KARINCA KOLONİSİ ALGORİTMASI İLE BİLGİSAYAR AĞLARININ TOPOLOJİK EN İYİLENMESİ

TOPOLOGICAL OPTIMIZATION OF COMPUTER NETWORKS USING ANT COLONY ALGORİTHM

YAVUZ SELİM ÖZDEMİR

Başkent Üniversitesi
Lisansüstü Eğitim Öğretim ve Sınav Yönetmeliğinin
BİLGİSAYAR Mühendisliği Anabilim Dalı İçin Öngördüğü
YÜKSEK LİSANS TEZİ
olarak hazırlanmıştır.

2008

Fen Bilimleri Enstitüsü Müdürlüğü'ne,

Bu çalışma, jürimiz tarafından **BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI'nda YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Başkan : Prof. Dr. İmdat KARA

Üye (Danışman) : Prof. Dr. Berna DENGİZ

Üye : Prof. Dr. Fulya ALTIPARMAK

ONAY

Bu tez 09/09/2008 tarihinde Enstitü Yönetim Kurulunca belirlenen yukarıdaki jüri üyeleri tarafından kabul edilmiştir.

...../...../.....

Prof. Dr. Emin AKATA

FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRÜ

TEŞEKKÜR

Bu çalışmada değerli görüş ve katkılarıyla her konuda yardımcı olan hocam Sayın Prof. Dr. Berna DENGİZ'e (tez danışmanı),

Bugüne kadar üzerimdeki emek ve katkılarından dolayı değerli hocam Prof. Dr. İmdat KARA'ya,

Eğitim hayatımın her döneminde her zaman maddi ve manevi destekleriyle yanımda olan biricik Annem ve Babama,

Görüş ve katkılarından dolayı değerli jüri üyelerime,

Çalışmam süresinde desteklerini esirgemeyen değerli arkadaşlarım Yiğit Koray GENÇ, Uğur BAÇ ve Mert AKTAŞ 'a,

Teşekkürü bir borç bilirim.

ÖZ

KARINCA KOLONİSİ ALGORİTMASI İLE BİLGİSAYAR AĞLARININ TOPOLOJİK OPTİMİZASYONU

Yavuz Selim ÖZDEMİR

Başkent Üniversitesi Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayarların ve bilgisayar ağlarının yaygınlaştığı günümüzde düşük maliyetli ve güvenilirliği yüksek bilgisayar ağlarının kurulumunun önemi giderek artmaktadır. Teknolojinin hızla gelişmesinden dolayı büyük boyutlu problemlerin çözülmesi üstel olarak artan hesaplama karmaşıklığı gerektirmekte ve tam sonucun hesaplanması imkansız hale gelmektedir. Bu problem için bugüne kadar Genetik Algoritma, Tabu Arama, Tavlama Benzetimi, Yapay Sinir Ağları gibi çeşitli sezgisel algoritmalar geliştirilerek kullanılmıştır. Bu çalışmada NP-zor bir problem olan bilgisayar ağlarının güvenilirlik kısıtı altında topolojik en iyilenmesi ele alınarak bu problemin çözümü için bilgiye dayalı melez bir karınca kolonisi algoritması geliştirilmiş ve algoritmanın etkinliği çalışma zamanı ve çözüm kalitesi açısından incelenmiştir.

Anahtar Sözcükler: Karınca Kolonisi Sezgiseli, Ağ Güvenilirliği, Topolojik Eniyileme.

Danışman: Prof. Dr. Berna DENGİZ, Başkent Üniversitesi, Endüstri Mühendisliği Bölümü.

ABSTRACT

TOPOLOGICAL OPTIMIZATION OF COMPUTER NETWORKS USING ANT COLONY ALGORİTHM

Yavuz Selim ÖZDEMİR

Başkent University, Institute of Science

Department of Computer Engineering

Today, with the proliferation of computer and computer networks, the need for designing low cost and reliable computer networks is crucial. Due to the recent technological developments solving the big size problems is yielding a computation complexity; and computation of the exact result is becoming impossible. To resolve this problem, past research developed and used intuitive algorithms such as, Genetic Algorithm, Tabu Search, Simulated annealing, Neural Networks. In this research, to solve this NP-hard problem, topological optimization of the computer networks within the reliability constraints, a knowledge based hybrid ant colony algorithm is developed and adopted. The efficiency of algorithm analyzed for computation time and solution quality.

Key Words: Ant Colony Heuristic, Network Reliability, Topological Optimization.

Adviser: Prof. Dr. Berna DENGİZ, Başkent University, Department of Industrial Engineering

İÇİNDEKİLER LİSTESİ

Sayfa

ŞEKİLLER LİSTESİ

Sayfa

ÇİZELGELER LİSTESİ

Sayfa

SİMGELER VE KISALTMALAR LİSTESİ

$R(G_x)$	x aday çözümünün güvenilirliği
G	olasılıksal grafik
R_0	istenilen network güvenilirliği
L	ağdaki bağlantıların tamamı
N	ağdaki düğümlerin tamamı
(i,j)	i ve j düğümleri arasındaki bağlantı
x_{ij}	i ve j düğümleir arasındaki hat, $x_{i,j} \in \{0,1\}$
x	ağın bağlantı topolojisi $\{x_{1,1}, x_{1,2}, ..., x_{i,j}, ..., x_{N,N-1}\}$
C^{ℓ}	toplam hat uzunluğu
p	hat güvenilirliği
q	hattın bozulma olasılığı (p+q)=1
$Ç_{uzayı}$	çözüm uzayı
c_{ij}	i ve j düğümleri arasındaki hattın maliyeti.
p_{ij}^{ℓ}	ℓ karıncasının i düğümünden j düğümüne geçme olasılığı
τ_{ij}	i ve j düğümleri arasındaki feromon matris değeri
τ_0	başlangıç feromon değeri
η_{ij}	i ve j düğümleri arasındaki sezgisel matris değeri
ρ	buharlaşma sabiti
α	feromon katsayısı
β	sezgisel katsayısı
ℓ	sitemdeki ℓ. karınca
N_i^{ℓ}	ℓ karıncasının uğramadığı i düğümleri
d_{ij}	i ve j düğümleri arasındaki uzaklık
m	toplam karınca sayısı
T	anlık tekrar sayısı
T^{ℓ}	toplam tekrar sayısı
C^{nn}	en yakın komşuluklardan oluşan tur uzunluğu

$R(G_x)_{MC}$	x'e göre MC simülasyonu ile tahmin edilen tüm-terminal network güvenilirliği

GSP	Gezgin Satıcı Problemi
KS	Karınca Sistemi
EKS	Elitist Karınca Sistemi
DTKS	Derece Tabanlı Karınca Sistemi
EEKS	En Büyük En Küçük Karınca Sistemi
KQ	Karınca-Q
KKS	Karınca Kolonisi Sistemi
ÖDH	Özdüzenleyici Haritalar
GA	Genetik Algoritma
TB	Tavlama Benzetimi
TB_udh	Tavlama Benzetimi Uzun Dönem Hafıza
TA	Tabu Arama
KSE	Kuş Sürüsü Eniyilemesi
DKİ	Değişken Komşu İniş Algoritması

1. GİRİŞ

Yirminci yüzyılın ortalarında keşfedilen ve son çeyreğinde kullanımı hızla yaygınlaşan bilgisayarlar, yirmi birinci yüzyılda da gün geçtikçe daha hızlı bir şekilde gelişimini sürdürmektedirler. Bilgisayarların ortaya çıkması ve teknolojideki gelişmeler ile birlikte bilginin paylaşılması, hızlı iletilmesinin sağlanması amacıyla bilgisayarlar arasında iletişim kurulması ve bilgisayar ağlarının kurulması gündeme gelmiştir. En küçük işletmelerden evlerimize kadar ulaşarak yaygınlaşan iletişim ağlarının kullanım oranı artışı ve bilgisayarların maliyetleri göz önüne alındığında, bilgisayar ağlarının maliyetlerinin önemi açıktır. Bu nedenle, bu alanda çalışan araştırmacılar güvenilir ve ekonomik ağ tasarımları elde etmek üzere ağ topolojisi problemleriyle ilgilenmeye başlamışlardır.

Bilgisayar ağlarının topolojik en iyilenmesi problemi, bir ağdaki düğüm (bilgisayar) ve hatların, bağlantı maliyetini en küçükleyen ve güvenilirlik, veri akışı ve hız gibi çeşitli performans kriterlerinden birini veya birkaçını sağlayan, düğümler arasındaki en iyi bağlantıyı veren tasarımı bulmak olarak tanımlanabilir. Günümüzde kullanılan bilgisayar ağlarında, büyük boyutlarda veri trafiğini taşıyan ve yüksek hızlarda çalışan bir ana bilgisayar ağı ve ana bilgisayar ağı ile kullanıcılar arasında veri akışı sağlayan yerel erişim ağlarının bulunduğu hiyerarşik bir yapı vardır.

Bilgisayar ağlarında güvenilirlik kriteri, ağdaki hatlardan biri veya birkaçının rasgele olarak arızalanması durumunda, ağın çalışmaya devam edebilmesinin sağlanması durumudur. Literatüre baktığımızda, güvenilirlik iki şekilde hesaplanmaktadır. Birincil olarak, istenilen düğüm çiftleri arasında (s – t güvenilirliği) belirlenen güvenilirlik düzeyinin sağlanması ve veri iletişiminin devam etmesi durumu söz konusudur. İkincil olarak ise, ağdaki bütün düğümler arasında (tüm – terminal güvenilirliği) belirlenen güvenilirlik düzeyinin sağlanması ve veri iletişiminin devam etmesi durumudur. Bu çalışmada güvenilirlik kısıtı altında en küçük maliyetli bilgisayar ağlarının topolojik tasarımı problemi ele alınmıştır. Söz konusu problemde, ana bilgisayar ağlarının topolojik tasarımı ve tüm – terminal güvenilirlik kısıtı ile ilgilenilmiştir.

Ana bilgisayar ağlarında güvenilirlik özellikle önemli bir başlıktır. Bunun sebebi, ana bilgisayar ağında meydana gelebilecek bir bozukluğun veya veri iletişimi kesintisinin tüm bilgisayar ağındaki veri iletişimini etkilemesidir. Oysaki erişim ağlarında meydana gelen veri iletişimi kesintileri, sadece bu yerel bilgisayar ağını etkilemekte, bilgisayar ağının genelinde bir soruna neden olmamaktadır.

1.1 Problemin Tanımı

Güvenilirlik kısıtı altında en küçük maliyetli bilgisayar ağlarının tasarımı NP-zor bir problemdir. Bunun iki ana sebebi bulunmaktadır, Birincisi, ağdaki tüm düğüm ve hat sayısına bağlı olarak çözüm uzayının üstel olarak büyümesidir. İkinci sebebi ise, bilgisayar ağının boyutları büyüdükçe, güvenilirlik hesabı için dikkate alınması gereken durumların sayısında da üstel artmaktadır. Bu iki sebepten ötürü, büyük boyutlu bilgisayar ağları, güvenilirlik kısıtı altında klasik eniyileme yöntemleri ile çözülememektedir.

Literatürde büyük boyutlu ağ tasarını problemleri için çeşitli birerlemeye dayalı metotlar ya da sezgisel yöntemler kullanılmıştır. Aggarwal et al. [1], Jan et al. [2] küçük boyutlu ağlar için birerlemeye dayalı yöntemleri kullanmışlardır. Sezgisel yöntemlerde ise Dengiz et al. [3] probleme özgü olarak Genetik Algoritma (GA) [4], Tabu Tarama (TA) [5], Tavlama Benzetimi (TB) [6], Yapay Sinir Ağları (YSA) [7], gibi yöntemler kullanılarak büyük boyutlu problemlerin çözümleri araştırılmıştır.

Bu çalışmada, Karınca Kolonisi Eniyilemesi yöntemi kullanılarak bir Melez Karınca Kolonisi Algoritması geliştirilmiş ve istenilen güvenilirlik kısıtı altında büyük boyutlu, en küçük maliyetli bilgisayar ağı tasarımı için uygulanmıştır.

1.2 Tez Düzeni

İkinci bölümde, bilgisayar ağlarının tasarımı ve güvenilirliği ile ilgili genel bilgilerin yanı sıra, maliyet kısıtı altında en küçük maliyetli topolojik ağ tasarımı ile ilgili bilgilere yer verilmiştir.

