be performed in various ways, including greedy pruning, orderly search, and depth limit methods. These methods ensure that the features transferred are relevant and appropriate for the new network.

The Feature Migration process starts with the selection of an initial network, from which the features are extracted. The extracted features are then transferred to the next network, where they serve as a starting point for further feature extraction. This process continues until all the networks in the set have been analyzed. The Feature Migration technique is useful in situations where the structure of the network undergoes changes over time, and it is necessary to maintain a consistent set of features for comparison purposes.

One of the challenges in Feature Migration is the potential for overfitting, where the transferred features may not be appropriate for the new network. To address this issue, methods such as the depth limit approach can be used to limit the size of the features that are transferred. This helps prevent the transfer of features that may not be relevant to the new network and ensures that the transferred features are suitable and appropriate for analysis.

After clique extraction, the next step is to transfer them to the subsequent population. To maintain diversity in the new population, a variable named Transfer Probability (T_p) is

introduced (Zou et al., 2021). In the CD-INSGA process, the value of (T_p) is set at 0.5. During the feature migration process, each feature's community structure is assigned to the initial solution found in each population according to the value of (T_p).

3.5.3. Initialization

In CD-INSGA, the Label Propagation Algorithm (LPA) (Raghavan et al., 2007) is used to generate the initial population, which provides efficient and top-notch solutions very quickly with minimal computational complexity. The node labeling process, in which nodes are assigned labels from the small cliques extracted in the feature extraction process, is carried out during the initialization stage. This process is repeated over subsequent iterations and is necessary for the feature transfer process.

3.5.4. Selection

Selection is the stage of choosing specific individuals from a population to pass them onto the crossover and mutation processes for the generation of a new population. In the NSGA-II algorithm (Deb et al., 2002), the tournament method is used in the selection process. In tournament selection (TOS) (Hingee & Hutter, 2008), a k number of individuals are randomly selected. After selection, they are ranked in accordance with their relative fitness, and finally, the fittest individuals are selected to participate in the reproduction steps. This whole process is applied n times for the whole population. The tournament method comes with the drawback of never allowing the individuals having low fitness to participate in crossover and mutation processes thus putting a strong limitation. It is believed that in some cases, individuals with low fitness can also have useful information that can help to obtain a better population, therefore, to eliminate this shortcoming, the inverse Pareto rank method has been used CD-INSGA to assign weight to each individual, and then making a random selection in accordance to the weights (Zou et al., 2021). The inverse Pareto rank method addresses this limitation by assigning a weight to each individual based on their relative Pareto rank. The Pareto rank is a measure of the non-domination of an individual solution compared to other solutions in the population. The inverse Pareto rank method assigns a higher weight to individuals with lower Pareto ranks, allowing them to participate in the reproduction processes with a higher probability. This helps to ensure diversity in new populations in order to prevent premature convergence to sub-optimal solutions. In summary, the inverse Pareto rank method is a modification of the traditional tournament

selection method that takes into account the Pareto ranks of individuals to assign weights and make a random selection for the reproduction processes in multi-objective optimization problems.

3.5.5. Crossover

Crossover is a crucial step in evolutionary algorithms where a new population is created through the combination of two or more parent populations. It is considered to be the main search operator in a genetic algorithm. The individuals in the parent populations are merged in a specific manner to produce a new set of individuals, forming child populations. It is essential to design an efficient crossover method that allows only the superior and diverse individuals from the parent populations to be passed on to the offspring populations (Tahmasebi et al., 2019). In NSGA-II (Deb et al., 2002), a simulated binary crossover (SBX) (Deb & Agrawal, 1995) operator is used for this purpose. A binary vector of equal length to the parents is generated randomly, and new populations are generated in iterations equal to the magnitude of the parent population. In case the binary vector value equals 1, the value of the individual from the first parent is taken, and if the value is 0, the individual from the second parent is taken to create a fresh offspring population. The disadvantage of the SBX crossover is that if the parent populations are nearly or completely identical, the child population will also be the same, reducing the chances of obtaining a diverse child population. To overcome this, the Cycle Crossover (CX) strategy (Puljić & Manger, 2013) is used in the proposed CD-INSGA method, which is implemented in NSGA-II for the first time. The pseudo-code algorithm of cycle crossover is provided in Table 3.1.

Table 3.1. Cycle Crossover Algorithm

Cycle Crossover Algorithm

Require: P_a, P_b > P_a, P_b = Parent populations
    $a \leftarrow [\,]$ > initialization of child of P_a
    $b \leftarrow [\,]$ > initialization of child of P_b
    Set $a[0] = P_a[0]$
    if $a[0]$ in P_b and $b[0]$ in P_a **then**
        Find index of $a[0]$ in P_b and $b[0]$ in P_a
        if index of a > index of b **then**
            loop.index = index of b
        else if index of a < index of b **then**
            loop.index = index of b
        else
            loop.index = index of a
        end if
        if index of $a[0] == 1$ **then**
            Insert value of $a[0]$ into position $b[1]$
            for i in range (0, loop.index) **do**
                insert remaining values of P_b in corresponding position in b.
            end for
        else
            Set b[index of a] = a[0]
            for i in range (0, loop.index) **do**
                insert remaining values of P_b in corresponding position in b.
            end for
        end if
        if index of $b[0] == 1$ then
            Set a[1] = b[0]
            for j in range (0, loop.index+1) **do**
                insert remaining values of P_a in corresponding position in a.
            end for
        end if
        for z in range $(loop.index + 1\,, length(a) - 1)$ **do**
            Insert remaining values of P_b in a.
            Insert remaining values of P_a in b.
        end for
    end if
    Return a, b

The entire procedure can be described in the following stages:

1. The crossover cycle begins by selecting the initial gene of the first parent and the first gene of the second parent.
2. The gene in the first position of the second parent is identified and moved to the corresponding gene in the first parent.

3. A vertical movement is made from the first parent's current gene to the second parent's gene.
4. If the gene of the second parent is identical to the initial gene of the first parent, step 6 is performed; otherwise, step 5 is executed.
5. A movement is made to the first parent's gene in accordance with the second parent's gene, then step 3 is repeated.
6. This entire process is repeated to generate the second offspring.
7. During the cycle, the genes from the first parent are duplicated in their corresponding positions in the first offspring, as demonstrated in Figure 3.9.
8. The genes from the second parent present in the cycle are then copied to their respective locations in the second offspring, as shown in Figure 3.9.
9. The genes left in the second parent are replicated in their corresponding positions in the first offspring, as demonstrated in Figure 3.10.
10. Finally, genes left in the first parent are replicated in their corresponding positions to the second offspring, as shown in Figure 3.10.
11. The final offspring populations are obtained by the sequence of genes generated in this process.

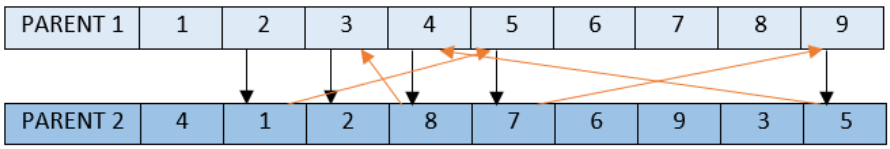

Figure 3.8. Schematic diagram of cycle formation.

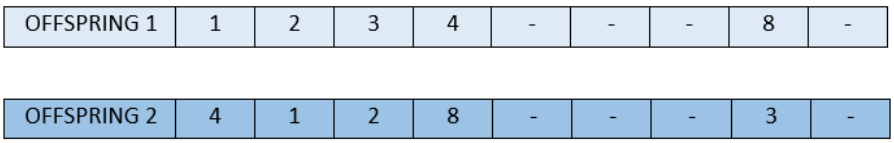

Figure 3.9. Schematic representation of the first stage of crossover cycle.

OFFSPRING 1	1	2	3	4	7	6	9	8	5

OFFSPRING 2	4	1	2	8	5	6	7	3	9

Figure 3.10. Schematic representation of offspring after crossover cycle completion.

3.5.6. Mutation

Mutation is a crucial component of genetic algorithms, serving to introduce diversity into the generation of new populations. The process of mutation is guided by the Mutation Probability (Mp), which must be kept low to prevent the genetic algorithm from deviating from its intended purpose and becoming a random search algorithm.

One commonly used method for performing mutation is called polynomial mutation. This approach involves randomly altering the labels of nodes in the offspring solution after each generation of offspring populations. The mutation process occurs by assigning a specific probability to each node, indicating the likelihood of changing its label to that of one of its neighboring nodes. This method helps sustain the genetic diversity of the population while also promoting the evolution of new and potentially better solutions. Therefore, the polynomial mutation has been applied in the proposed algorithm.

3.5.7. Crowding distance elimination (CDE)

The Crowding Distance function is a commonly used tool in genetic algorithms to maintain diversity within a population and determine the density present in non-dominated solutions. This function approximates the relative density of the non-dominated solutions surrounding a specific solution.

To evaluate the crowding distance, the set of solutions is arranged in ascending order according to their objective function values, and the average distance between two neighboring solutions is calculated. Solutions located at the boundaries having the highest and lowest values of the objective function are given an infinite value for their crowding distances, as shown in Figure 3.11. To calculate the total crowding distance value, the aforementioned procedure is applied to all objective functions, and the resulting crowding distance values are then summed.