Üçüncü bölümde, kullanılan karınca kolonisi algoritması detaylı bir şekilde incelenmiş, karıncaların gerçek hayattaki davranışları ve bilgisayar ortamında kullanılan prensipleri ile algoritmanın adımları hakkında detaylı bilgi verilmiştir.

Dördüncü bölümde, öncelikle literatürde bilgisayar ağı tasarımı problemi için günümüze kadar kullanılan yaklaşımlar ve yapılan çalışmalar hakkında detaylı bir araştırma yapılmış, daha sonra karınca kolonisi algoritması kullanarak bilgisayar ağı tararımı problemini inceleyen çalışmalar hakkında bilgi verilmiştir.

Beşinci bölümde, geliştirilen Melez Karınca Kolonisi Algoritması ele alınmış ve algoritma detaylı bir şekilde açıklanmıştır. Algoritmada kullanılan parametreler açıklanmıştır. Bulunan sonuçlar algoritmanın etkinliğini göstermek amacı ile literatürde bulunan en iyi sonuçlar ile çözüm uzayında aranan nokta sayısı kriterlerine göre incelenmiş ve karşılaştırılmıştır. Daha sonra en iyi çözümleri klasik eniyileme yöntemleri ile bulunamayan büyük boyutlu test problemleri Melez Karınca Kolonisi Algoritması ile çözülerek problem boyutu ve bulduğu en küçük maliyete göre değerlendirilmiştir.

Altıncı bölümde, önerilen algoritma, küçük, orta ve büyük boyutlu test problemleri üzerinde uygulanmıştır. Elde edilen sonuçlar literatürdeki önerilen diğer algoritmaların sonuçları ile çözüm kalitesi ve performans ölçütlerine göre karşılaştırılmıştır.

Son olarak sonuçlar bölümünde, çalışmada elde edilen sonuçlar değerlendirilmiş ve önerilen algoritmanın sağladığı faydalara değinilmiştir.

2. BİLGİSAYAR AĞLARININ TOPOLOJİK TASARIMI

Bilgisayarların her geçen gün daha hızlı geliştiği günümüzde, bilgisayarlar arası bağlantılar ve veri alışverişi git gide daha çok önem kazanmaktadır. Bilgisayarlar arasında hızlı ve güvenilir bağlantıların kurulabilmesi, küresel dünyada büyüyen bilgisayar ağ tasarımları için önemli problemlerden bir tanesi durumundadır. Bilgisayar ağlarının en önemli görevleri farklı işlem gücüne sahip birden çok bilgisayar arasında veri ve dosya paylaşımının sağlanması, ses, görüntü, yazı aktarımı yapılması, ortak kullanılan programlar arasında veri akışının sağlanması ve paralel işlem yapılmasına olanak sağlayabilmesidir.

2.1 Bilgisayar Ağlarında Kullanılan Topolojiler

Bilgisayarlar arası bağlantılar temel olarak ikiye ayrılmaktadır. Birinci tür Yerel Erişimli Ağlaradır (Local Area Networks). Bu ağlar genelde birbirine yakın bilgisayarları bağlamak için kullanılmaktadır. Örnek olarak, ev, okul, hastane, kütüphane gibi aynı ortamda bulunan bilgisayarlar arasında bağlantı ve veri transferi sağlamak amacıyla kullanılmaktadır.

İkinci olarak ise Geniş Erişimli Ağlar'dan (Wide Area Network) bahsedebiliriz. Bu tip ağlar fiziksel olarak birbirine uzak bilgisayarları veya yerel erişimli bilgisayar ağlarını birbirine bağlamak için kullanılmaktadır. Geniş erişimli ağların en bilinen ve en büyük örneği internet ağıdır.

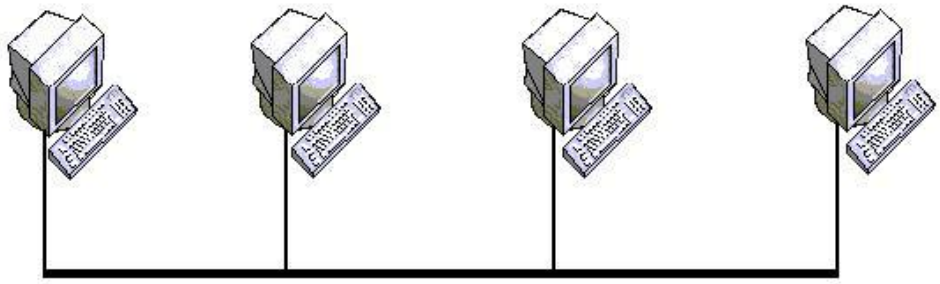
Çeşitli Yerel Erişimli Ağları birbirine bağlamak için ise Ana omurga ağı (Backbone Network) kullanılmaktadır. Ana omurga ağları yüksek kapasiteli hatlardan oluşan ve yüklü miktarda veriyi taşıyabilen ağlardır.

Bilgisayar ağlarında kullanılan her bir bilgisayar veya terminal bir düğüm olarak nitelendirilmektedir. Bu düğüm noktalarına bağlanmış bir veya birden çok bağlantı olabilir. Bu bağlantılar, bilgisayar ağının topolojisini belirler. Literatürde en çok kullanılan bilgisayar topolojileri aşağıda verilmektedir:

1. Doğrusal Ağ Topolojisi (Bus Network)
2. Yıldız Ağ Topolojisi (Star Network)
3. Halka Ağ Topolojisi (Ring Network)
4. Ağaç Ağ Topolojisi (Tree Network)
5. Karmaşık Ağ Topolojisi (Mesh Network)

2.1.1 Doğrusal topoloji

Doğrusal topolojiler tüm düğümler tek bir hat üzerinden birbirlerine bağlandığı ve iki adet bitiş noktasının bulunduğu ağlardır. Genellikle birkaç ağı birbirine birleştiren omurga ağlarında, düşük maliyeti ve kurulum kolaylığı sayesinde de ufak yerel alan ağlarında kullanılmaktadır. Bu tip ağlarda, veri transferinde meydana gelen arızaların kaynağını bulmak çok zordur. Ayrıca ağın genel güvenilirlik düzeyi çok düşük olduğu için güvenilirliğin yüksek olması istenen durumlarda tercih edilmemektedir.

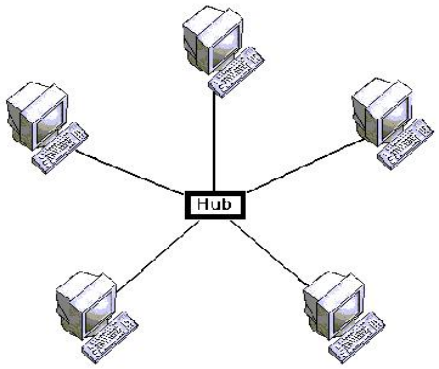


Şekil 2.1 Doğrusal Ağ Topolojisi

2.1.2 Yıldız topoloji

Yıldız ağ topolojisi yerel alan ağlarında ve küçük boyutlu ağlarda kurulum kolaylığı ve pratikliğinden dolayı en çok tercih edilen yapıdır. En basit haliyle, merkezi bir bilgisayar, anahtar veya tekrarlayıcıya bağlı bilgisayarlardan oluşmaktadır. Ağa bağlı çevre cihazlarının bozulması ağın genel performansını etkilememektedir.

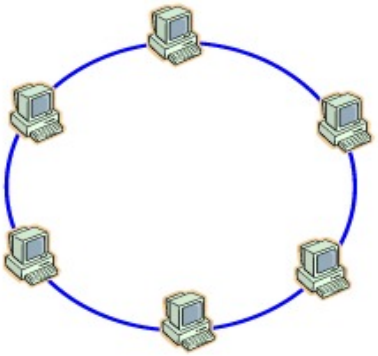
Yıldız ağ topolojisinde merkezi tekrarlayıcıya olan bağımlılık çok fazladır. Bu cihazda meydana gelebilecek her türlü aksaklık tüm ağı etkilemekte ve bütün veri alışverişini durdurabilmektedir. Ağ büyüklüğü ve ağın genel performansı da tekrarlayıcının kapasitesine bağlıdır. Ayrıca yıldız ağ topolojilerinde ağ büyüklüğü arttıkça kablo düzenlemesi de oldukça karmaşıklaşmaktadır.



Şekil 2.2 Yıldız Ağ Topolojisi

2.1.3 Halka topoloji

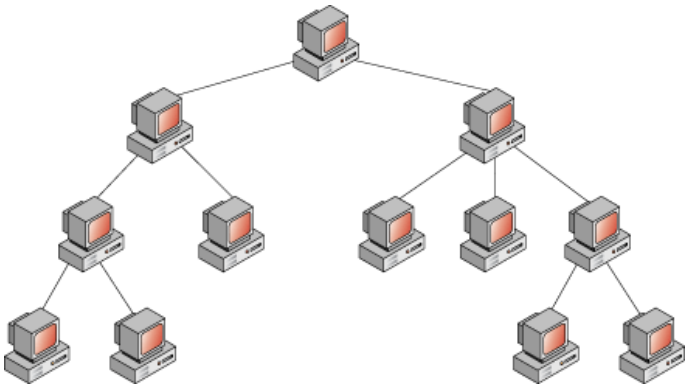
Halka topolojilerinde, ağdaki her bir bilgisayar bir halka oluşturacak bir şekilde diğer iki bilgisayara bağlanmaktadır. Sistem genelde sadece bir hattın bozulmasından etkilenmeden çalışmasına devam edebilmektedir. Halka topolojisinde, merkezi bir sunucuya ihtiyaç duyulmaması, düzenli bir yapısının olması, yıldız ağ topolojisine göre daha büyük veri yüklerini kaldırabilmesi ve büyük boyutlu ağların kurulmasına izin veren bir yapısının bulunması avantaj olarak gösterilebilir.



Şekil 2.3 Halka Ağ Topolojisi

2.1.4 Ağaç topoloji

Birden çok yıldız topolojinin doğrusal bir topoloji üzerinde birleştirilmesi ile oluşan topolojiye ağaç topoloji denir. Sistemin genel güvenilirlik seviyesi düşüktür. Doğrusal ağ topolojisinde olduğu gibi omurga ağda meydana gelen bir kopma tüm ağı etkiler ve çalışmamasına neden olur. Ayrıca diğer ağ yapılarına göre kurulması ve ayarlanması oldukça zordur.

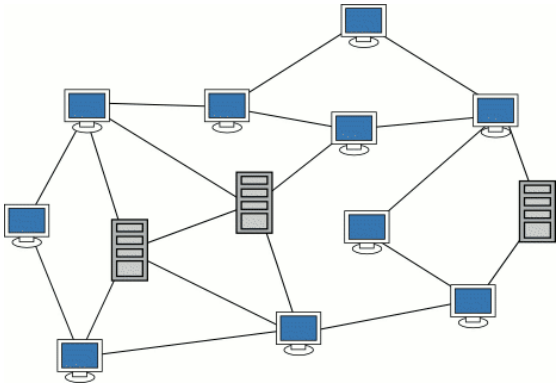


Şekil 2.4 Ağaç Ağ Topolojisi

2.1.5 Karmaşık topoloji

Karmaşık topoloji bir diğer adı ile de Mesh ağı olarak bilinir. Bu topolojilerde ağdaki herhangi iki bilgisayar arasında veri aktarımı yapılacağı zaman, birden çok yol

izlenebilir. Eğer ağdaki her bilgisayar birbirine doğrudan bağlantılı olursa buna tam bağlı ağ denir. Böyle durumdaki ağlarda herhangi bir bilgisayar veya hat bozulduğunda ağda bir kesinti olmaz. Bilgisayarlar arasındaki bağlantı sayısı arttıkça ağın güvenilirliği de artacaktır. Doğal olarak bu durum maliyetleri de arttırmaktadır. Güvenilirliğin önemli olduğu pek çok sistemde karmaşık ağlar kullanılmaktadır.