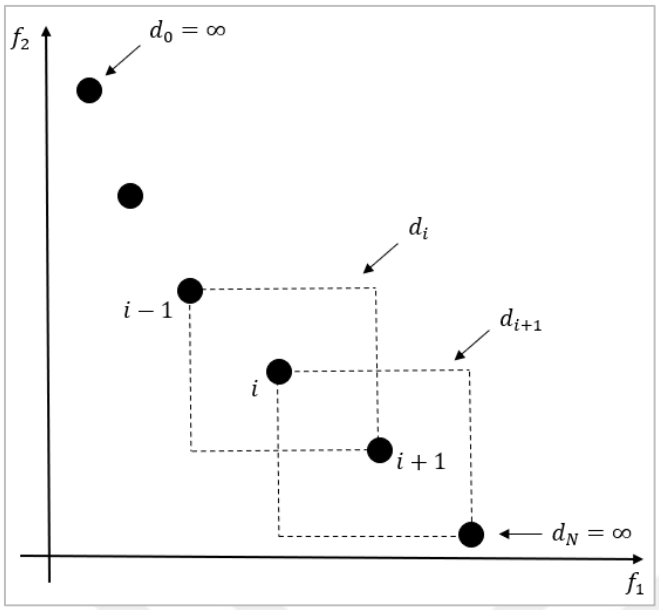

Figure 3.11. Graphical representation of Crowding Distance mechanism.

However, this traditional approach to the crowding distance function has a flaw in that it does not consider the effect of eliminating dense solutions on the neighborhood solutions (Liu & Chen, 2019; Tahmasebi et al., 2019). To address this issue, an improved method called the Crowding Distance Elimination (CDE) (Liu & Chen, 2019) has been implemented in the CD-INSGA algorithm. This method involves first calculating the crowding distance for solutions in the same Pareto front level, and then deleting the solution having the lowest value of crowding distance. The aforementioned procedure is reiterated until the desired count of non-dominated solutions is attained. A pseudo-code algorithm for the crowded distance is provided in Table 3.2.

Table 3.2. Crowding Distance Elimination Algorithm

Crowding Distance Elimination Algorithm	
Require: m, P	> m = Objective Functions, P = Pareto front
for $i = 1$ to $length(P)$ **do**	
$P[i].distance \leftarrow 0$	> initialization
end for	
for each objective k **do**	
$P = sort(P, k)$	> sort by objective k
$P[0] = \infty$	
$P[-1] = \infty$	> to always select the boundary points
end for	
for $j = 2$ to $length(F) - 1$ **do**	> find crowding distance
$P[i].distance = \frac{(P[j+1].value - P[j-1].value)}{interval}$	
end for	
$k \leftarrow MinimumDistance(P)$	> finding the solution having minimum crowding distance
$P[k] \leftarrow []$	> deleting the solution with the least value of crowding distance
return P.distance	

The CD-INSGA algorithm operates iteratively across several generations. Figure 3.12 illustrates the complete structure of the algorithm as a flowchart. At the start of the first generation (t=1), an initial population is generated through the Label Propagation algorithm.

In each subsequent iteration (k=1 to n, where n is a user-defined input), the algorithm calculates Modularity, NMI, and Normalized Cut for the current population to identify the Pareto optimal solutions. The Pareto ranks of individuals in the population are then determined. The Crowding Distance Elimination (CDE) method is employed to identify the best solution among those with identical Pareto rank. The CDE method calculates the overall crowding distance value by summing the individual values of the crowding distance for all objective functions. The CDE process involves computing the crowding distance for the solutions at the same Pareto front level and then eliminating the solution with the lowest crowding distance value.

This process is repeated for the remaining solutions until the number of non-dominated solutions reaches a specific quantity. A solution with the highest community score is selected from the Pareto solutions.

At the start of the next iteration (t=2), useful features from the solutions obtained in the previous iteration are extracted. This is followed by a feature migration step, where the extracted features are migrated to the population of the subsequent generation. The entire process, excluding the initial population generation, is repeated for n number of generations, producing new populations and selecting the most suitable individuals each time.

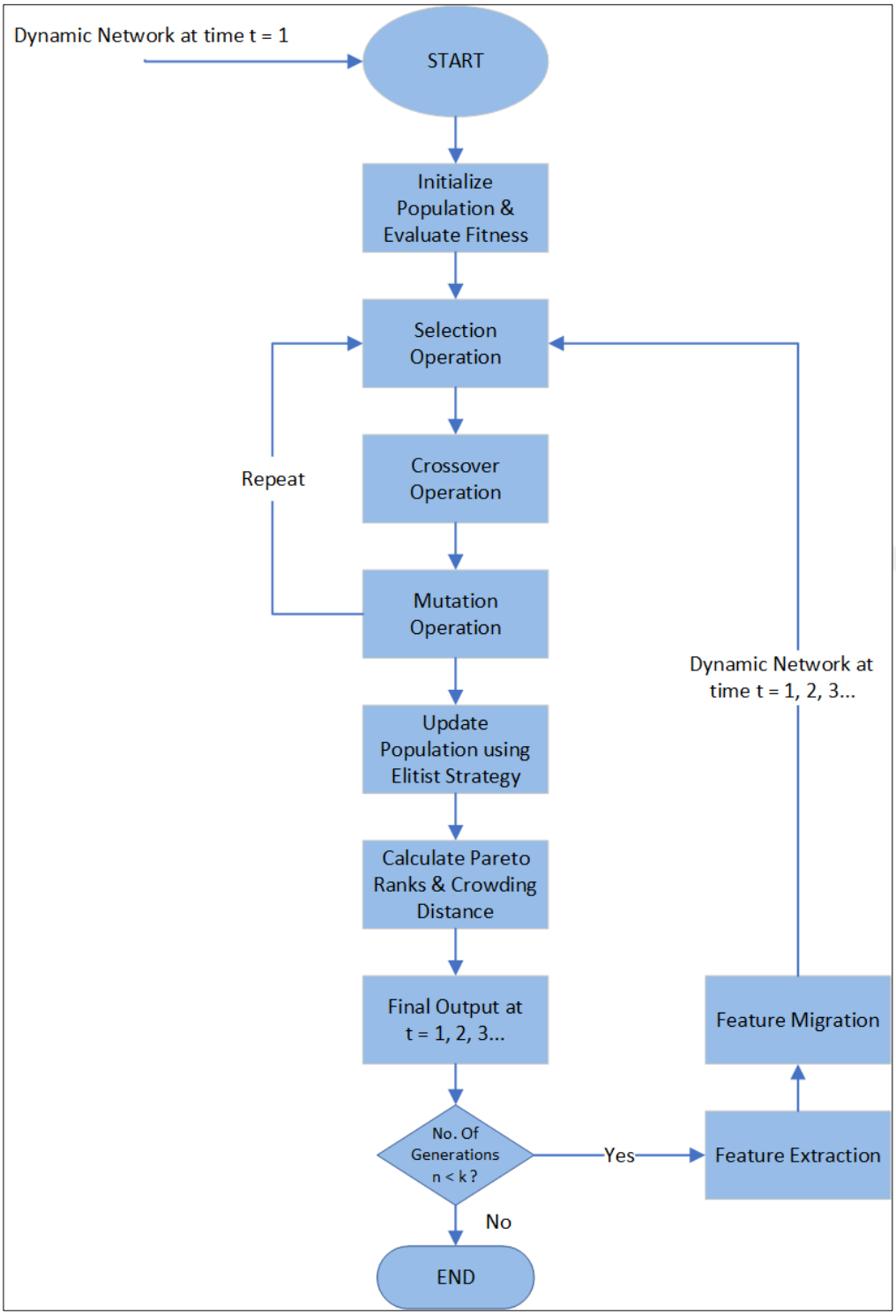

Figure 3.12. Flowchart of CD-INSGA

4. RESULTS AND DISCUSSION

This section presents the practical experiments and outcomes of the proposed CD-INSGA algorithm. The algorithm is compared with several existing algorithms named TMOGA (Zou et al., 2021), FacetNet (Lin et al., 2008), DYNMOGA (Folino & Pizzuti, 2013), and DYNMODPSO (Yin et al., 2021) to evaluate its performance. TMOGA uses the feature transfer algorithm to incorporate useful communities found in previous solutions into the current population. FacetNet is an efficient algorithm that views communities as hidden nodes. DYNMOGA is based on NSGA-II and focuses on finding communities present inside dynamic networks. DYNMODPSO, on the other hand, is a heuristic algorithm that uses the temporal smoothness framework. The Normalized Mutual Information (NMI) is set as a criterion to analyze the results and determine the similarity of the community partitions. The value of NMI varies from 0 to 1, with a closer value to 1 indicating higher similarity between the compared communities.

The implementation of CD-INSGA is done in Python language with version 3.6. The experiments were carried out on a Core i7 computer, a 10th generation processor, and 8GB of RAM.

4.1. Parameter Setting

The CD-INSGA algorithm requires the setting of seven parameters before its execution. These parameters are population size, the number of generations, Community Internal Density (CID) threshold, Crossover probability (Cp), Mutation probability (Mp), Transfer probability (Tp), and the maximum depth of the search tree (Md). To ensure a fair comparison with other algorithms, the population size and the number of generations have been designated as 200 and 100, respectively. The CID threshold is a crucial parameter that determines the formation of small cliques within a community. A clique refers to a group of nodes inside a community having a clustering effect greater than the CID threshold. If the CID threshold has a value of 0, the entire sub-graph is represented as a feature. Once the CID threshold attains a value of 1, only the well-connected sub-graphs are used to represent features. The CID threshold is set at 0.8 in the experiments. The max depth (Md) parameter is set to put a limit on the size of cliques. Setting an appropriate value for (Md) can help reduce the search space, resulting in minimizing the search complexity. The (Md) parameter has been set to 5. The features are transferred to the next generation only based on a

predetermined probability referred to as the Transfer Probability (Tp). This parameter helps in maintaining the diversity of the population. A very high value of (Tp) can result in a reduction of diversity in the population; therefore, it has been set to 0.5. For the generation of a new population in the genetic algorithm, the values for (Cp) and (Mp) are fixed at 0.8 and 0.2, respectively.