Şekil 2.5 Karmaşık Ağ Topolojisi

2.2 Ağ Topolojisi Tasarımının Karmaşıklığı

Bir bilgisayar ağı $G = (N, L, p)$ olarak gösterilebilir. Burada N bilgisayarları, L bilgisayarlar arasındaki bağlantıları ve p bu bağlantı hatlarının güvenilirliğini göstermektedir. Topolojinin büyüklüğüne bağlı olarak L, eşitlik 2.1 'deki formüle göre hesaplanır.

$$L = \frac{N(N-1)}{2} \tag{2.1}$$

Çözüm uzayı ise, L değerine bağlı olarak eşitlik 2.2 'deki gibi üstel bir artış sergilemektedir.

$$Ç_{uzayı} = 2^L \tag{2.2}$$

Örnek olarak, yedi bilgisayardan oluşan bir topoloji için L=21 ve $Ç_{uzayı}$=2,10*10^6 olurken 20 bilgisayarlı bir topoloji için L=190 ve $Ç_{uzayı}$=1,56*10^{57} değerlerine ulaşmaktadır.

2.3 Ağ Tasarımında Kullanılan Güvenilirlik Kriteri

Son yıllarda bilgisayar ve haberleşme ağların kullanımı ve önemi, önceki yıllara göre çok daha büyük bir hızla artmaktadır. Yeni ağların tasarımlarında maliyet, performans, üretilen veri hacmi, bağlantı gereksinimleri ve güvenilirlik gibi pek çok kriter rol oynamaktadır [8]. Bu kriterlerin içinde en önemli olanı ise, kurulan ağın çalışmasına devam edebilmesi açısından, güvenilirlik kriteridir. Özellikle omurga ağlarda sistemin sürekliliği oldukça önemlidir.

Ağ büyüklükleri arttıkça ağlardaki bozulmalar ve problemler da paralel bir artış göstermektedir. Bu hataların bir kısmı yazılımsal bir kısmı ise donanımsal hatalardır. Bu tezde, ağlardaki donanımsal bozulmalar altında ağın hayatta kalması ve veri aktarımına devam etmesi incelenmiştir. Ele alınan problemde bilgisayarların bozulmadığı, hatların arızalandığı durum dikkate alınmıştır.

2.3.1 Ağ güvenilirliğinde belirli kriterler

Ağ güvenilirliğinde belirli ölçüler, ağın çalışamaz hale gelmesi için arızalanması veya devre dışı kalması gereken bileşenler ile ilgilenmektedir. Altıparmak [9] tarafından yapılan çalışmada, bağlılık, birleşme, çap, çevre ve ağın birleşme yüzeyi gibi parametreler aşağıdaki gibi açıklanmıştır.

Bağlılık: Bir ağdaki her bir düğüm noktasının ağın tamamına en az bir hat ile bağlı olması anlamına gelmektedir. Yani ağın içerisindeki tüm düğüm noktalarına en az bir hattın bağlı olması anlamına gelmektedir. Bununla birlikte ağdaki tüm düğümlerin birbirine bağlı olması için ağın en az bir yayılan ağacı kapsaması gerekmektedir. Ağın güvenilirliğinin yüksek olması için bağlılık derecesinin de

yüksek olması gerekmektedir. Düğüm çiftleri arasında en az iki hat olması, yani 2-bağlılık, güvenilirliğin yüksek olmasının istendiği durumlarda ve ana şebeke tasarımlarında kullanılmaktadır.

Birleşme: Bir ağdan m düğümlü alt ağlar elde edebilmek için, ağdan çıkartılması gereken hatların yada düğümlerin en küçük sayısıdır. Yüksek güvenilirliğe sahip bir ağda, elde edilen tüm alt ağlar için birleşme değerinin büyük olması gerekmektedir.

Çap: Ağdaki düğüm çiftleri arasındaki en kısa yolların içindeki en uzun yol bize ağın çapını gösterir. Çap değeri ne kadar düşük olursa, ağın güvenilirliği de o kadar yüksektir.

Çevre: Ağdaki başlangıç ve bitiş düğümleri aynı olan yol olarak tanımlanmaktadır. Çevrim değerinin uzunluğu kapsadığı hatların toplam uzunluğuna eşittir. Bir ağın çevrim değeri ise ağdaki çevrim değerlerinin en küçük uzunluğuna eşittir. Çevrim değerinin büyük olması, ağın güvenilirliğinin yüksek olduğunu göstermektedir.

Ağın Birleşme Düzeyi: Bir ağı kendisinden daha küçük en az iki veya daha fazla alt ağa bölmek için çıkartılması gereken en az düğüm veya hat sayısı, ağın düğüm yada hat birleşme düzeyi değerini vermektedir. Birleşme düzeyi değerinin yüksek olması ağın güvenilirlik düzeyinin de yüksek olduğunu göstermektedir.

2.3.2 Ağ güvenilirliğinde olasılıklı kriterler

Olasılıklı ölçüler ile ölçümlenen güvenilirlikte, ağda bulunan parçaların belki olasılıklarla çalışması veya bozulması durumu vardır. Olasılıklı güvenilirliğin hesaplanmasında literatürde üç farklı problem türü vardır. Bunlar iki terminal güvenilirlik problemleri, tüm terminal güvenilirlik problemleri ve k terminal güvenilirlik problemleridir.

İki terminal Güvenilirlik Problemi: Bilgisayar ağındaki bir kaynak s düğümünden hedef t düğümü arasında en az bir işlevsel yol ile bağlılığın olması olasılığının hesaplandığı problemlerdir. ($T = \{s, t\}$)

Tüm Terminal Güvenilirlik Problemleri: Ağdaki her bir v_1, v_2 düğüm ikililerinin arasında en az bir işlevsel yol ile bağlılığın olması olasılığının hesaplandığı problemlerdir. ($T = V$) Bir başka deyişle ağın en az bir yayılan ağacı kapsaması gerekmektedir.

K Terminal Problemleri: Ağdaki K adet belirlenmiş düğüm noktasının ($2 \leq K \leq n$ olmak üzere) tamamını birbirine bağlayan en az bir işlevsel yol ile bağlılığın olması olasılığının hesaplandığı problemlerdir.

2.3.3 Ağ güvenilirliğinin değerlendirilmesi

Ağ güvenilirliğinin değerinin hesaplanması ve değerlendirilebilmesi için literatürde kullanılan üç temel yaklaşım varadır. Bunlar, ağın güvenilirliğinin tam değerin hesaplanması, ağ güvenilirliğinin sınırlarının bulunması ve güvenilirlik değerinin benzetim veya diğer tahmin yöntemleri ile tahmin edilmesidir.

Tam Değer Hesabı: Tasarlanan bir haberleşme veya bilgisayar ağının güvenilirliğinin tam değerinin hesaplanması için literatürde kullanılan çeşitli yöntemler vardır. Bunlara örnek olarak Kesme/Yol kümesinin sayımı metotları ve Durum sayma metotları verilebilir. Ancak tam değerin hesaplanabilmesi için gereken işlem sayısı, ağın büyüklüğüne bağlı olarak üstel bir hızla artmaktadır. Ağdaki hat sayısı L olduğuna göre, ağın güvenirliliğinin tam hesaplanması için 2^L farklı durumun hesaplanmasına ihtiyaç vardır. Bu dezavantajından dolayı, büyük boyutlu problemlerde tercih edilen bir yöntem değildir.

Sınır Hesabı: Bir haberleşme veya bilgisayar ağında güvenilirlik değerlerinin alt ve üst sınır değerlerinin hesaplanması oldukça tercih edilen bir yöntemdir. Literatürde

bu amaçla Jan et al. [10] tarafından üst sınır formülü ortaya atılmıştır. Üst limit yaklaşımı detaylı açıklaması ve bu tezde önerilen yaklaşımda nasıl kullanıldığı bölüm 5.2 ve 5.3.3'de anlatılmaktadır.

Tahmin: Büyük boyutlu bilgisayar ağlarında güvenilirliğin tam değerinin hesaplanamamasından dolayı güvenilirlik değeri tahmin edilmeye çalışılmaktadır. Ağ güvenilirliğinin tahmininde literatürde en çok yararlanılan yöntem Yeh et al [11] tarafından önerilen Monte Carlo Simülasyonu yöntemidir. Bu yöntem üst sınır metoduna göre tam değere daha yakın sonuçlar vermektedir ancak sonuçların hesaplanabilmesi için daha fazla işlem zamanına ihtiyaç duyulmaktadır. Yöntemin algoritması ve bu tezde önerilen yöntem içerisindeki kullanımı bölüm 5.2 ve 5.3.4'de anlatılmaktadır.

3. KARINCA KOLONİSİ ALGORİTMASI

Günümüzde sezgisel yöntemler pek çok gerçek hayat probleminin çözümünde kullanılmaktadır. Özellikle son yıllarda en iyi sonucu bulunamayan problemler için pek çok yeni sezgisel yöntem geliştirilmiş ve uygulanmıştır. Bu sezgisel yöntemlerin amacı, en iyi sonuca olabildiğince yakın sonuçlar bularak problemi çözmektir.

Sezgisel yöntemler genelde doğadaki doğal olaylardan, fizik kurallarından, canlıların evrimleşmesinden, çeşitli biyolojik olaylardan ve hayvan davranışlarından esinlenilerek ortaya çıkarılmaktadır. Bunlara örnek olarak en çok bilinen sezgisel yöntemlerden birkaçı; Tavlama Benzetimi, Tabu Arama, Genetik Algoritma, Kuş Sürüsü Algoritması, Yapay Sinir Ağları, Karınca Kolonisi Algoritması olarak sıralanabilir. Bu sezgisel algoritmalar literatürde bir çok NP-zor kombinatoryal eniyileme problemlerinin çözümünde başarıyla kullanılmıştır.

Pek çok sezgisel algoritma; nöron, kromozom, karınca ve kuş gibi çeşitli aracı veya aracılar kullanmaktadır. Bu algoritmalarda aracılar tek başına veya birlikte kullanılabilmektedir. Amaç ise bir başlangıç çözümünden başlayarak, çözümü geliştirmek ve daha iyi sonuçlara ulaşmaktır. Birden çok aracı kullanıldığında, aracılar arasında yarışmacı-işbirlikçi yaklaşım izlenerek her iterasyonda daha iyi çözümler elde etmeye çalışılmaktadır. Aracılar arasında yarışma ortamı bulunmasına rağmen her bir aracının bulduğu iyi sonuç, diğer aracılar ile paylaşılır ve bu işbirlikçi yaklaşım sayesinde aracıların daha iyi çözümlere yönelmesi sağlanır. Evrimsel yaklaşımda ise, amaç, her iterasyonda, bulunan çözümü daha iyiye taşıyacak özelliklerini kullanarak, yeni çözümler elde etmektir.

Karıncalar sosyal yapıya sahip canlı topluluklarıdır. Kurmuş oldukları dayanışma sistemi sayesinde çok zor problemleri bile hiç zorlanmadan çözebilirler. İyi yapılandırılmış bir sosyal düzenleri vardır ve böylece problemlerini daha küçük problem parçalarına ayırarak kolayca üstesinden gelebilirler. Karınca kolonisi algoritması karıncaların doğal davranışlarından esinlenilerek geliştirilmiştir ve

büyük boyutlu eniyileme problemlerinin çözümünde her geçen gün daha çok kullanılan bir çözüm yöntemidir.

Karıncaların kendi kendilerine organize olarak büyük boyutlu problemleri hızlı ve kolay çözebilme yeteneklerinin bilgisayar ortamına taşınması ile ortaya çıkan karınca kolonisi sezgisel algoritması, ilk kez Marco Dorigo tarafından 1992 yılında doktora tezinde önerilmiştir. Daha sonra Colorni, Maniezzo ve Dorigo'nun ilk incelediği algoritma Karınca Sistemi (KS) algoritmasıdır [12]; [13]. Bu algoritmada diğer karınca kolonisi algoritmalarında olduğu gibi, birden çok aracı karınca paralel olarak çalışmakta ve aralarında basit bir haberleşme kurmaktadırlar. Dorigo, Maniezzo ve Colorni kendi algoritmalarını, ayrık hesaplama, olumlu geri dönüşüm ve yapısal artan sezgisel metotlarının bir kombinasyonu olarak tanımlamaktadırlar [14]; [15]. Karıncaların davranışları bilgisayar ortamında taklit edilmeye çalışılmış ve ilk olarak Gezgin Satıcı Probleminin (GSP) çözümünde kullanılmaya başlanmıştır. Çeşitli araştırmacılar tarafından GSP problemine uygulanan karınca kolonisi algoritması olumlu sonuçlar vermiştir. Colorni, Dorigo ve Maniezzo geliştirdikleri yeni yaklaşımı her türlü kombinatoryal eniyileme probleminin çözümü için önermektedirler [14]; [15].

Dorigo, Maniezzo ve Colorni KS algoritmasını Gezgin Satıcı Problemi çözümünde Tavlama Benzetimi (TB) ve Tabu Arama (TA) algoritmaları ile karşılaştırmışlardır [16]. Karınca Sistemi algoritması ve bu algoritmanın farklı versiyonları literatürdeki pek çok problem türüne uygulanmıştır. Araştırmacılar oldukça olumlu sonuçlar elde etmişler ve bunun sebeplerini karınca sisteminin sürekli iyileşen sonuçlara ulaşması, kötü çözümlerden kaçınması, yerel en iyilere takılmaması ve ilk adımlarda bile iyi sonuçlara ulaşabilmesi olarak göstermektedirler. Literatürde Karınca Kolonisi yöntemi ile yapılan çalışmalar, Krishnaiyer and Cheraghi [17] ve Stutzle and Dorigo [18]'nun çalışmalarından uyarlanan çizelge 3.1'de gösterilmektedir.

Çizelge 3.1 Karınca Kolonisi Yöntemi İle Yapılan Çalışmalar [17]; [18]

Problem Adı	Yazarlar	Yazım Yılı	Kullanılan Algoritmanın Adı
Gezgin Satıcı Problemi	Dorigo, Maniezzo & Colorni	1991	KS (AS)
	Gambardella & Dorigo	1995	Karınca-Q (Ant-Q)
	Dorigo & Gambardella	1996	Karınca-Q (Ant-Q)
	Dorigo & Gambardella	1996	ACS & ACS-3-opt
	Gambardella & Dorigo	1996	KKS (ACS)
	Stützle & Hoos	1997	EEKS (MMAS)
	Bullnheimer, Hartl & Strauss	1997	KSderece (ASrank)
	Bullnheimer, Kotsis & Strauss	1998	KS & Paralelleme (AS & Parallelization)
	Botee & Bonabeau	1998	E-ACO
	Stützle & Dorigo	1999	KS, KKS, EEKS, KarıncaElit, KarıncaDerece, (AS, ACS, MMAS, ANTelite, ANTrank)
	Cordon, ve arkadaşları	2000	BWKS (BWAS)
	Middendorf , Reischle & Schmeck	2002	MCAA
	Montgomery & Randall	2002	KKS (ACS)
Bağlantı-Yönelimli Ağ Rotalama Problemi	Schoonderwoerd, Holland, Bruten & Rothkrantz	1996	ABC
Kablosuz Ağ Rotalama Problemi	Di Caro & Dorigo	1997	KarıncaNet &KarıncaNet-FA (AntNet & AntNet-FA)
	Subramanian, Druschel & Chen	1997	Düzenli Karıncalar (Regular Ants)

	Heusse, Guerin, Snyers & Kuntz	1998	CAF
	van der Put & Rothkrantz	1998	ABC-B
Optik Ağ Rotalama Problemi	Navarro Varela & Sinclair	1999	KKO-VWP (ACO-VWP)
Haberleşme Ağlarında Dinamik Rotalama Problemi	Zhou & Liu	1999	IAA

Çizelge 3.1 devam ediyor

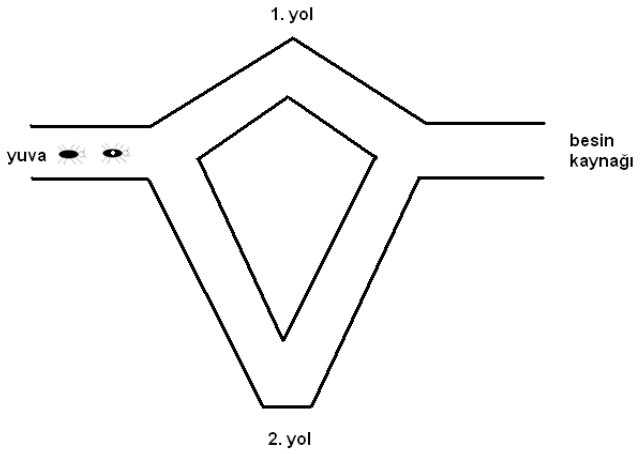
3.1 Karınca Kolonisi Yaklaşımı

Doğal hayatta karıncaların karşılaştığı en büyük problemlerden bir tanesi buldukları besin kaynağını sınırlı sayıda işçi karınca ile yuvalarına taşımaktır. Bu taşıma sırasında zaman da önemli bir faktördür. Zira karıncaların yürüme hızlarının sabit olduğu bilindiğine göre, hızlı bir taşıma yapabilmeleri için en kısa yolu kullanarak besin kaynağındaki besinleri yuvalarına götürmeleri gerekmektedir. Yapılan doğal yaşam gözlemlerinde karıncaların yuvaları ve besin kaynağı arasındaki en kısa yolu kullanarak taşıma yaptıkları ispatlanmıştır. En kısa yolun belirlenmesinde ise karıncalar Feromon adı verilen bir kimyasaldan yararlanmaktadırlar.

Feromon karıncaların yürüdükleri yola bıraktıkları kimyasal bir hormondur. Bu hormon karıncaların yürüdükleri yolu işaretlemelerini sağlar. Böylece karıncalar daha önce bir başka karıncanın yürüdüğü yolu bulabilir ve aynı yoldan gidebilirler.

Karıncalar doğal hayatta tercih ettikleri yolun uzunluğunu daha önceden bilmemektedirler. Bunun için yolda yürüdükleri mesafe boyunca her birim yola eşit

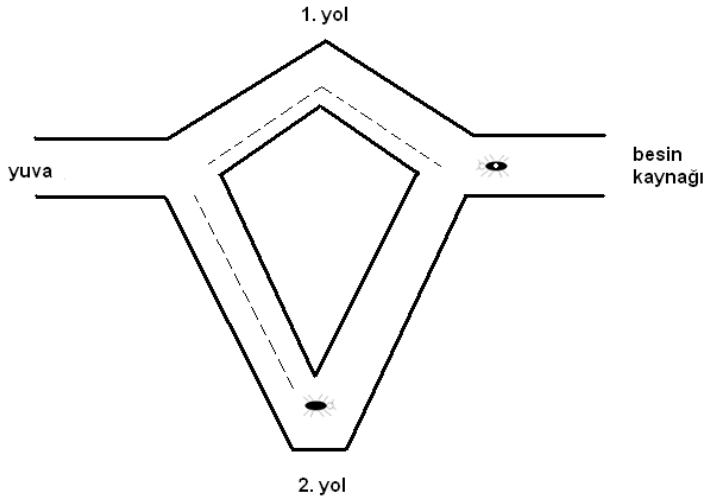
miktarda birim Feromon bırakmaktadırlar. Goss et al. [19] tarafından köprü deneyi karıncaların doğal ortamlarında, her zaman en kısa yolu nasıl seçtiklerini göstermektedir.



Şekil 3.1 Köprü Deneyi Başlangıç Durumu [19]

Karıncalar, Şekil 3.1 deki gibi daha önce hiç görmedikleri bir yol ile karşılaştıklarında rasgele bir şekilde birinci yolu veya ikinci yolu seçmektedirler. Yollarda daha önceden hiç bir feromon izi olmadığından dolayı birinci ve ikinci yolun karıncalar tarafından tercih edilme olasılığı aynıdır. Burada karıncaların yol seçimlerinde yolun uzunluğunun hiç bir önemi yoktur çünkü karıncalar yola başlamadan önce yolun uzunluğunu bilmemektedirler.

Yolların uzunluklarının birbirine oranı olan θ değeri eşitlik 3.1'de verilmiştir. Böyle bir varsayım altında bir numaralı yolu takip eden benekli karınca şekil 3.2'de görüldüğü gibi besin kaynağına ulaştığı anda iki numaralı yolu takip eden beneksiz karınca yolu henüz tamamlayamamış olacaktır.

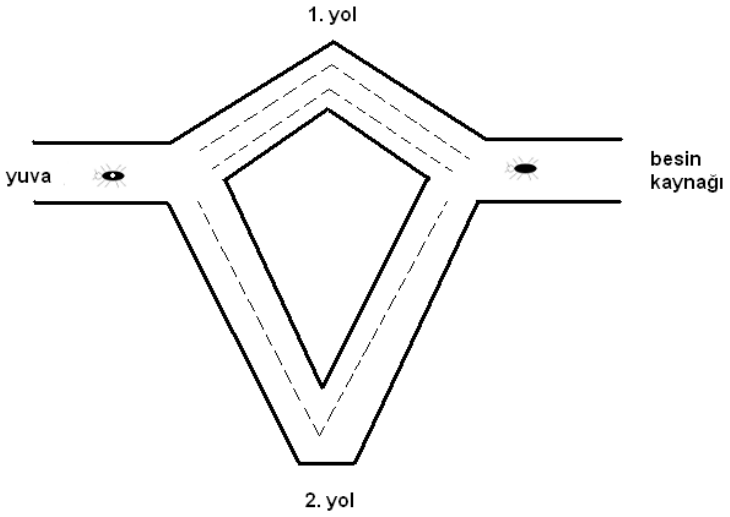


Şekil 3.2 Köprü Deneyi Birinci Adım [19]

$$\theta = \text{yol1/yol2} = 1/2 \qquad (3.1)$$

Şekil 3.2'de görüldüğü gibi, benekli karınca yuvasına geri dönmek istediğinde tekrar yol tercihi yapmak zorunda kalacaktır. Fakat bu sefer birinci yolda bir birimlik feromon düzeyi olduğundan dolayı birinci yolu tercih etme eğilimi içinde olacaktır.

Beneksiz karınca besin kaynağına ulaştığında benekli karınca yuvasına geri dönmüştür. Şekil 3.3'de görüldüğü gibi, beneksiz karınca yuvasına geri dönmek istediğinde tekrar yol tercihi yapmak zorunda kalacaktır. Beneksiz karınca yol tercihini yaparken her iki yoldaki feromon düzeylerine bakacak ve dönüşte bir numaralı yolu tercih etme eğilimi içinde olacaktır. Böylece bir numaralı yoldaki feromon düzeyi artacak ve kolonideki diğer karıncalar da bir numaralı yolu tercih etme eğilimi içinde olacaklardır.



Şekil 3.3 Köprü Deneyi İkinci Adım [19]

Yoldaki feromon düzeyi ne kadar artarsa, karıncaların o yolu seçme eğilimleri de o kadar artmaktadır. Böylece karıncalar yuvaları ile besin kaynakları arasındaki en kısa yolu bulabilmektedirler.

3.2 Karınca Kolonisi Algoritmasının Temel Yapısı

Bölüm 3'de, Çizelge 3.1'de verildiği üzere Karınca Kolonisi Algoritması birçok problemin çözümünde başarı ile uygulanmıştır. Bu çalışmada ele alınan bilgisayar ağlarının tasarımı problemi iki aşamada ele alınarak çözülmüştür. Birinci aşamada; düğümleri birleştiren gezgin satıcı turu oluşturulmaktadır. İkinci aşamada ise turnuva seçim mekanizması ile düğümler arasına hatlar ekleyerek istenilen güvenilirlik düzeyine ulaşılmıştır. Birinci aşamada oluşturulan gezgin satıcı turu iki-bağlılık kısıtını en küçük maliyet ile sağlamaktadır. Çünkü GSP herhangi bir düğümden başlayarak her düğüme en az bir kez uğrayarak başladığı düğüme geri dönen en kısa yolu bulur.

Birinci aşama sonucunda elde dilen en küçük maliyetli iki-bağlı tasarım olan halka topolojisi ile yola çıkılarak daha kısa sürede istenilen R_0 güvenilirliğini sağlayan

topoloji tasarımının elde edilebileceği düşünülmüştür. Bu amaçla ikinci aşamada turnuva seçim kuralı ile rassal seçilen hatların ilavesi ile sonuçta istenilen kısıtları sağlayan tasarıma ulaşılmıştır.

Dorigo, gerçek hayatta ve doğada karıncaların yuvaları ve besin kaynakları arasındaki en kısa yolu bulmaları gerçeğinden hareketle karınca kolonisi algoritmasını geliştirmiştir. Karınca kolonisi algoritmasının oldukça hızlı çalışan bir sezgisel algoritmadır ve büyük boyutlu problemlerde kaliteli sonuçlar vermektedir. Bu gerekçeler dikkate alındığında, bu problemin çözümünde büyük boyutlu topoloji tasarımı için ilk adımda elde edilecek GSP turu için Karınca Algoritması ile başlangıç çözümü oluşturulmuştur.