4.2. Datasets

To carry out comprehensive tests, synthetic datasets, and large-scale complex datasets are utilized. The description of the datasets is discussed below:

4.2.1. Synthetic dataset

The study utilizes two synthetic datasets, namely SYN-FIX and SYN-VAR, as put forward by Newman and Girvan (Girvan & Newman, 2002). A parameter named "Z" is employed to regulate the count of edges linked to each community. In the following sections, a brief overview of both datasets is provided.

- SYN-FIX: One of the defining characteristics of the SYN-FIX dataset is that it has a fixed count of edges that connect each community. This means that irrespective of the community size, the number of edges connecting each community remains constant. This makes it an ideal dataset for studying community structures where the number of connections between communities is fixed and unchanging. There are a total of 128 nodes in the SYN-FIX network containing 4 communities. 32 are present inside each community so 16 is the mean degree of every node. The total amount of snapshots is 10, in each snapshot, on a random basis 3 nodes are picked from every community and then again, on a random basis, they are allocated to a different community to create a new network (Besharatnia et al., 2021). The fixed number of edges connecting each community in the SYN-FIX dataset is controlled by a parameter named "Z". This parameter allows researchers to fine-tune the count of edges connecting each community, which in turn affects the degree of community structure present in the network. The parameter Z manages the number of edges that will be connected with the nodes that exist outside of the community. In the experiments performed, the value of Z has been set as 3 and 6.
- SYN-VAR: It is a variable dataset, which changes the number of communities over time. This implies that the number of edges connecting each community is

proportional to the size of the community. In other words, larger communities will have a greater number of edges connecting them than smaller communities. The network on the whole contains a total number of 256 nodes, forming 4 communities with 64 nodes in each community. This dataset also contains 10 snapshots. For the second snapshot, the dataset selects 8 nodes from each community in a random fashion in order to generate a new community of 32 nodes present inside. The process is performed on repeat till the fifth snapshot is done. Subsequently, in the next five snapshots, the nodes are returned to their original communities. Eventually, the mean degree of each node is half the size of its respective community (Besharatnia et al., 2021). Same as SYN-FIX, this dataset also has the Z parameter to be set. The value of Z has been set as Z= 3 and 6.

4.2.2. Large scale dataset

The accuracy of the current research is investigated on large-scale complex datasets using Normalized Mutual Information (NMI) as the evaluation criteria. Four unique varieties of dynamic networks, each comprising of 1000 nodes and spanning across 5 different time intervals, and 33 communities were used. The nodes were randomly reassigned to different communities in each time step with 20% of the nodes being affected. The four dynamic networks were created using the tool described in reference (Greene et al., 2010). These dynamic networks underwent four different types of evolutionary events.

- Birth and Death Model: The Birth and Death model is a type of dynamic network model used to investigate the development of dynamic networks. The model allows for the inclusion or exclusion of nodes from the network as time progresses. The term "birth" pertains to the act of adding fresh nodes to the network, while "death" refers to the removal of existing nodes. This model is useful in simulating real-world scenarios where the arrangement of a network is subject to modifications as time elapses, such as the growth and shrinkage of a social network or the emergence and vanishing of nodes in a biological network. Within the Birth and Death Model, a random sample comprising of 10% of nodes is chosen from a current community to form new communities, while a 10% of the community is removed. This model allows researchers to study the impact of these changes on the network's properties and structure, such as its distribution of degrees, community structure, and measures of centrality.

- Expansion and Contraction Model: The Expansion and Contraction dataset model refers to a community evolution scenario where certain communities increase in size (expand) and others decrease in size (contract) over time. In this model, 10% of nodes from present communities are selected to create new communities, and a new set of communities is created by taking out 10% of nodes from existing communities. This process results in a dynamic network where the size of communities fluctuates over time, with some expanding and others contracting. The Expansion and Contraction dataset model provides a way to study the behavior of algorithms designed for detecting communities within networks where community structure is constantly changing and evolving. In the conducted experiments, during each time interval, 10% of the initial communities expanded by a factor of 25% of their original size, and another 10% of the original communities' contract by 25% of their original size.
- Intermittent Communities Model: The Intermittent Communities dataset model is a type of complex network in which communities appear and disappear periodically. In this model, all network snapshots share an equivalent number of communities as the previous one, but some communities disappear and others appear. The process of community disappearance and appearance is random and follows the condition that a community is not completely eliminated and some nodes still remain to form a new community in the next snapshot. This model simulates real-world complex network scenarios where communities change and evolve over time. In the Intermittent Communities dataset model, during every interval of the dynamic network, 10% of the original communities are temporarily removed. These communities are then added back in the next time step, giving the appearance of intermittent behavior in the community structure.
- Merging and Splitting Model: The Merging and Splitting dataset model is a dynamic network evolution model where communities are merged and split. During each snapshot, a random number of nodes from two or more original communities are selected to form a new community, while a random number of nodes are picked from a single community to establish two or more fresh communities. The merging and splitting process continues with each time step, resulting in dynamic and evolving communities. At each snapshot, 10% of the total communities are separated from

their original community and merged with 10% of other communities to form new communities.

4.3. Experimental Results

This section encompasses a rigorous evaluation of five distinct algorithms, namely CD-INSGA, TMOGA, FacetNet, DYNMOGA, and DYNMODPSO. The experimentation is conducted with meticulous attention to the parameter settings outlined in Section 4.1. The datasets employed for these experiments are comprehensively detailed in Section 4.2, offering an understanding of the contextualized conditions under which the algorithms are assessed. Through these experiments, the goal is to provide a nuanced and comparative analysis of the algorithms, shedding light on their individual and collective performance across diverse datasets and settings.

4.3.1. Results using synthetic dataset

The comparative analysis of five algorithms across diverse datasets is detailed in Tables 4.1, 4.2, 4.3, and 4.4. In the proposed methodology, three distinct objective functions Modularity, Normalized Cut, and NMI were employed. However, for the purpose of result comparison, emphasis was placed on the average NMI values over ten iterations. The normalized mutual information (NMI) score, ranging from 0 to 1, serves as a metric wherein a score closer to 1 signifies a heightened resemblance between the compared communities, indicative of superior results in community detection.

Table 4.1 encapsulates the outcomes of the comparison using the SYN-FIX dataset, with the parameter Z set at three. Across nine iterations, the CD-INSGA algorithm consistently exhibited the most promising results. Notably, in the initial iteration, the TMOGA algorithm outperformed CD-INSGA, DYNMOGA, and DYNMODPSO. However, in subsequent iterations, a convergence occurred, with CD-INSGA, TMOGA, and DYNMOGA algorithms yielding identical results, all attaining the highest NMI value of 1.

Table 4.1. Results with SYN-FIX Z = 3 dataset

Algorithm	1	2	3	4	5	6	7	8	9	10	Time (s)
CD-INSGA	0.943	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	1433
TMOGA	**1**	1	1	1	1	1	1	1	1	1	1836
FacetNet	0.857	0.860	0.860	0.860	0.860	0.858	0.860	0.831	0.857	0.860	**13**
DYNMODPSO	0.979	0.979	0.979	0.979	0.979	0.979	0.979	0.979	0.979	0.979	2499
DYNMOGA	0.975	1	1	1	1	1	1	1	1	1	1789

In terms of computational efficiency, FacetNet had the lowest time complexity but came with a trade-off with regard to result quality. On the contrary, the DYNMODPSO algorithm required the highest amount of time to complete. In general, CD-INSGA delivered the best results, while FacetNet delivered the worst. A graphical representation of the results obtained from the SYN-FIX Z=3 dataset can be seen in Figure 4.1.

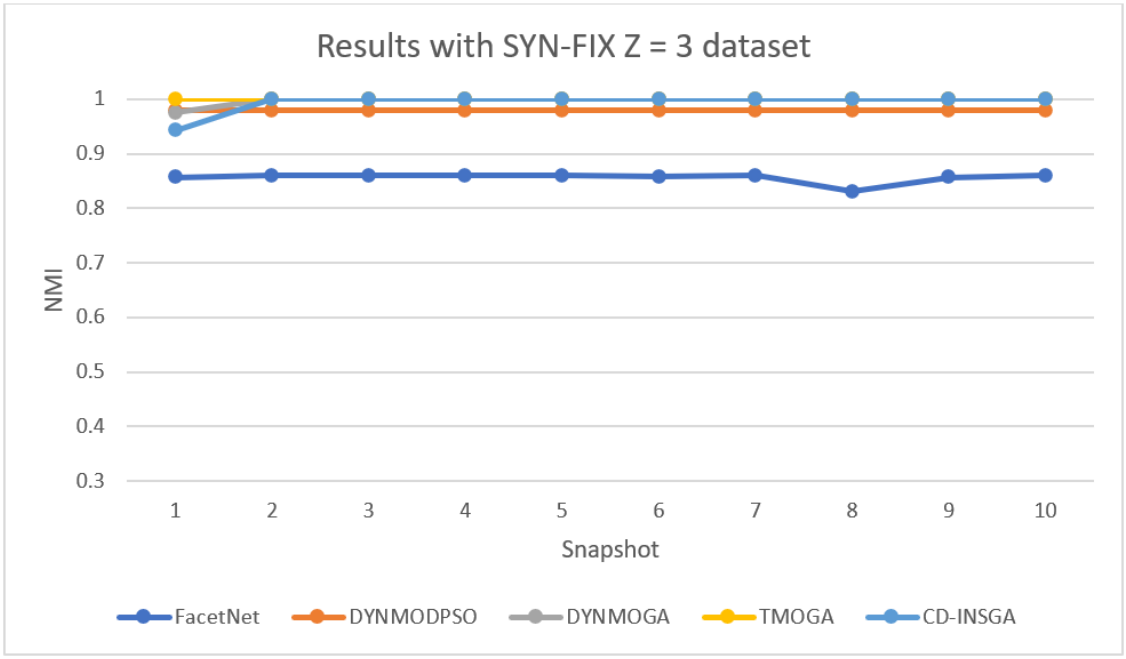

Figure 4.1. Graphical representation of results using SYN-FIX Z=3 dataset

Table 4.2 delineates the comparative outcomes of algorithms utilizing the SYN-FIX dataset, with the parameter Z set at six. The findings underscore the supremacy of the CD-INSGA algorithm over the FacetNet algorithm across all iterations. Notably, in a singular iteration, the TMOGA, DYNMODPSO, and DYNMOGA algorithms outperformed the CD-INSGA

algorithm. However, across the majority of iterations, the CD-INSGA algorithm consistently delivered superior results, showcasing a significant performance differential.

Table 4.2. Results with SYN-FIX Z = 6 dataset

Algorithm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Time (s)**
CD-INSGA	0.947	**1**	0.872	**1**	**1**	**1**	**1**	**1**	**1**	**1**	1558
TMOGA	**1**	1	**1**	1	1	1	1	1	1	1	1831
FacetNet	0.667	0.667	0.668	0.669	0.635	0.534	0.667	0.634	0.667	0. 668	**13**
DYNMODPSO	0.988	1	0.852	0.649	0.922	0.743	0.661	0.810	0.929	0. 656	2444
DYNMOGA	0.886	0.962	0.950	0.950	0.929	0.975	0.942	0.950	0.975	1	1896

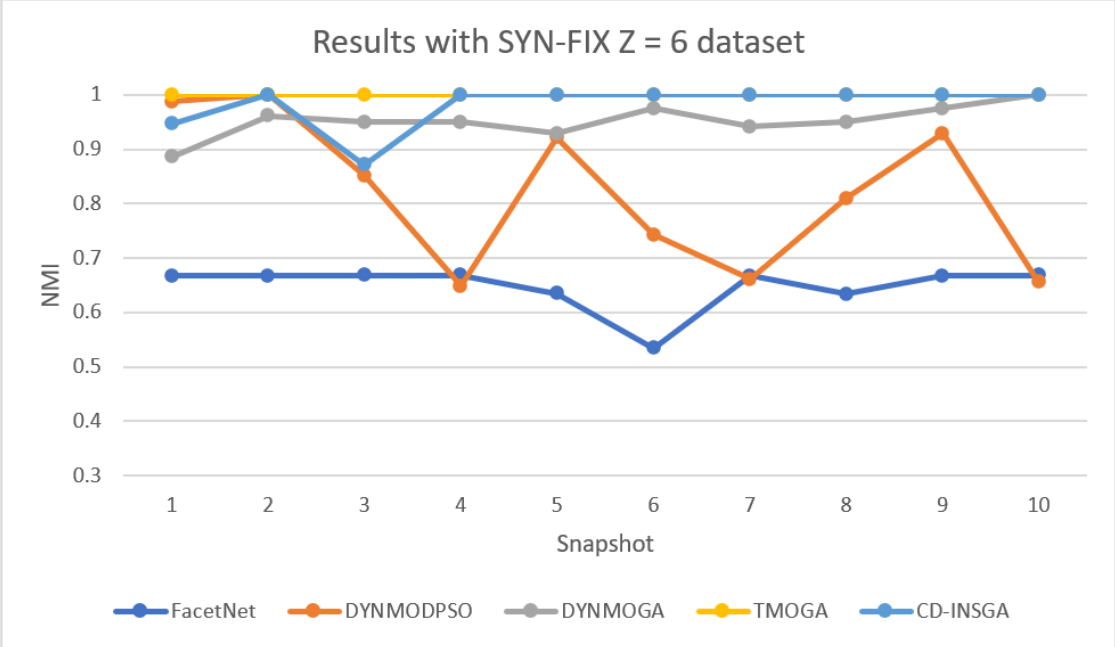

Figure 4.2. Graphical representation of results using SYN-FIX Z=6 dataset

It is noteworthy that the TMOGA algorithm exhibited better performance than the CD-INSGA algorithm but at the expense of higher time complexity. Assessing both results and computational efficiency, the CD-INSGA algorithm emerged as the top-performing algorithm, whereas the FacetNet algorithm, despite displaying the best time complexity, yielded comparatively inferior results.

Figure 4.2 visually encapsulates the outcomes derived from the SYN-FIX dataset (Z=6), providing a graphical representation of the performance of all algorithms under consideration.

The SYN-VAR dataset, characterized by a Z value of three, served as the testbed for experimentation, and the ensuing results are detailed in Table 4.3. Across nine iterations, the CD-INSGA algorithm consistently outperformed its counterparts. Noteworthy is the TMOGA algorithm's attainment of the maximum value in all iterations, albeit accompanied by a heightened time complexity when compared to CD-INSGA. The performance of the DYNMOGA algorithm mirrored that of the CD-INSGA algorithm in each iteration.

Upon averaging results and considering computational efficiency, the CD-INSGA algorithm emerged as the optimal performer, excelling both in terms of results and efficiency. In contrast, the FacetNet algorithm exhibited the least favorable performance, albeit with the advantage of the lowest time complexity.

Table 4.3. Results with SYN-VAR Z = 3 dataset

Algorithm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Time (s)**
CD-INSGA	0.842	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	3061
TMOGA	**1**	1	1	1	1	1	1	1	1	1	3715
FacetNet	1	0.815	0.759	0.702	0.741	0.8	0.824	0.866	0.923	1	**69**
DYNMODPSO	1	0.989	1	0.975	0.975	1	0.995	0.966	0.889	0.973	4685
DYNMOGA	0.934	1	1	1	1	1	1	0.99	1	1	3577

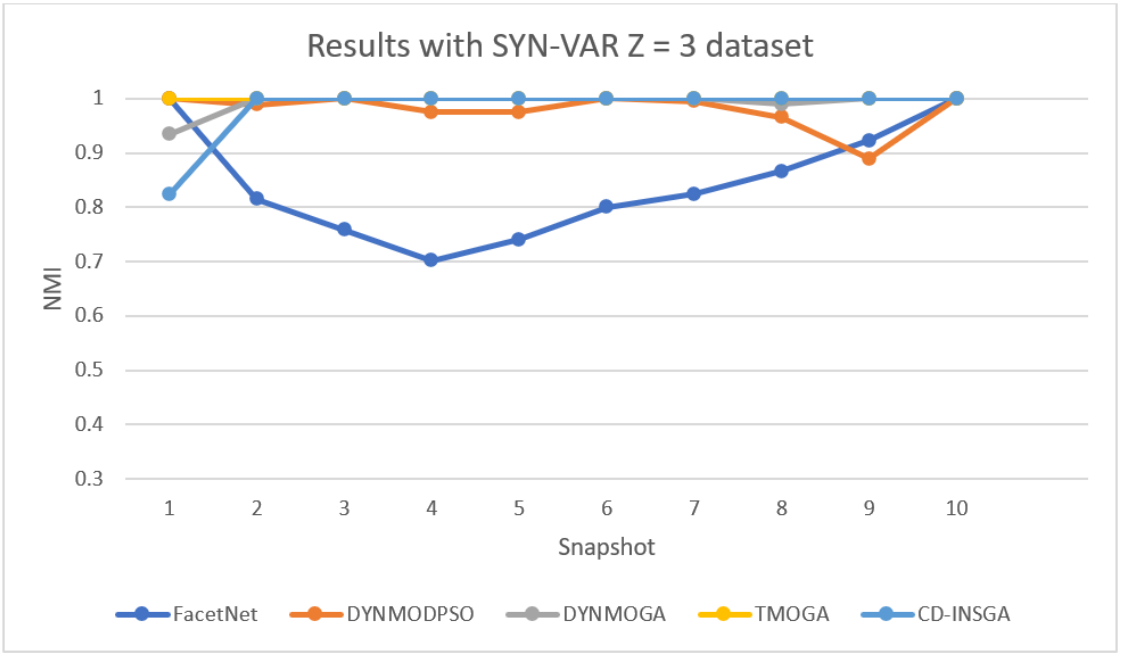

Figure 4.3. Graphical representation of results using SYN-VAR Z=3 dataset

Figure 4.3 offers a graphical representation of the results garnered from all algorithms under scrutiny using the SYN-VAR dataset with Z set at three.

The results obtained from using the SYN-VAR dataset with a Z value set at six are presented in Table 4.4. For this dataset, the CD-INSGA algorithm outperformed all other algorithms under consideration, except for the TMOGA algorithm, which delivered better results during the first iteration. The SYN-VAR dataset with Z=6 was the most complex dataset among all others.

It is worth noting that the results obtained from all algorithms other than the CD-INSGA algorithm were inconsistent and time-consuming when using this dataset. The superior performance of the CD-INSGA algorithm proves that it is the most suitable method for identifying communities in complex networks. The statistical results are presented graphically in Figure 4.4.