Gezgin Satıcı Problemi literatürde uzun yıllardır araştırılan ve çözüm aranılan bir NP-zor problemdir. Gezgin Satıcı Probleminde amaç, bir satıcının bulunduğu şehirden başlayıp, her şehre sadece bir kez uğradıktan sonra başladığı şehre geri dönen en kısa turu bulmaktır. Bir başka deyiş ile Gezgin Satıcı Problemi en kısa Hamilton turunu bulmayı amaçlamaktadır. Hamilton turu 19. yüzyıl'da yaşamış bir matematikçi olan William Hamilton tarafından bulunan ve bir çizelge üzerindeki her noktadan sadece bir kez geçerek başladığı noktada biten turdur. Herhangi bir Gezgin Satıcı Problemi $(n-1)!/2$ farklı Hamilton turu içermektedir. Genel olarak GSP, eşitlik (3.2) deki gibi ifade edilebilir.

$$G=(N,A) \tag{3.2}$$

Burada N şehirlerin kümesini, A ise bu şehirleri birbirine bağlayan yolları göstermektedir. Her bir yol $(i, j) \in A$ olarak gösterilir. Burada (i,j) ikilisi i ve j şehirleri arasındaki uzaklığı yani d_{ij}'yi ifade etmektedir ve $i, j \in N$ olmak zorundadır.

Gezgin Satıcı Probleminde kullanılan Karınca Sistemi algoritması için literatürde üç farklı yaklaşım bulunmaktadır. Birinci yaklaşım 1991 yılında Dorigo et al. [14] tarafından önerilen Karınca-Yoğunluğu algoritmasıdır. Bu algoritmada karıncalar her düğüm geçişlerinde feromon güncellemesi yapmaktadırlar.

İkinci yaklaşım Karınca-Niceliği algoritmasıdır ve Colorni, Dorigo ve Maniezzo [12] tarafından önerilmiştir. Bu algoritmada da Karınca-Yoğunluğu algoritmasında olduğu gibi karıncalar her düğüm geçişlerinden sonra feromon güncellemeleri yapılmaktadır.

Üçüncü yaklaşım ise Karınca-Tur Algoritması olarak literatürde yer almaktadır ve Dorigo [20] tarafından 1992 yılında önerilmiştir. Karınca-Tur algoritmasında diğer Karınca Sistemi algoritmalarından farklı olarak karıncaların feromon güncellemeleri bütün karıncalar turlarını tamamladıktan sonra yapılmaktadır.

Günümüzde Karınca Sistemi denildiği zaman kullanılan yaklaşım üçüncü yaklaşım olan Karınca-Tur algoritması yaklaşımıdır. Bunun sebebi Karınca-Tur algoritması yaklaşımı ile birinci ve ikinci yaklaşımların sonuçlarına göre daha iyi sonuçlar elde edilebilmesidir.

Karınca sistemi ile Gezgin Satıcı Probleminin çözümü için Dorigo and Stützle [21] tarafından önerilen algoritmanın kapalı hali şekil 3.4'de verilmektedir. Algoritmanın daha detaylı hali ise bölüm 5'de geliştirilen Bilgisayar Ağı Tasarımı için geliştirilen Melez Karınca Kolonisi Algoritması'nın içerisinde görülebilir.

Adım 1 Başla;
    Feromon ve Sezgisel Matrislerini Oluştur;
    Algoritmanın Başlangıç Değerlerini Belirle;
Adım 2 Çözümleri Oluştur;
    Tekrar;
        Karıncaların Başlangıç Şehirlerini Belirle;
        Karınca Turunu Oluştur;
        Feromon Düzeylerini Güncelle;
    Durma Koşulu Sağlanıncaya Kadar;
Adım 3 Dur.

Şekil 3.4 Karınca Sistemi Algoritması

Karınca sistemi yaklaşımı temel olarak feromon yapısı ve sezgisel matrisi, başlangıç durumunun belirlenmesi, karınca turunun oluşturulması ve daha sonra feromon düzeylerinin güncellenmesi olarak dört ana başlık altında toplanabilir. Bu başlıklar altında Karınca Sistemi Algoritmasının daha detaylı bir incelemesi yapılacaktır.

3.2.1 Feromon matrisi ve sezgisel matrisi

Feromon yapısı Karınca Sistemi'nin ve diğer Karınca Kolonisi Algoritmalarının en önemli özelliklerinden bir tanesidir. Karıncaların düğümler arasında bir seçim yapabilmeleri için gereken olasılık değerinin oluşturulması için feromon matrisi ve sezgisel matrisi olmak üzere iki adet matris kullanılması gerekmektedir. Ayrıca bu matrislerde tutulan bilgiler sayesinde karıncaların evrimsel bir yaklaşım izleyerek sürekli daha iyi sonuçlara ulaşmaları da mümkün olmaktadır.

Feromon matrisinin görevi karıncaların tercih ettikleri hatlarda bıraktıkları feromon düzeyinin tutulmasıdır. Hattan geçen her bir karınca hattın üzerine belli bir miktarda feromon bırakmaktadır. Feromon matrisi τ_{ij} sembolü ile ifade edilir ve i düğümünden j düğümüne giden hattaki feromon düzeyini gösterir.

Olasılık değerinin oluşturulmasında önemli olan bir diğer matris de Sezgisel Matrisidir ve η_{ij} sembolü ile gösterilir. Sezgisel matrisin başlangıç değeri eşitlik (3.3)'de verilmektedir. Burada i düğümü ile j düğümü arasındaki uzaklık ters orantılı olarak kullanılmaktadır.

$$\eta_{ij} = 1/d_{ij} \tag{3.3}$$

Olasılık Matrisi'nin değeri algoritmanın çalışması sırasında değişmez ve hep sabit kalır. Bu matrisin kullanılmasındaki temel amaç, karıncaların seçecekleri yolu belirlerken sadece feromon düzeyine bağlı kalmadan yol uzunluklarını da göz önünde bulundurmalarını sağlamaktır. Sadece feromon matrisi kullanılırsa

karıncalar yerel en iyi değerlere takılabilir ve genel en iyi değeri bulmaları imkansız hale gelebilir.

3.2.2 Başlangıç durumu

Karınca Sisteminde ilk olarak sistemde kaç adet karınca olacağının belirlenmesi gerekmektedir. Dorigo and Stützle [21] karınca sistemi algoritmasında karınca sayısının toplam şehir sayısına eşit olması gerektiğini söylemektedir. Burada şehir sayısını n ve karınca sayısını m olarak alacak olursak; $m = n$ olmaktadır.

Karınca Sisteminde ikinci olarak feromon matrisinin başlangıç değerleri belirlenmektedir. Feromon matrisi başlangıcı için Dorigo and Stützle [21] tarafından önerilen başlangıç değeri $\forall(i, j)$'ler için eşitlik (3.4) 'de verilmektedir. Burada feromon matrisindeki tüm yolların aynı değeri yani τ_0 başlangıç değerini aldığı görülmektedir.

$$\tau_{ij} = \tau_0 = m / C^{nn} \tag{3.4}$$

Burada daha önce bahsedilmeyen C^{nn} değeri ise en yakın komşuluk sezgiseliyle oluşturulan tur değeridir. Eğer feromon matrisi başlangıç değerleri olan τ_0 değeri sıfır olsa idi, feromon düzeylerinin artması uzun zaman aldığından başlangıçta algoritma birkaç kere feromon değerlerinin artması için en iyi sonuçlardan uzak değerler bulunacaktı. Bununla birlikte, karıncalar ilk birkaç tur oluşumunda istenilen çeşitlilikte farklı çözüme ulaşamayacak ve aynı turların gelme ihtimali artacaktır. Feromon matrisine başlangıç olarak eşitlik (3.4) 'deki değerleri atandığında ise Karınca Sistemi algoritmasının çok daha hızlı bir şekilde çözüme ulaştığı ve ilk çözümlerde daha iyi çözümlerden başladığı Dorigo and Stützle [21] tarafından ispatlanmıştır.

Üçüncü olarak belirlenmesi gereken parametre buharlaşma katsayısıdır. Bu katsayı sayesinde tüm karıncalar turlarını tamamladıktan sonra sisteme eklenen feromon değerlerinden belli bir miktar buharlaşma gerçekleşmektedir. Bu

katsayının amacı, algoritmayı yerel en iyi değerlerine takılmaktan kurtarmaktır. Karıncaların sürekli olarak aynı yolu seçmeleri halinde seçilen yollarda feromon birikimi olacak ve yeni alternatif yolların seçilmesi birkaç tur oluşumu sonrasında nerdeyse imkansız hale gelecektir. Feromon buharlaşma katsayısının değeri ρ ile gösterilir. Dorigo and Stützle [21] buharlaşma katsayısını 0.5 olarak önermektedir.

Karınca Sisteminde ve diğer Karınca Kolonisi Algoritmalarında ortak olarak kullanılan bir diğer değer Sezgisel Matris değeridir. Bu matris topolojideki yolların uzunluk bilgisini tutmaktadır ve olasılık karar değerinin hesaplanmasında kullanılmaktadır. Sezgisel matrisin başlangıç değeri her bir hat için hattın uzunluğu ile ters orantılı olarak eşitlik (3.5)'deki gibi Dorigo and Stützle [21] tarafından önerilen haliyle kullanılmaktadır.

$$\eta_{ij} = 1/d_{ij} \tag{3.5}$$

Başlangıç durumunda belirlenmesi gereken diğer değerler ise olasılık karar formülünde kullanılan alfa (α) ve beta (β) değişkenlerinin değerleridir. Bu değerler, olasılık karar değerinin oluşturulmasında önemli bir rol oynamaktadır. α değeri feromon matrisinin olasılık karar değerine olan etkisini, β değeri ise sezgisel matrisin olasılık karar değerine olan etkisini göstermektedir. Bu değerler için Dorigo and Stützle [21] tarafından önerilen değerler kullanılmıştır. Araştırmacılar, alfa değeri için "1" ve beta değeri için iki ile beş değerleri arasında değerler önermektedir. Beta değeri problemin yapısına göre verilen aralıkta değişmektedir.

Son olarak sistemdeki her bir karıncanın hangi şehirden yola çıkacağının belirlenmesi gerekmektedir. Bunun için Dorigo and Stützle [21] rasgele olarak karıncalara başlangıç şehirlerinin atanması gerektiğini söylemektedir.

3.2.3 Karınca turunun oluşturulması

Karınca Sistemi algoritmasının ve karıncaların başlangıç durumlarının belirlenmesinden sonra her bir karınca bir tur oluşturmaya başlayacaktır. Her bir karınca seçeceği bir sonraki şehri belirlemektedir. Karıncalar bu seçim işlemi için eşitlik (3.6) 'de verilen olasılık karar formülünü kullanmaktadırlar.

$$P_{i,j}^{\ell} = \frac{\left[\tau_{ij}\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}{\sum_{l \in N_i^{\ell}} \left[\tau_{il}\right]^{\alpha}\left[\eta_{il}\right]^{\beta}} \quad , \text{ eğer } j \in N_i^{\ell} \qquad (3.6)$$

Burada;

p_{ij}^{ℓ}	ℓ karıncasının i düğümünden j düğümüne geçme olasılığı
τ_{ij}	i ve j düğümleri arasındaki feromon matris değeri
η_{ij}	i ve j düğümleri arasındaki sezgisel matris değeri
α	feromon katsayısı
β	sezgisel katsayısı
N	ağdaki düğümlerin tamamı

Her bir karınca için eşitlik (3.6) 'de verilen olasılık karar formülünü kullanarak daha önce gidilmeyen tüm şehirlerin seçilme olasılık karar değerleri hesaplanır. Bu olasılık karar değerleri kullanılarak Rulet Çemberi seçim mekanizması yardımı ile her bir karınca için seçilecek bir sonraki şehir bulunmuş olur. Seçim işlemi tüm karıncalar için tekrarlanır ve bütün karıncalar aynı anda hareket ederek bir sonraki şehre geçerler. Burada önemli olan noktalardan bir tanesi, bütün karıncaların aynı anda seçilen şehirlere gitmeleridir. Böylece tüm karıncalar turlarını aynı anda tamamlamış olurlar.