Table 4.4. Results with SYN-VAR Z = 6 dataset

Algorithm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Time (s)**
CD-INSGA	0.805	**1**	**1**	**1**	**1**	**1**	**1**	0.982	**1**	**1**	2706
TMOGA	**0.985**	0.988	1	1	1	1	1	**1**	1	1	3683
FacetNet	0.667	0.532	0.421	0.346	0.375	0.416	0.467	0.546	0.602	0.647	**24**
DYNMODPSO	0.896	0.937	0.755	0.892	0.969	0.962	0.864	0.861	0.659	0.602	4694
DYNMOGA	0.459	0.58	0.744	0.832	0.813	0.921	0.871	0.842	0.878	0.586	3638

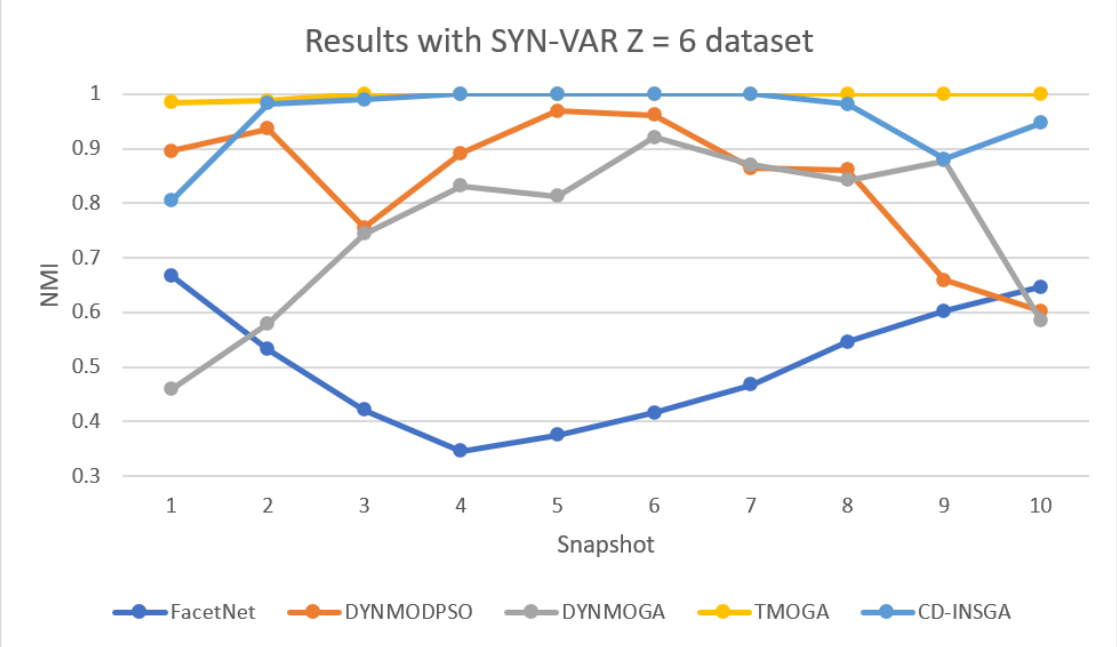

Figure 4.4. Graphical representation of results using SYN-VAR Z=6 dataset

As demonstrated by the experimental results, it is apparent that the CD-INSGA algorithm outperforms the FacetNet and DYNMODPSO algorithms under all conditions. TMOGA mostly outperforms CD-INSGA in the initial iterations, with comparable results in subsequent iterations. DYNMOGA performs better only when the network is less complex, i.e., when there are fewer edges connecting the communities. Although FacetNet boasts an impressive time complexity, its performance is significantly compromised. The proposed algorithm has a lower time complexity than TMOGA, DYNMODPSO, and DYNMOGA, making it the best choice for complex dynamic synthetic networks in terms of both time and computational complexity.

4.3.2. Results using large-scale dataset

The outcomes of the conducted experiments on large-scale datasets are presented in Tables 4.5-4.8. Table 4.5 presents the outcomes of the comparative analysis across five algorithms using the birth and death dataset. Notably, CD-INSGA consistently outperforms DYNMODPSO and DYNMOGA across all iterations. FacetNet demonstrates superior performance in the initial iteration, yet its evolutionary process results in a perceptible decline in subsequent iterations.

Conversely, the TMOGA algorithm exhibits noteworthy performance by surpassing CD-INSGA in two iterations and maintaining competitive outcomes in the remaining iterations. It is evident from the results that the time complexity of CD-INSGA is the least when compared to all other algorithms. Furthermore, the time complexity of the alternative algorithms renders them unsuitable for large-scale datasets.

Table 4.5. Results with Birth and Death Dataset

Algorithm	**1**	**2**	**3**	**4**	**5**	**Time (s)**
CD-INSGA	0.824	0.861	**0.973**	**0.979**	0.882	**885**
TMOGA	0.919	**0.929**	0.936	0.936	**0.945**	6103
FacetNet	**0.967**	0.927	0.754	0.925	0.898	15572
DYNMODPSO	0.832	0.855	0.793	0.809	0.811	34389
DYNMOGA	0.757	0.770	0.769	0.754	0.769	15886

Table 4.6 presents the experimental outcomes concerning the expansion and contraction dataset. Similar to the previous dataset, CD-INSGA consistently outperforms DYNMODPSO and DYNMOGA across all iterations. FacetNet, in alignment with the preceding dataset, demonstrates an initial superiority by producing optimal results in the first

two iterations. However, its efficacy diminishes as the algorithm progresses through subsequent iterations.

Table 4.6. Results with Expansion and Contraction Dataset

Algorithm	1	2	3	4	5	Time (s)
CD-INSGA	0.828	0.868	**0.965**	**0.971**	0.888	**742**
TMOGA	0.927	0.933	0.934	0.944	**0.971**	6266
FacetNet	**0.969**	**0.972**	0.688	0.858	0.836	19256
DYNMODPSO	0.811	0.837	0.788	0.779	0.802	36564
DYNMOGA	0.752	0.762	0.735	0.757	0.761	16998

Remarkably, TMOGA showcases better performance than the proposed CD-INSGA in the final iteration, suggestive of its superior genetic processes. Once again, CD-INSGA proves to be the most time-efficient algorithm, achieving results in the shortest duration. This reaffirms CD-INSGA's suitability for large-scale networks.

Table 4.7 presents the findings from experiments conducted on the intermittent community dataset. Consistent with previous datasets, DYNMODPSO and DYNMOGA exhibit inferior performance. FacetNet demonstrates superiority by outperforming all algorithms in the initial two iterations, albeit experiencing a decline in performance as iterations progress.

TMOGA exhibits comparatively better performance than CD-INSGA in nearly all iterations. However, it is essential to note that its elevated time complexity renders it unsuitable for large-scale networks. This observation underscores the trade-off between performance and scalability, emphasizing the nuanced considerations when selecting algorithms for varying network characteristics.

Table 4.7. Results with Intermittent Communities Dataset

Algorithm	1	2	3	4	5	Time (s)
CD-INSGA	0.833	0.864	**0.976**	**0.985**	0.879	**754**
TMOGA	0.914	0.925	0.932	0.948	**0.946**	5858
FacetNet	**0.934**	**0.981**	0.789	0.833	0.872	19157
DYNMODPSO	0.804	0.815	0.792	0.794	0.793	34038
DYNMOGA	0.747	0.761	0.766	0.763	0.769	15542

Table 4.8 presents the outcomes of experiments conducted on the merging and splitting dataset, facilitating a comparative analysis of five algorithms. In line with previous datasets,

DYNMODPSO and DYNMOGA consistently exhibit suboptimal performance. FacetNet initially excels, producing superior results in the first iteration but experiences a progressive decline in performance.

Conversely, CD-INSGA and TMOGA deliver comparable results, yet the remarkable time efficiency of CD-INSGA sets it apart from TMOGA. This underscores the exceptional time complexity of CD-INSGA compared to its counterpart. These findings emphasize the significance of not only evaluating performance but also considering computational efficiency in algorithm selection.

Table 4.8. Results with Merging and Splitting Dataset

Algorithm	**1**	**2**	**3**	**4**	**5**	**Time (s)**
CD-INSGA	0.826	0.868	0.878	**0.993**	**0.920**	**701**
TMOGA	0.902	**0.918**	**0.917**	0.945	0.916	6490
FacetNet	**0.952**	0.914	0.915	0.913	0.902	10770
DYNMODPSO	0.832	0.832	0.784	0.749	0.748	37223
DYNMOGA	0.752	0.742	0.753	0.721	0.715	15565

From the outcomes of the conducted experiments on large-scale datasets that are presented in Tables 4.5-4.8, it can be observed that the proposed CD-INSGA algorithm shows better performance than DYNMODPSO and DYNMOGA in all snapshots of all datasets, even with the changing communities during each time step. DYNMODPSO initially performs well in the first two snapshots due to its correction technique, but its performance deteriorates in subsequent snapshots. DYNMOGA, which had performed well on synthetic datasets, performs poorly on large-scale datasets. FacetNet achieves optimal results, especially in the first timestep of each dataset, however, it has a significant drawback of having a high time complexity, almost double that of CD-INSGA, making it unsuitable for large-scale datasets. On the other hand, TMOGA has comparable results with CD-INSGA, but a higher time complexity. Taking into account both performance and time complexity, CD-INSGA performs the best.

Upon deeper examination of the results obtained from both synthetic and large-scale networks, it becomes evident that the CD-INSGA algorithm yields notably efficient outcomes concerning Normalized Mutual Information (NMI). This indicates a close resemblance between the communities identified by CD-INSGA and the ground truth communities, thereby suggesting the formation of stable community structures within the

network. Notably, significant emphasis has been placed on evaluating the time complexity of this algorithm.

Analysis of the time complexity results reveals that CD-INSGA ranks as the second-best performer when applied to synthetic datasets. FacetNet exhibits the best time complexity among synthetic datasets but fares poorly in terms of NMI, rendering it impractical for real-world applications. Conversely, CD-INSGA demonstrates superior time complexity performance across large-scale datasets, surpassing all other algorithms. Remarkably, CD-INSGA is specifically tailored to manage larger networks and address real-world problems. Its scalability properties allow it to achieve proportional performance enhancements with increasing input data size.

The integration of cycle crossover (CX) within CD-INSGA enhances the algorithm's balance between exploration and exploitation, thereby accelerating convergence and facilitating quicker solution discovery. Additionally, the implementation of Crowding Distance Elimination (CDE) contributes to algorithmic efficiency by effectively removing less pertinent individuals, thereby reducing time complexity. Notably, feature transfer mechanisms have proven instrumental in extracting valuable features, expediting algorithmic convergence and subsequently reducing time complexity.

Convergence analysis

The convergence of large-scale datasets has been showed in Figures 4.5-4.9. The convergence analysis based on the Pareto convergence graph sheds light on the algorithm's ability to navigate towards optimal solutions. The Birth and Death dataset poses a complex optimization challenge due to its dynamic nature, characterized by fluctuations in birth and death rates across various demographic segments. The Expansion and Contraction Dataset presents a multifaceted optimization challenge, characterized by the dynamic interplay between expansion and contraction dynamics across diverse spatial and temporal scales. The analysis of convergence patterns based on the Pareto convergence graph provides valuable insights into the algorithm's convergence behavior in this complex scenario. The Intermittent Communities Dataset presents a challenging optimization landscape characterized by the intermittent emergence and dissolution of community structures within a dynamic social network. The Merging and Splitting Dataset also presents an optimization challenge because of the dynamic interplay between merging and splitting phenomena within evolving systems.

The analysis of convergence patterns based on the Pareto convergence graph provides valuable insights into the algorithm's convergence behavior in this intricate scenario.

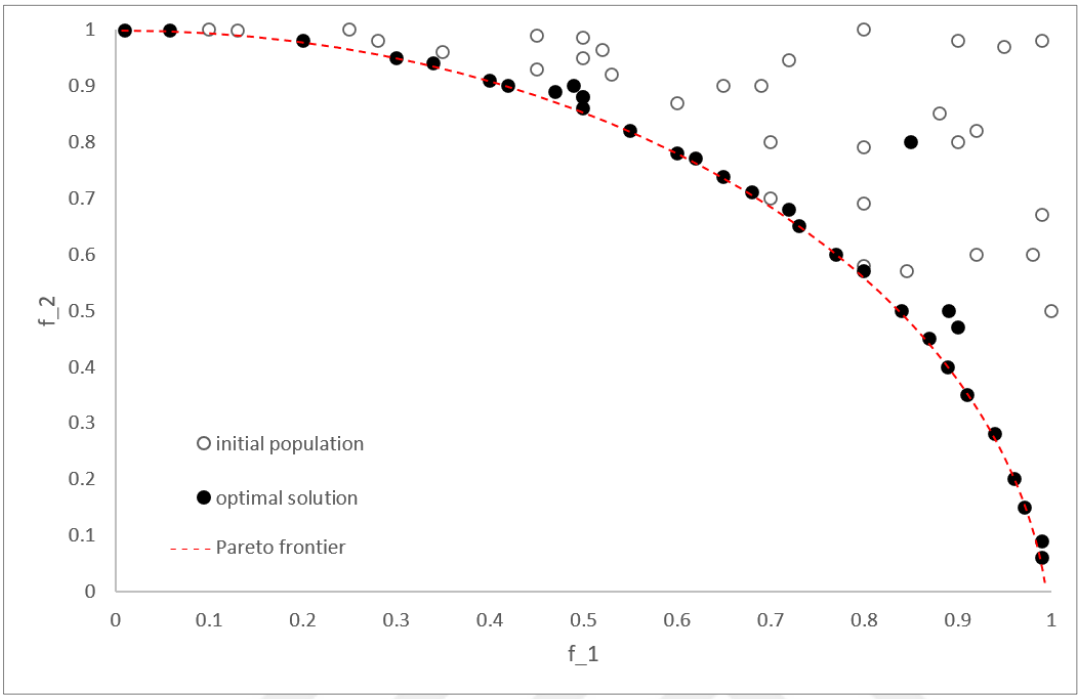

Figure 4.5. Obtained nondominated solutions with CD-INSGA on Birth and Death Dataset

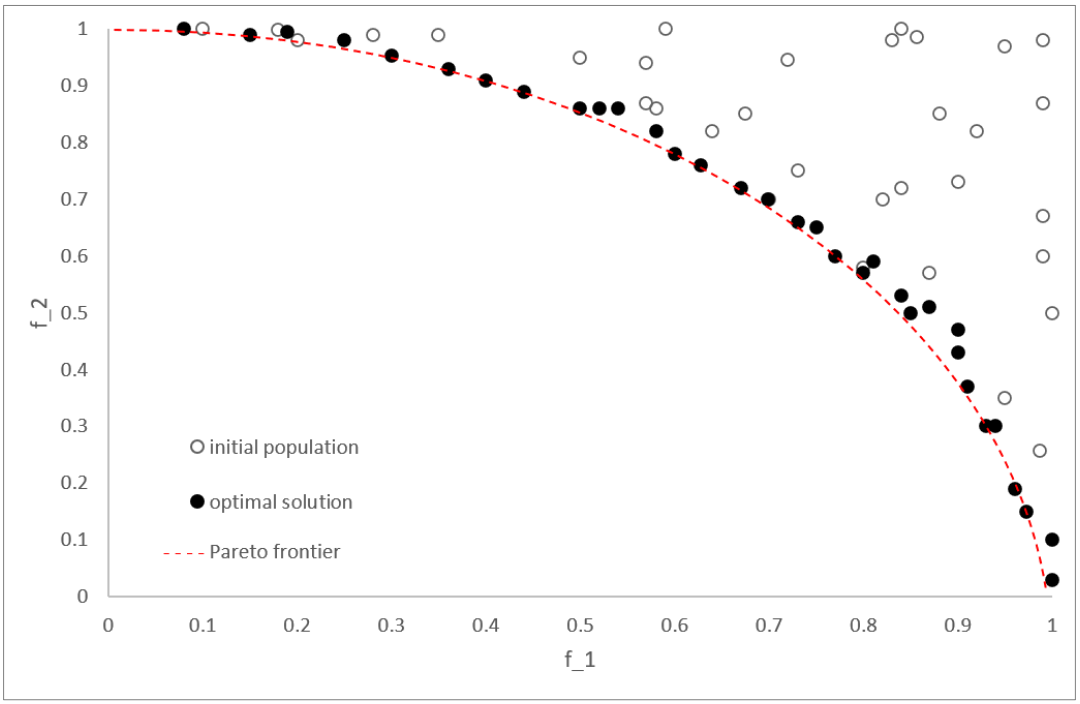

Figure 4.6. Obtained nondominated solutions with CD-INSGA on Expansion and Contraction Dataset

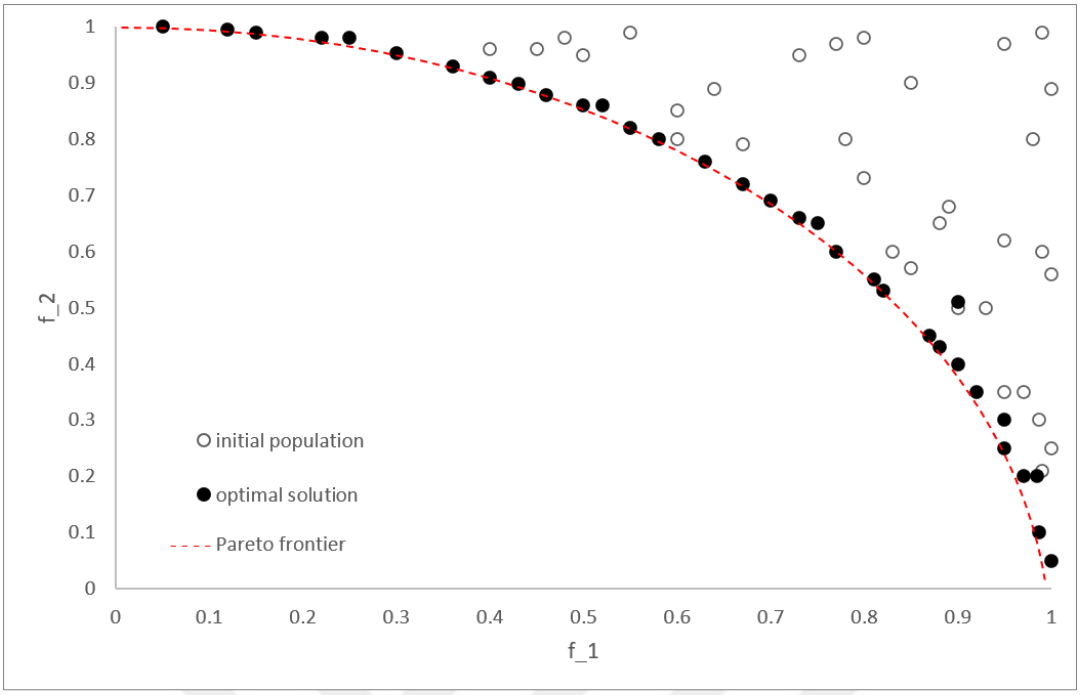

Figure 4.7. Obtained nondominated solutions with CD-INSGA on Intermittent Communities Dataset