3.2.3.1 Rulet çemberi seçim mekanizması

Rulet çemberi seçim mekanizması genetik algoritmada da kullanılan bir seçim mekanizmasıdır. Genetik algoritmada yığın içinden ebeveyn seçimi yapmak için kullanılmaktadır. Zaman karmaşıklığı $O(N^2)$ olarak hesaplanmış olan metot, Goldberg [22] tarafından önerilmiştir. Karınca Sistemi algoritmasının içinde ise karıncaların bulundukları şehirden gidecekleri şehri seçme işleminde kullanılmaktadır.

Karınca Sistemi algoritmasında kullanılan rulet çemberi seçim mekanizmasında çember, şehirlerin olasılık karar değerleri ile orantılı aralıklara bölünür. Burada aralığın genişliği, temsil ettiği şehrin seçilme olasılığını göstermektedir. Çemberdeki aralıkların toplamı ise, karıncanın daha önce uğramadığı tüm şehirlerin olasılık karar değerleri toplamına eşittir. Seçim aşamasında çember çevrilir ve sıfır ile toplam olasılık karar değeri aralığında rasgele bir sayı üretilir. Üretilen sayının düştüğü aralıktaki şehir, karıncanın bir sonraki adımda uğrayacağı şehir olarak belirlenir. Karınca Sistemi içerisinde kullanılan Rulet Çemberi Seçim mekanizmasının algoritması şekil 3.5'de verilmektedir.

Adım 1 Başla
Adım 2 Toplam_olasılık=0
Adım 3 Tüm karıncalar için;
    Eğer karınca [k] daha önce [j] şehrine girmişse
        Seçim_olasılığı = 0
    Değilse
        Seçim_olasılığı = $P_{i,j}^{k}$
        Toplam_olasılık = Toplam_olasılık + Seçim_olasılığı
Adım 4 r = rasgele_sayı [0, Toplam_olasılık]
Adım 5 j = 1
Adım 6 c = Seçim_olasılığı[j]

Sekil 3.5 KS için Rulet Çemberi Algoritması

Adım 7 ($c < r$) Koşulu sağlanıncaya kadar;

$j = j + 1$

$c = c +$ Seçim_olasılığı[j]

Adım 8 Karıncanın seçeceği şehir = j

Adım 9 Dur

Sekil 3.5 devam ediyor

3.2.4 Feromon düzeyi güncellemesi

Bölüm 3.1'de açıklandığı gibi karıncalar seçecekleri yolu belirlemek için aday yoldan kendilerinden önce giden karıncaların bırakmış olduğu feromon değerlerine bakmaktadırlar.

Feromon düzeylerinin güncellenmesi işlemi karınca kolonisi algoritmasının temel yapı taşlarından bir tanesidir. Tüm karınca kolonisi algoritması çeşitlerinde bulunan bu adım, aynı zamanda diğer algoritmalar ile karınca sistemi algoritmasını birbirinden ayıran önemli özelliklerden biridir. Feromon düzeyleri güncellenirken, güncellemenin hangi adımda yapıldığı nasıl güncellendiği de büyük önem taşımaktadır. Karınca sistemi algoritmasında feromon değerleri tüm karıncalar turlarını tamamladıktan sonra güncellenmektedir.

Feromon düzeyinin güncellenmesi işlemi, karıncaların yürüdükleri yollara feromon bırakmaları ve bırakılan bu feromonların buharlaşması olarak iki ana başlık altında toplanabilir.

3.3.4.1 <u>Feromon buharlaşması</u>

Tüm karıncalar turlarını tamamladıktan sonra ilk olarak sistemdeki feromonlar belli oranda buharlaştırılır. Bu buharlaşma işlemi ağda bulunan tüm yollara uygulanır. Buharlaşma için ρ sabit değeri kullanılır. Kullanılan bu sabit değer (0,1)

aralığındadır. Algoritmaya eşitlik (3.7)'de olduğu gibi uyarlanmaktadır. Buharlaşma sabitinin değeri sıfır olursa sistemde buharlaşma olmaz. En çok feromon biriken tur, tüm karıncalar tarafında seçilir ve algoritma bulduğu bu yerel en iyi değerine takılır. Onun için buharlaşma sabitinin değeri sıfırdan büyük olmalıdır. Eğer ρ değeri bire eşit olursa karıncalar tarafından bir önceki tura ait hiç bir feromon değeri hatırlanmayacağından dolayı algoritmanın hafızası silinmiş olur. Böyle bir durumda, feromon değerlerinin güncellenmesine gerek kalmaz ve algoritma sadece en yakın komşuluk turunu bulabilir.

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij}, \qquad \forall (i,j) \in L \tag{3.7}$$

Feromon buharlaşmasının amacı, algoritmada elde edilen kötü turların unutulmasını sağlamak ve bulunan iyi turlarda sürekli feromon birikmesini engellemektir. Burada dikkat edilmesi gereken nokta, buharlaştırma işleminde ne çok fazla ne de çok az feromon buharlaştırmaktır. Dorigo, karınca sisteminde kullanılan buharlaşma sabiti için, yapmış olduğu çalışmalar sonucunda 0,5 değerini önermektedir. Dorigo'ya göre bu değer, hem karıncaların bulduğu kötü tur değerlerinin unutulmasını sağlamakta hem de bulunan iyi tur değerlerinde feromon birikmesini önleyerek karıncaların yeni turlar aramasına fırsat tanımaktadır.

3.3.4.2 Yeni feromon eklenmesi

Feromon buharlaşması işlemi yapıldıktan sonra, yeni karınca turları oluşturulmadan önce, son adım olarak karıncaların yürüdükleri yollara feromon eklemeleri yapılır. Karınca Sistemi algoritmasında karıncaların oluşturdukları turların iyi veya kötü olduğuna bakılmaksızın, tüm karıncaların aynı oranda feromon güncellemesi yapmasına izin verilir. Sisteme feromon ekleme işlemi için (3.8) 'de verilen ifade kullanılmaktadır.

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}, \qquad \forall (i,j) \in L \tag{3.8}$$

Burada $\Delta\tau_{ij}^{k}$ feromon güncellemesi yapacak olan k karıncasının üzerinden geçtiği yola ekleyeceği feromon düzeyini temsil etmektedir. Eşitlik (3.9) ve (3.10) 'da $\Delta\tau_{ij}^{k}$ değerinin nasıl tanımlandığı gösterilmektedir.

$\Delta\tau_{ij}^{k} = 1/C^{k}$, eğer (i,j) yolu T^{k}'ya ait ise (3.9)

$\Delta\tau_{ij}^{k} = 0$, diğer durumlar (3.10)

Yukarıdaki eşitlikte, C^{k}, k karıncasının oluşturduğu tur uzunluğunu göstermektedir. Bu turdaki tüm yollar ise T^{k} ile ifade edilmektedir. Eşitlik (3.9) ve (3.10) 'da görüldüğü gibi, daha iyi bir tur değerine sahip olan karınca sistemdeki yürüdüğü yollara daha fazla feromon bırakabilmektedir. Ayrıca daha çok karınca tarafından tercih edilen yollarda, feromon birikmesi daha fazla olmaktadır. Bu yolların daha sonraki adımlarda oluşturulacak turlarda daha fazla karınca tarafından tercih edilme olasılığı artacaktır.

3.3 GSP İçin Karınca Kolonisi Algoritmaları

Karınca Sistemi algoritması gezgin satıcı problemlerinin çözümünde en çok kullanılan ve oldukça iyi sonuçlar veren bir algoritmadır. Bu problemlerin çözümü için farklı yaklaşımlar da önerilmiştir. Literatürde, gezgin satıcı problemine uygulanabilen dört farklı Karınca Kolonisi algoritması vardır. Bunlar; Elitist Karınca Sistemi (EKS), Derece Tabanlı Karınca Sistemi (DTKS) ve En Büyük En Küçük Karınca Sistemi (EEKS) ve Karınca Kolonisi Sistemi (KKS) olarak isimlendirilmiştir.

Elitist Karınca Sistemi algoritması ilk olarak Dorigo et al. [14] tarafında önerilmiş ve daha sonra yine Dorigo et al. [16] tarafından geliştirilmiştir. Algoritmanın her iterasyonunda en iyi turu yapan karıncan ödüllendirilmektedir. Bu algoritmada feromon güncellemesi sırasında, en iyi tura ait olan hatlardaki feromonlara belli bir ölçüde daha fazla feromon bırakılmasını sağlamaktadır.

Karınca Sistemi için bir diğer önerme ise Derece Tabanlı Karınca Sistemidir. Bu sistem Bullnheimer et al. [23] tarafından 1999 yılında önerilmiştir. Algoritmanın temelinde karıncaların oluşturdukları turları tur uzunluklarına göre karıncaları derecelendirmek ve karıncaların derecelerine göre feromon güncellemelerini sağlamak fikri yatmaktadır. Bu yaklaşıma göre en iyi tur uzunluğunu elde eden karınca sisteme en fazla feromonu bırakırken en kötü tur uzunluğunu bulan karınca sisteme en az feromonu bırakmaktadır.

En Büyük En Küçük Karınca Sistemi, Stützle and Hoos [24] tarafından geliştirilmiş bir algoritmadır. Bu algoritmada her iterasyonun sonunda sadece o iterasyona kadar en iyi dereceyi yapan karıncanın feromon güncellemesi yapmasına izin verilmektedir. Tüm karıncaların aynı yolu takip etmesini önlemek amacıyla ise feromon güncellemeleri belirlenen alt ve üst limitler arasında tutulmaktadır. Feromonların buharlaşması ile beraber karıncalar her iterasyonda daha kısa turlar yapmaya özendirilmektedir.

Yapılan bir diğer çalışmada Dorigo and Gambardella [25] tarafından önerilen Karınca Kolonisi Sistemi (KKS) algoritması Karınca-Q üzerinden oluşturulmuş bir yaklaşımdır. Karınca Kolonisi Sisteminde feromon işlemleri sadece en iyi turu içeren hatlar üzerinden yapılmaktadır. Ayrıca hatlar üzerinde çok fazla feromon birikmesini önlenmesi için de çeşitli kontrol mekanizmaları eklenmiştir.

4. LİTERATÜR İNCELEMESİ

Çalışmanın bu kısmında ilk olarak bilgisayar ağlarının topolojik tasarımı ile ilgili yapılan çalışmalara değinilecektir. Daha sonra ise karınca kolonisi algoritması ile çözülen ağ tasarımı problemleri ile ilgili literatür verilecektir.

4.1 Bilgisayar Ağlarının Topolojik Tasarımı İle İlgili Çalışmalar

Bilgisayar ağlarının yapıları ile ilgili literatürdeki ilk çalışmalardan birisi Boorstyn and Frank [26] tarafından yapılmıştır. Bu çalışmada ağ tasarımı problemlerinden işlemci yerleşimi problemi, terminal atama problemi, en küçük yayılan ağaç probleminin kısıtlandırılmış bir hali olan terminal yerleşim problemi ve dağınık bilgisayar ağlarının topolojik yerleşimi problemi ve ana ağ düğüm yerleşimi problemleri incelenmiştir.

Aggaewal and Suresh [27] bilgisayar ve haberleşme ağlarının güvenilirliğini değerlendirmek için yayılan ağaç temelli bir metot geliştirmişlerdir. Ağ güvenilirliği ve s-t terminal güvenilirliği bütün hatların eşit olasılıkta arızalandıkları varsayımı altında incelenmiş ve örneklenmiştir.

Chopra et al. [28], bilgisayar ağları için, bilgisayarların bulunduğu düğümlerin bilindiği varsayımı ile, maliyet kısıtını dikkate alarak s-t güvenilirliğini en çoklayan yeni bir algoritma sunmuşlardır.

Güvenilirlik kısıtı altında bilgisayar ağlarının topolojik en iyilenmesi için Venetsanopoulos and Singh [29] tarafından geliştirilen algoritmada, güvenilirlik kısıtı için yeni bir ölçme sistemi kullanılmış ve buna dayalı olan sezgisel bir algoritma geliştirilmiştir. Geliştirilen algoritmanın çözüm hızını arttırdığı gösterilmiştir.