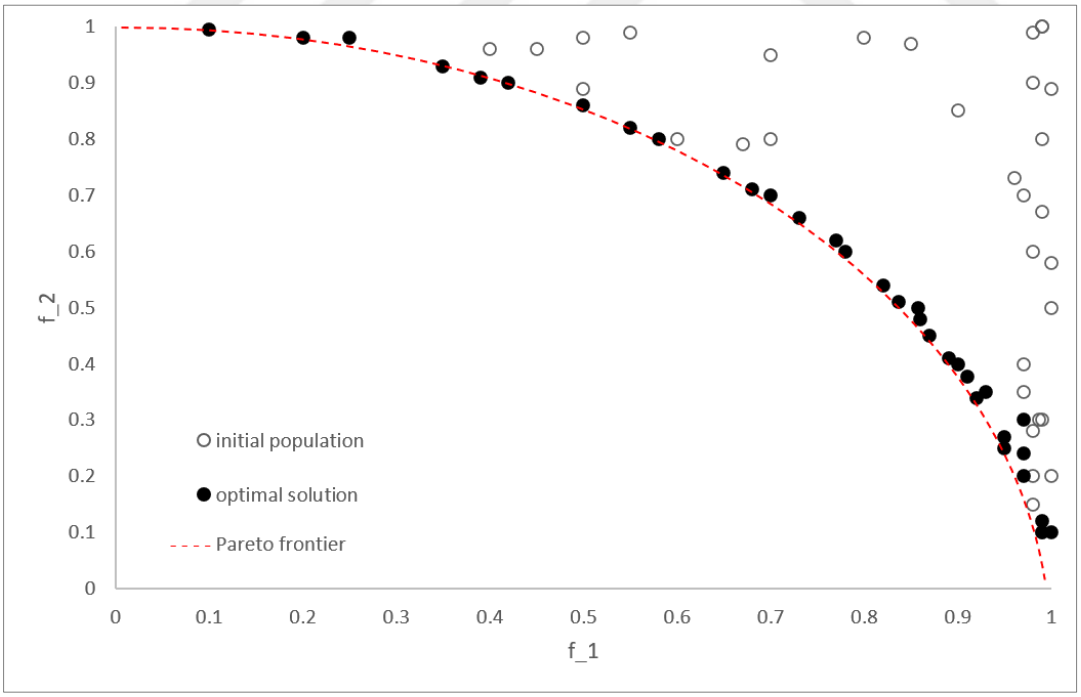

Figure 4.8. Obtained nondominated solutions with CD-INSGA on Merging and Splitting Dataset

Initial Population Distribution:

The initial population distribution, depicted by hollow points in the Pareto convergence graph, showcases a diverse sampling of solutions spanning the solution space. Given the dataset's dynamic nature, the algorithm initiates exploration by diversifying the population, ensuring comprehensive coverage of potential solutions across varying dynamic dataset scenarios.

Pareto Frontier Exploration:

The red dotted line representing the Pareto frontier serves as a pivotal reference for convergence analysis. The algorithm's iterative progress reveals a gradual convergence towards the frontier, indicative of its adeptness in uncovering non-dominated solutions amidst the fluctuating dynamics of all of the datasets used. The convergence trajectory highlights the algorithm's adaptability in exploring and exploiting trade-offs across multiple objectives.

Optimal Solutions Attainment:

The solid points denoting optimal solutions attained in the final generation signify the algorithm's convergence prowess. Positioned strategically along the Pareto frontier, these solutions epitomize a balanced compromise between the changing nature of the datasets, reflecting the algorithm's ability to navigate the dataset's intricate dynamics effectively. The convergence pattern observed underscores the algorithm's capability to converge towards a diverse set of high-quality solutions catering to the multi-objective optimization objectives inherent in the network.

In an optimal convergence scenario, the Pareto convergence graph would exhibit a concentrated cluster of solid points aligned closely along the Pareto frontier. This configuration would signify the algorithm's ability to converge towards a diverse set of optimal solutions, effectively capturing the nuanced trade-offs between the dynamics of networks.

The convergence pattern observed in the Birth and Death Dataset Pareto convergence graph, as shown in Figure 4.5, demonstrates the algorithm's ability to efficiently converge towards optimal solutions despite the dataset's dynamic nature. The gradual convergence towards the Pareto frontier signifies the adeptness of CD-INSGA in balancing exploration and exploitation to uncover non-dominated solutions. The strategic positioning of optimal

solutions along the Pareto frontier shows the algorithm's effectiveness in addressing conflicting objectives related to birth and death rates. The convergence towards diverse and high-quality solutions reflects the algorithm's robustness in addressing multi-objective optimization challenges.

Figure 4.6 shows the convergence trajectory depicted in the Expansion and Contraction Dataset Pareto convergence graph which showcases the capability of the proposed CD-INSGA algorithm to navigate complex spatial and temporal dynamics. The algorithm's iterative progress towards the Pareto frontier demonstrates its proficiency in uncovering non-dominated solutions amidst dynamic expansion and contraction phenomena. The optimal solutions along the Pareto frontier highlights the algorithm's ability to capture fine trade-offs inherent in spatial and temporal planning. By effectively navigating the dynamic expansion and contraction dynamics, the algorithm's efficacy in addressing optimization challenges across diverse spatial and temporal scales can be observed.

The convergence pattern observed in the Intermittent Communities Dataset Pareto convergence graph, represented in Figure 4.7, reflects the proficiency of CD-INSGA algorithm in navigating the intermittent emergence and dissolution of community structures. It can be seen that the initial population was distributed all over the solution space. However, after the completion of all generations, the algorithm successfully takes out the optimal solutions that exist along the Pareto frontier.

In Figure 4.8, the convergence trajectory of the Merging and Splitting Dataset Pareto convergence graph showcases the algorithm's ability to navigate complex merging and splitting phenomena within evolving systems. The algorithm's convergence towards the Pareto frontier underscores its ability to uncover non-dominated solutions amidst dynamic system dynamics. It can be observed that the algorithm reinforces its excellence to be used for analyzing system dynamics and management in network sharing behaviors like merging and splitting datasets.

The consistent convergence patterns observed across the four datasets underscore the versatility and effectiveness of CD-INSGA in addressing complex optimization challenges. By showcasing its adaptability and scalability across diverse datasets, the algorithm solidifies its credibility as a robust optimization tool capable of delivering high-quality communities across various domains and scenarios. In summary, the convergence patterns observed in the Pareto convergence graphs not only validate the effectiveness of the

proposed algorithm but also enhance its credibility as a versatile and reliable optimization tool capable of addressing complex multi-objective optimization challenges across diverse datasets and domains.

5. CONCLUSIONS AND SUGGESTIONS

The exploration of community detection within networks stands as a critical endeavor, particularly when addressing the intricacies of dynamic networks prevalent in real-world scenarios. Despite their undeniable significance, dynamic networks have often been marginalized in past research, primarily due to their inherent complexity. This study sought to redress this imbalance by directing its attention towards dynamic methods, centering its investigation on the widely utilized NSGA-II algorithm which is a powerful method for solving optimization problems with multiple conflicting objectives. Its motivation lies in addressing the challenges posed by such problems, and its use spans various domains where decision-makers need to explore trade-offs among competing goals.

In the course of this study, the identified areas for improvement in NSGA-II led to the development of the CD-INSGA method. This novel approach not only refines the crossover method through the innovative Cycle Crossover (CX) technique but also enhances the crowding distance method with the application of the improved Crowding Distance Elimination (CDE) approach. The iterative elimination of solutions with the least crowding distance until the desired non-dominated solutions are achieved marks a significant advancement.

Furthermore, the optimization of three objective functions—Modularity, NMI, and Normalized Cut—served as a robust evaluative framework for the acquired communities, with Normalized Mutual Information acting as the performance metric. The incorporation of a feature transfer learning strategy in the evolutionary algorithm, leveraging previously discovered vital information in subsequent iterations, emerged as a strategic enhancement.

Analysis of the time complexity results reveals that CD-INSGA ranks as the second-best performer when applied to synthetic datasets. FacetNet exhibits the best time complexity among synthetic datasets but fares poorly in terms of NMI, rendering it impractical for real-world applications. Conversely, CD-INSGA demonstrates superior time complexity performance across large-scale datasets, surpassing all other algorithms. Remarkably, CD-INSGA is specifically tailored to manage larger networks and address real-world problems. Its scalability properties allow it to achieve proportional performance enhancements with increasing input data size.

The proposed CD-INSGA method not only excels in producing enhanced community structures and optimal solutions but also exhibits a remarkable capacity for preserving solution diversity. Rigorous experimentation with synthetic dynamic networks and large-scale dynamic network datasets unequivocally demonstrated the superior performance of the proposed algorithm, surpassing established methods such as TMOGA, FaceNet, DYNMODPSO, and DYNMOGA. The augmentation of the number of objective functions and the marked improvement in time complexity further affirm the adaptability of the suggested technique for identifying communities within complex large-scale dynamic networks.

Future Directions:

Looking ahead, there are promising avenues for extending the impact of the CD-INSGA method. Specifically, future work should prioritize the implementation of the proposed method for handling overlapping communities. This enhancement holds the potential to significantly broaden the applicability of the algorithm to real-life networks where the phenomena of community overlap is prevalent. Additionally, delving into further refinements in the feature transfer learning strategy and investigating its adaptability to diverse domains would be instrumental in advancing the algorithm's efficacy. Continuous exploration and refinement in these directions will contribute to the ongoing evolution of community detection algorithms and their practical deployment in complex dynamic network environments.

REFERENCES

Alamsyah, A., & Rahardjo, B. (2014). Community detection methods in social network analysis. *Advanced Science Letters*, *20*(1), 250-253.

Arowolo, M. O., Isiaka, R. M., Abdulsalam, S. O., Saheed, Y., & Gbolagade, K. A. (2017). A comparative analysis of feature extraction methods for classifying colon cancer microarray data. *EAI endorsed transactions on scalable information systems*, *4*(14), e2-e2.

Atay, Y., Koc, I., Babaoglu, I., & Kodaz, H. (2017). Community detection from biological and social networks: A comparative analysis of metaheuristic algorithms. *Applied Soft Computing*, *50*, 194-211.

Azad, S., & Devi, S. (2020). Tracking the spread of COVID-19 in India via social networks in the early phase of the pandemic. *Journal of travel medicine*, *27*(8), taaa130.

Bagchi, T. P. (1999). Multiobjective scheduling by genetic algorithms. *Springer Science & Business Media*, 181-186.

Besharatnia, F., Talebpour, A., & Aliakbary, S. (2021). Metaheuristic multi-objective method to detect communities on dynamic social networks. *International Journal of Computational Intelligence Systems*, 14(1), 1356-1372.

Chi, Y., Song, X., Zhou, D., Hino, K., & Tseng, B. L. (2009). On evolutionary spectral clustering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *3*(4), 1-30.

Danon, L., Diaz-Guilera, A., Duch, J., & Arenas, A. (2005). Comparing community structure identification. *Journal of statistical mechanics: Theory and experiment*, *2005*(09), P09008.

de França, F. O., & Coelho, G. P. (2015). A flexible fitness function for community detection in complex networks. *Complex Networks VI: Proceedings of the 6th Workshop on Complex Networks CompleNet 2015, Springer International Publishing*, 1-12.

Deb, K., & Agrawal, R. B. (1995). Simulated binary crossover for continuous search space. *Complex systems*, *9*(2), 115-148.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, *6*(2), 182-197.

Dhillon, I. S., Guan, Y., & Kulis, B. (2004). Kernel k-means: spectral clustering and normalized cuts. *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 551-556.

Folino, F., & Pizzuti, C. (2013). An evolutionary multiobjective approach for community discovery in dynamic networks. *IEEE Transactions on Knowledge and Data Engineering*, *26*(8), 1838-1852.

Girvan, M., & Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12), 7821-7826.

Goldberg, D. E. (1989). Optimization, and machine learning. *Genetic algorithms in Search, 157-200*.

Greene, D., Doyle, D., & Cunningham, P. (2010). Tracking the evolution of communities in dynamic social networks. *2010 international conference on advances in social networks analysis and mining, IEEE*, 176-183.

Groşan, C., Oltean, M., & Oltean, M. (2003). The role of elitism in multiobjective optimization with evolutionary algorithms. *Acta Univ. Apulensis Math. Inform*, 83-90.

Hasanoglu, M. S., & Dolen, M. (2020). Comparison of multi-objective and single-objective approaches in feasibility enhanced particle swarm optimization. *Journal of the Faculty of Engineering and Architecture of Gazi University*, *35*(2), 887-900.

Hingee, K., & Hutter, M. (2008). Equivalence of probabilistic tournament and polynomial ranking selection. *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, 564-571.

Ishibuchi, H., Tsukamoto, N., & Nojima, Y. (2008). Examining the effect of elitism in cellular genetic algorithms using two neighborhood structures. *Parallel Problem Solving from Nature–PPSN X: 10th International Conference, Dortmund, Germany, September 13-17, 2008. Proceedings 10, Springer Berlin Heidelberg*, 458-467.

Koc, I. (2022). A fast community detection algorithm based on coot bird metaheuristic optimizer in social networks. *Engineering Applications of Artificial Intelligence*, *114*, 105202.

Li, C., Tang, Y., Tang, Z., Cao, J., & Zhang, Y. (2022). Motif-based embedding label propagation algorithm for community detection. *International Journal of Intelligent Systems*, *37*(3), 1880-1902.

Liakos, P., Ntoulas, A., & Delis, A. (2017). COEUS: community detection via seed-set expansion on graph streams. *2017 IEEE International Conference on Big Data (Big Data) IEEE*, 676-685.

Lin, Y.-R., Chi, Y., Zhu, S., Sundaram, H., & Tseng, B. L. (2008). Facetnet: a framework for analyzing communities and their evolutions in dynamic networks. *Proceedings of the 17th international conference on World Wide Web*, 685-694.

Liu, J., & Chen, X. (2019). An improved NSGA-II algorithm based on crowding distance elimination strategy. *International Journal of Computational Intelligence Systems*, *12*(2), 513-518.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software*, *69*, 46-61.

Mu, C., Zhang, J., Liu, Y., Qu, R., & Huang, T. (2019). Multi-objective ant colony optimization algorithm based on decomposition for community detection in complex networks. *Soft Computing*, *23*, 12683-12709.

Panchal, A. (2021). *Overlapping community detection in social networks*, Published Masters Thesis, San Jose State University, California.

Pizzuti, C. (2008). GA-Net: A Genetic Algorithm for Community Detection in Social Networks. *International conference on parallel problem solving from nature, Berlin, Heidelberg: Springer Berlin Heidelberg*, 1081-1090.

Prabhu, S. M., Subramanyam, N., & Girdhar, R. (2021). Containing COVID-19 pandemic using community detection. *Journal of Physics: Conference Series* (Vol. 1797, No. 1, p. 012008).

Puljić, K., & Manger, R. (2013). Comparison of eight evolutionary crossover operators for the vehicle routing problem. *Mathematical Communications*, *18*(2), 359-375.

Purshouse, R. C., & Fleming, P. J. (2002). Why use elitism and sharing in a multi-objective genetic algorithm? *Proceedings of the 4th Annual Conference on Genetic and Evolutionary computation*, 520-527.

Raghavan, U. N., Albert, R., & Kumara, S. (2007). Near linear time algorithm to detect community structures in large-scale networks. *Physical review E*, *76*(3), 036106.

Roghani, H., Bouyer, A., & Nourani, E. (2021). PLDLS: A novel parallel label diffusion and label Selection-based community detection algorithm based on Spark in social networks. *Expert Systems with Applications*, *183*, 115377.

Rostami, M., Berahmand, K., & Forouzandeh, S. (2021). A novel community detection based genetic algorithm for feature selection. *Journal of Big Data*, *8*(1), 1-27.

Sampson, R. J., & Laub, J. H. (1997). A life-course theory of cumulative disadvantage and the stability of delinquency. *Developmental theories of crime and delinquency*, *7*, 133-161.

Shaik, T., Ravi, V., & Deb, K. (2021). Evolutionary multi-objective optimization algorithm for community detection in complex social networks. *SN Computer Science*, *2*, 1-25.

Shang, R., Zhang, W., Zhang, J., Feng, J., & Jiao, L. (2022). Local community detection based on higher-order structure and edge information. *Physica A: Statistical Mechanics and its Applications*, *587*, 126513.

Shishavan, S. T., & Gharehchopogh, F. S. (2022). An improved cuckoo search optimization algorithm with genetic algorithm for community detection in complex networks. *Multimedia Tools and Applications*, *81*(18), 25205-25231.

Tahmasebi, S., Moradi, P., Ghodsi, S., & Abdollahpouri, A. (2019). An ideal point based many-objective optimization for community detection of complex networks. *Information Sciences*, *502*, 125-145.

Tharwat, M., Fudzee, M. F. M., Kasim, S., Ramli, A. A., & Ali, M. K. (2021). Multi-objective NSGA-II based community detection using dynamical evolution social network. *Int. J. Electr. Comput. Eng.*, *11*(5), 4502-4512.

Wickramasinghe, A. N. (2021). *Community detection in social networks with an application to COVID-19 data*, Published Masters Thesis, The University of Manitoba, Winnipeg.

Wickramasinghe, A. N., & Muthukumarana, S. (2021). Social network analysis and community detection on spread of COVID-19. *Model Assisted Statistics and Applications*, *16*, 37-52.

Yin, Y., Zhao, Y., Li, H., & Dong, X. (2021). Multi-objective evolutionary clustering for large-scale dynamic community detection. *Information Sciences*, *549*, 269-287.

Zhang, L., & Yan, X. (2019). An improved LFM algorithm based on fitness function and community similarity. *2019 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA / BDCloud / SocialCom / SustainCom)*, 9-15.

Zhao, X., Liang, J., & Wang, J. (2021). A community detection algorithm based on graph compression for large-scale social networks. *Information Sciences*, *551*, 358-372.

Zhou, H., & Zhang, Y. (2022). Unsupervised community detection algorithm based on graph convolution network and social media. *Mobile Information Systems, 2022(1), 1-10*.

ZhuЃ, X., & GhahramaniЃн, Z. (2002). Learning from labeled and unlabeled data with label propagation. *ProQuest Number: INFORMATION TO ALL USERS, Carnegie Mellon University, Pittsburgh.* URL: https://mlg.eng.cam.ac.uk/zoubin/papers/CMU-CALD-02-107.pdf. Last Access Date: 10.12.2022.

Zou, J., Lin, F., Gao, S., Deng, G., Zeng, W., & Alterovitz, G. (2021). Transfer learning based multi-objective genetic algorithm for dynamic community detection. *arXiv preprint arXiv:2109.1513*.

CURRICULUM VITAE

Personal Information

Surname, first name	: TARRAR, Kainat Akhtar
Nationality	: Pakistan
E-mail	:

Education

Degree	Education Unit	Date of graduation
Master's Degree	Gazi University / Computer Sciences	2024
Bachelor	COMSATS University of Information Technology, Islamabad.	2019
Highschool	Islamabad Model School for Girls	2014

Work Experience

Year	Place	Duty
2023-Present	Veribox Software	Net Core Developer
2021-2023	Eryaz Software	Full Stack Engineer
2018-2019	COMSATS University of Information Technology, Islamabad.	React Native Developer

Foreign languages

Urdu, Hindi, Punjabi, English, Turkish.

Publications

Tarrar, K. A., & Atay, Y. (2023). Multi-Objective Evolutionary Approach to Discover Subnetworks, V-*International Antalya Scientific Research and Innovative Studies Cong.*, July 26-28, 2023, Antalya, Türkiye.

Hobbies

Coding, cooking.



GAZİLİ OLMAK AYRICALIKTIR