Kumar et al. [30], var olan bir bilgisayar ağının genişletilmesi problemini ele almıştır. Var olan bilgisayar ağının güvenilirlik kısıtını bozmadan, maliyet kısıtı altında yeni düğüm ve hatlar eklenmesinde genetik algoritmayı kullanmışlardır. Geliştirilen algoritmanın (Genetic Algorithm Based Computer Network Expansion Methodology - GANE) amaç fonksiyonu üzerinde yapılabilecek değişiklikler ile bu problemin farklı türlerinin çözümü için de kullanılabileceği belirtilmiştir. GANE ile elde edilen sonuçlar birerleme metodu ile elde edilen çözümler ile karşılaştırılmış ve algoritmanın hesaplama zamanı açısından daha iyi olmasına rağmen problem için en iyi sonucu garanti etmemektedir.

Kumar, Pathak, Gupta and Parsaei [31], bilgisayar ağlarının topolojik tasarımı problemine genetik algoritma tabanlı genelleştirilmiş bir algoritma sunmuşlardır. Yapılan çalışmada çap, ortalama uzaklık ve bilgisayarların güvenilirliği parametreleri dikkate alınmıştır. Bu yaklaşım aynı zamanda geniş alan ağlarına da uygulanabilmektedir. Elde edilen sonuçlar ayrıntılı araştırma sonuçları ile karşılaştırılmış ve geçerliliği kanıtlanmıştır.

Costamagna et al. [32], tavlama benzetimi yöntemi kullanılarak haberleşme ağlarının topolojik en iyilenmesi üzerine geliştirdikleri algoritmayı sunmuşlardır. Çalışmanın sonuçları literatürdeki diğer yöntemler ile bulunan sonuçlarla karşılaştırılmış ve önerilen algoritmanın başarılı olduğu gösterilmiştir.

Deeter and Simith [33], topolojik yerleşim problemi için, tüm terminal güvenilirlik kısıtı altında genetik algoritma kullanılmıştır. Her hat için farklı hat güvenilirlikleri olmak üzere, yaklaşık veya tam sistem güvenilirliği hesaplanmış ve literatürdeki test problemleri üzerinde algoritmanın esnek ve etkin çalıştığı gösterilmiştir.

Hahuja [34], bilgisayar ağlarında kapasite ve performans probleminin çözümü üzerine yoğunlaşmıştır. Problemi genetik algoritma kullanılarak, sabit maliyet kısıtını dikkate alarak, en büyük güvenilirlik değeri için çözmüş ve sonuçlar literatürdeki en iyi çözümler ile karşılaştırılmıştır.

Dengiz, Altıparmak and Simith [35], belirli bir güvenilirlik kısıtı altında, en küçük maliyetli bilgisayar ağı topolojisini oluşturmak için genetik algoritmayı kullanmışlardır. Ağ gösterimini 0 -1 şeklinde kodlamışlar ve uygun olmayan çözümler için ceza fonksiyonundan yararlanmışlardır. Bu çalışmada geliştirilen algoritmanın sistem güvenilirliğini daha hızlı hesaplayabilmesi için Monte Carlo benzetimi ile güvenilirlik tahmini yapılmıştır.

Pierre and Legault [36], dağıtık bilgisayar ağlarının topolojik en iyilenmesi için gecikme ve güvenilirlik kısıtları altında çözüm arayan bir algoritma sunmuşlardır. Genetik algoritmaya dayalı bir yöntem geliştirmişler ve bu yöntem ile çözülen orta büyüklükteki problemler için literatürdeki bilinen yöntemlere göre daha iyi sonuçlar elde etmişlerdir.

Pierre and Elgibaoui [37] tarafından yapılan çalışmada, tabu arama metodu kullanılarak, güvenilirlik kısıtı atında en küçük maliyetli bilgisayar ağı tasarımı problemleri için bir algoritma sunulmuştur. Geliştirilen algoritma her iterasyonda mevcut topolojinin komşuluklarını üretmekte ve üretilen bu çözümlerden güvenilirlik kısıtını sağlayan en küçük maliyetli çözüm seçilmektedir. Bu işlem aday topolojinin komşuluğu kalmayıncaya kadar devam etmektedir. Bu yaklaşım genelleştirilmiş yerel arama metodu olarak isimlendirilmiştir. On iki ile otuz düğüm arasındaki büyüklüklerden oluşan test problemleri için elde edilen sonuçlar literatürdeki diğer algoritmalar ile karşılaştırılmıştır. Karşılaştırma sonuçlarına göre önerilen algoritmanın daha iyi sonuçlar verdiği gösterilmiştir.

Costamagna et al [38], tabu arama algoritmasını kullanarak yaptıkları çalışmada çoklayıcı merkezlerinin yerleşimi problemi üzerinde durmuşlardır. Oluşturdukları ağda fiber optik kablo kullanmışlar ve fiber optik kabloların yüksek maliyetinden dolayı yayılan ağaç yapısını tercih etmişlerdir. Seçilen çoklayıcıların durumunu belirtmek üzere ikili (0-1) kodlama sistemi kullanmışlardır. Geliştirdikleri algoritmanın yapısında dinamik tabu listesi, frekans temelli uzun dönemli hafıza ve durdurma koşulu kullanan yazarlar, elde ettikleri sonuçları literatürdeki diğer sonuçlarla karşılaştırmışlar ve çözüm kalitesi açısından gelişme sağlamışlardır.

Gheng [39], ana bilgisayar ağları için toplam hat maliyetini en aza indirmeyi hedefleyen ve sadece bir hat bozulmasına izin verilen bir yaklaşım geliştirmiştir. Genetik algoritma kullanarak geliştirilen yaklaşım ile literatürdeki test problemleri bu çalışmada çözülmüştür.

Deeter and Smith [40], haberleşme ve bilgisayar ağlarının tasarımı problemini için genetik algoritmayı kullanarak, maliyet ve güvenilirlik kısırlarını dikkate alan genelleştirilmiş bir yaklaşım ile çözmüşlerdir. Araştırma sonuçları geliştirilen algoritmanın daha hızlı ve verimli çalıştığını göstermektedir.

Altıparmak, Dengiz and Smith [41], genetik algoritma ile maliyet kısıtı altında güvenilirliğin en büyüklenmesi problemine sezgisel bir yaklaşım uygulamışlardır. Düğüm noktaları ve aralarındaki hatları tamsayı olarak kodlamışlar ve literatürdeki test problemlerini çözülmüşlerdir.

Konak and Smith [42], bilgisayar omurga ağ tasarımı problemi için melez bir genetik algoritma sunmuşlardır. Hat yoğunluğunun da göz önüne alındığı araştırmanın sonuçları geliştirilen algoritmanın literatürdeki diğer çalışmalara göre oldukça etkin olduğunu göstermişlerdir.

Liu and Iwamura [43], genetik algoritma, bağımlı şans çok amaçlı programlama, bağımlı şans amaç programlama yöntemlerini kullanarak çoklu güvenilirlik amaçlı problemler için yeni bir yaklaşım geliştirmişlerdir. Geliştirilen yaklaşımın etkinliği literatürdeki sayısal örnekler üzerinde gösterilmiştir.

Aboelfotoh and Al-Sumait [44], tüm terminal güvenilirlik kısıtını dikkate alarak en küçük maliyet için topolojik yerleşimin bulunmasını amaçlamışlardır. Yapay sinir ağları tabanında geliştirilen Opti-Net algoritması ile yapılan araştırmada literatürdeki test problemleri çözülmüş ve genetik algoritma ile karşılaştırılmıştır. Test sonuçlarına göre özellikle büyük boyutlu problemlerde daha iyi sonuçlar elde edilmiştir.

Fard and Lee [45], var olan bilgisayar ağının güvenilirliğini arttırmak için yeni bir algoritma sunmuşlardır. Düğüm çiftlerinin arasına yeni hat eklenmesi ile oluşan yayılan ağaç sayılarına bakılarak hangi düğüm çifti arasına yeni hat eklenilmesi gerektiği hesaplanmaktadır. Sayısal bir güvenilirlik hesaplaması yapılmasına ihtiyaç duyulmayan bu yöntemde ayrıca ağın derece matrisinden de faydalanılmaktadır.

Koide et al. [46] yaptıkları çalışmalarında, Jan ve arkadaşları tarafından, tüm terminal güvenilirlik kısıtı altındaki topolojik ağların en iyilenmesi problemi için önerilen algoritmayı geliştirmişlerdir. Algoritmalarını, farklı hat olasılıklarına sahip topolojik problemleri çözebilmek için uygun hale getirmişlerdir. Ayrıca algoritmaya çeşitli eklemeler yaparak hızlandırmışlar ve daha kaliteli çözümler elde ettiklerini belirtmişlerdir.

Kumar, Parida and Gupta [47], haberleşme ağlarının topolojik tasarımı için genetik algoritma tabanlı ve çok ölçütlü bir model tasarlamışlardır. Birbirlerine en az bir ölçüte göre baskın gelemeyen çözümlerden faydalanılmış ve Pareto Yakınsak Genetik Algoritma ile çözüm aramışlardır. Elde edilen test sonuçları diğer sezgisel yöntemler ile elde edilen sonuçlarla karşılaştırılmıştır.

Srivaree-ratana, Konak and Smith [48], tüm terminal ağ güvenilirliğinin tahmini için yapay sinir ağları yöntemini kullanmışlardır. Geliştirilen yöntem için gereken eğitim sürecinde üst limit yöntemi kullanılmıştır. Yapay sinir ağları ile geliştirilen algoritma, literatürdeki diğer yöntemler ile karşılaştırılmış ve işlem zamanı açısından çok büyük kazançlar sağlanmıştır.

Altıparmak, Dengiz and Smith [49], maliyet kısıtını dikkate alarak ağ güvenilirliğinin en büyüklenmesi problemi için bir sezgisel geliştirerek literatürdeki önerilen yöntemler ile genel bir karşılaştırma yapmışlardır. Tepe tırmanma, tavlama benzetimi, genetik algoritma ve bu algoritmanın karma bir versiyonu olan memetik algoritmayı bilinen test problemlerinin çözümünde kullanmışlardır. Elde ettikleri

sonuçlara göre memetik algoritmanın performansı diğer algoritmalara göre daha iyi olduğunu belirtmişlerdir.

Mandal et al. [50], gerçek hayat kısıtlarına yakın güvenilirlik kısıtları kullanarak omurga ağ tasarımı üzerinde durmuşlardır. Ağdaki herhangi bir hat arızalandığında da ağın çalışmaya devam etmesi kısıtı altında RAS algoritmasını kullanarak, toplam maliyeti en küçüklemeye çalışmışlardır. Araştırma sonuçlarını genetik algoritma ile bulunan sonuçlarla karşılaştırmışlar ve geliştirdikleri algoritma ile daha kaliteli sonuçlar elde etmişlerdir.

Altıparmak, Dengiz and Smith [51], tüm terminal bilgisayar ağlarının güvenilirliğini tahmin etmek için yapay sinir ağları yöntemini kullanmışlardır. Çalışmada homojen ve heterojen hat güvenilirlikleri de dikkate alınmıştır. Yapay sinir ağlarının eğitim sürecinde rassal ve deneysel tasarım yöntemleri kullanılmış ve denesel tasarım ile elde edilen eğitim setinin daha iyi sonuçlar verdiği belirtilmiştir.

Altıparmak, Gen, Dengiz and Smith [52], güvenilirlik kısıtı altında bilgisayar ağlarının topolojik en iyilenmesi için ağ tabanlı ve bulanık mantık kontrollü bir genetik algoritma yapısı (flc-NB GA) sunmuşlardır. Araştırmada Prüfer sayı tabanlı bir kodlama yöntemi kullanılmış, iki noktalı çaprazlama yapılmış ve mutasyon işlemi için yerel arama yöntemi kullanılmıştır. Geliştirilen algoritmanın bulduğu sonuçlar, dal sınır algoritması, genetik algoritma tabanlı ikilik gösterim ve ağ tabanlı genetik algoritma sonuçları ile karşılaştırılmıştır.

Shao et al. [53], maliyet kısıtını dikkate alarak, dağıtımlı erişime sahip ağların güvenilirliğini en iyilemeye çalışmışlardır. Öncelikle maliyet kısıtlı güvenilirlik problemi kombinatoryal ağaç arama işlemi olarak ele alınmıştır. Daha sonra bu yöntemin çok fazla işlem gücü ve zaman gerektirmesi sebebi ile daha hızlı bir yöntem olan daraltma ve arama algoritması uygulanmıştır. Geliştirilen algoritmanın etkinliği simülasyon ve örnek olay incelemeleri ile test edilmiştir.

Xiong and Gong [54], ağ güvenilirliğinin tahmini için oransal yaklaşım algoritması kullanılmıştır. Geliştirilen algoritma tüm terminal sistemlere rahatlıkla uygulanabildiği ve güvenilirlik fonksiyonu eğrisinin sağlandığı gösterilmiştir.

Reichelt, Gmilkowsky and Linser [55], tekrarlı yerel arama metodunu kullanarak ağ topolojilerinin en iyilenmesi problemini çözmüşlerdir. Burada, yerel arama algoritmasının geliştirilmiş bir versiyonu olan tekrarlı yerel arama algoritmasında daha düşük maliyetli ve güvenilirlik kısıtını bozmayan çözümler aranmaktadır. Araştırma sonuçları genetik algoritma yöntemi ile karşılaştırılmış ve önerilen algoritmanın daha iyi olduğu göstermektedir.

Marseguerra et al. [56], genetik algoritma ve Monte Carlo simülasyonu bir arada kullanılarak, bilgisayar ağlarının topolojik tasarımı için çok amaçlı bir eniyileme modeli geliştirilmiştir. Güvenilirlik tahmini için hem bağlantıların hem de düğüm noktalarının güvenilirlik düzeylerinden faydalanmışlardır. Birbirine en az bir kritere göre baskın gelemeyen çözüm kümeleri arasından, karar vericiler yardımı ile, çözümlerin risk profillerine göre seçim yapılmıştır. Önerilen bu algoritma ile farklı tasarımlara ulaşmak mümkün olmuştur.

Gen, Kumar, Kim [57], bilgisayar ağlarının tasarımı problemi için yayılan ağaç tabanlı melez bir genetik algoritma uygulamışlardır. Önerilen yöntem derece-kısıtlı en küçük yayılan ağaç problemleri, yetkilendirilmiş en küçük yayılan ağaç problemleri, sabit yüklü taşıma problemleri ve yerel alan ağ tasarımı problemlerine uygulanabilmektedir. Araştırma sonuçları, genetik algoritma yaklaşımının bu alanda önemli bir potansiyele sahip olduğunu göstermektedir.

Reichelt and Rothlauf [58] haberleşme ağı tasarımı problemini güvenilirlik kısıtını dikkate alarak en küçük maliyet için çözmüştür. Bu çalışmada evrimsel yöntemlere dayalı iki yeni algoritma sunulmuştur. Bulunan uygun olmayan çözümler için düzeltme algoritması uygulanmıştır. Uygun olmayan çözümlere ceza yöntemi uygulayan evrimsel algoritmalar ile önerilen algoritmalar karşılaştırılmış ve önerilen algoritmaların daha iyi sonuçlar verdiğini gösterilmiştir.

Konak and Bartolacci [59] bilgisayar ağlarının tasarımının en iyilenmesi problemi için güvenilirlik kısıtı yerine ağ esnekliği kısıtını dikkate alarak en küçük maliyetli tasarımı bulmayı amaçlamışlar ve bunun için karma genetik algoritma kullanmışlardır. Ağ esnekliğinin tahmin edilmesi için ağdaki trafik durumundan faydalanan bir yöntem geliştirmişlerdir. Kullanılan karma genetik algoritmada, özelleştirilmiş yerel arama operatörleri ve basitleştirilmiş uygun ceza fonksiyonları kullanılmış, oldukça etkin sonuçlar sunmuşlardır.

Cancela and Petingi [60], bilgisayar ağlarının topolojik tasarımı probleminde güvenilirlik kısıtı yerine düğümler arası direkt uzaklığı, yani çapı dikkate almışlardır. K-terminal güvenilirliğin yerine tasarladıkları çap kısıtlı güvenilirlik yaklaşımını baskınlık durumu ile birlikte değerlendirmiştir

Khan and Engelbrecht [61], yerel bilgisayar ağlarının topolojik tasarımı için birbirleriyle ters orantı içerisinde olan maliyet, güvenilirlik, ağ gecikmesi ve kaynak ile hedef arasındaki üst üste binen yollar gibi amaçları bir araya getiren yeni bir bulanık operatörü geliştirmişlerdir. Tasarladıkları operatörü literatürde bilinen diğer bir bulanık operatörü ile karşılaştırmışlar ve önerdikleri operatörün performansının daha iyi olduğunu göstermişlerdir.

Lucio at al. [62], bilgisayar ve haberleşme ağlarının topolojik tasarımı problemini, ağdaki yoğunluğu temel alarak yeni bir yöntem ile çözmüşlerdir. Tepe tırmanma metodunu geliştirerek hızlı yerel arama (FLS) ve yerel en küçük değerlerden kurtulabilen yönlendirilmiş yerel arama (GLS) yöntemlerini geliştirmişlerdir. Ayrıca bu iki yöntemi birleştirmişlerdir. Oluşturdukları yeni algoritmayı genetik algoritma ve tavlama benzetimi algoritması ile karşılaştırmışlar, önerdikleri algoritmanın daha az parametre ile daha iyi sonuçlar verdiğini göstermişlerdir.

Hui [63], büyük boyutlu ağ en iyilenmesi problemleri için kullanılan güvenilirlik tahmini yöntemlerinin çok fazla işlem zamanı gerektirmesinden dolayı tüm ağın güvenilirliğine ihtiyaç duymadan güvenilirlik derecesini tahmin eden bir yöntem

geliştirmiştir. Yazar tarafından önerilen algoritma, diğer yöntemlere göre 30.000 kat daha hızlı sonuçlar verebilmektedir.

Marqueza and Rocco [64], tüm terminal güvenilirlik kısıtı altında en küçük maliyetli amaç fonksiyonuna sahip bilgisayar ağlarının topolojik en iyilenmesi problemi için evrimsel melez bir algoritma kullanmıştır. Olasılıklı çözümler ve monte Carlo simülasyonundan faydalanmışlar ve geliştirdikleri algoritmayı literatürdeki test problemleri ile test etmişlerdir. En iyi çözümü bilinmeyen problemler için, literatürdeki bulunabilen en iyi değerlere göre %7 ile %21 arasında bir iyileşme kaydetmişlerdir.

Sem and Malhotra [65], yirmi düğüme sahip yada daha küçük boyutlu topolojik ağ tasarımı problemlerini genetik algoritma tabanlı yeni bir yöntem ile çözmüştür. Tüm terminal güvenilirlik kısıtı altında en küçük maliyetli ağ tasarımı problemini dikkate almışlar ve önerdikleri algoritmayı literatürdeki diğer genetik algoritma tabanlı yaklaşımların işlem zamanları ile karşılaştırmışlar ve daha verimli sonuçlar elde etmişlerdir.

4.2 Bilgisayar Ağlarının Topolojik Tasarımında Karınca Kolonisi Algoritması'nı Kullanan Çalışmalar

Premprayoon and Wardkein [66], haberleşme ve bilgisayar ağlarının tasarımını güvenilirlik kısıtını dikkate alarak en küçük maliyet için çözmüştür. Araştırmacılar önerdikleri karınca kolonisi algoritmasını tabu arama algoritması ve yerel arama ile karşılaştırmışlardır. Zaman ve elde edilen sonuçlar açısından daha verimli sonuçlar elde etmişlerdir.

Qoubutra, Premprayoon and Wardkein [67], bilgisayar ağlarının topolojik tasarımı problemini, güvenilirlik ve maliyet kriterlerini dikkate alarak geliştirilmiş bir karınca kolonisi eniyilemesi yaklaşımı ile çözmüşlerdir. Birbirleri ile işbirliği içindeki karıncalar ile paralel arama yaptırılmıştır. Elde edilen sonuçlar yerel arama ve tabu

arama algoritmaları ile karşılaştırılmış, önerilen algoritmanın daha etkin olduğu görülmüştür.

Dengiz, Altıparmak and Belgin [68], ağ tasarımı problemi için istenilen güvenilirlik düzeyi kısıtı altında en küçük maliyeti amaçlamışlardır. Bunun için Tavlama Benzetimi ve Karınca Kolonisi Eniyilemesi yöntemlerini bir arada kullanarak yeni bir melez karınca kolonisi algoritması (h_ACO) geliştirmişlerdir. Önerdikleri algoritmayı iki farklı genetik algoritma yaklaşımı ile karşılaştırmışlardır. Araştırma sonuçları Melez Karınca Kolonisi Eniyilemesi Algoritmasının oldukça verimli ve gelecek çalışmalarda kullanılabilecek bir yöntem olduğunu kanıtlamaktadır.

5. BİLGİSAYAR AĞI TASARIMI İÇİN MELEZ KARINCA KOLONİSİ ALGORİTMASI

Bu bölümde, güvenilirlik kısıtı altında en küçük maliyetli bilgisayar ağı tasarımı problemi için bu çalışmada geliştirilen Melez Karınca Kolonisi Algoritması ve bu algoritmanın özelliklerinden bahsedilmektedir. Öncelikle problemin tanımı ve kullanılan varsayımlar hakkında bilgi verilmekte, daha sonra geliştirilen algoritma ve çalışma yapısı anlatılmaktadır. Ardından, bu algoritma içerisinde kullanılan mekanizmalardan bahsedilmektedir. Son olarak ise algoritmanın sağladığı avantajlar anlatılmaktadır.

5.1 Problemin Tanımı ve Kullanılan Varsayımlar

Bilgisayar ağları, n tane düğüm içeren V düğümler kümesi ile ℓ tane hat içeren E hatlar kümesinden oluşan olasılıklı bir ağdır ve G = (V, E) eşitliği ile gösterilebilir. Düğümler ağdaki bilgisayarların bulundukları noktaları ve hatlar ise bu bilgisayarlar arasındaki iki yada tek yönlü hatları ifade etmektedir.

Bu çalışmada, iki yönlü hatların kullanıldığı bir bilgisayar ağının güvenilirlik kısıtı altında en küçük maliyetli topolojik tasarımı ile ilgilenilmektedir. Problemin matematiksel modeli eşitlik (5.1)'de görülmektedir. Ayrıca bu modeldeki her çözümün en azından 2-bağlı bir ağ olması dikkate alınan bir ön kısıttır. 2-bağlı ağlarda, her bir düğüm ağa en az iki farklı yol ile bağlanmaktadır. Böylece bilgisayar ağındaki her bir düğüm noktasından en az iki adet bağlantı hattı geçmektedir.

En Küçük $\quad f(x) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} x_{ij}$

Kısıt $\quad \mathrm{Re}\, l(x) \geq \mathrm{Re}\, l_0 \qquad (5.1)$

Burada;

$$x(i,j)\begin{cases}1 & \text{(i,j) düğüm çifti arasında hat varsa}\\ 0 & \text{diğer durumlar}\end{cases}$$

x : x_{ij} hatlarından oluşan aday ağı

$f(x)$: x aday ağının toplam maliyetini

c_{ij} : (i,j) düğüm çifti arasındaki hattın toplam maliyetini

$\mathrm{Re}l(x)$: x aday ağının güvenilirlik değerini

$\mathrm{Re}l_0$: istenilen güvenilirlik kısıtını göstermektedir.

Bu problemde dikkate alınan varsayımlar ise şunlardır;

1. Ağdaki bütün düğümlerin yeri bilinmektedir ve düğüm arızaları dikkate alınmamaktadır.
2. Ağdaki bütün c_{ij} değerleri bilinmektedir.
3. Ağdaki bir hattın çalışma olasılığı p'dir ve her hat eşit çalışma olasılığına sahiptir.
4. Her hat iki yönlüdür.
5. Hat arızalanmaları birbirinden istatistiksel olarak bağımsızdır.
6. Tüm düğüm çiftleri arasında birbirlerine doğrudan bağlantı yapılabilmektedir.
7. Tüm - Terminal güvenilirliği dikkate alınmıştır.

5.2 Geliştirilen Algoritma

Bu bölümde bilgisayar ağlarının tasarımı için önerilen Melez Karınca Kolonisi Algoritması incelenmektedir. Geliştirilen algoritma temel olarak beş adım halinde özetlenebilir. Bu adımlar sırası ile, başlangıç değerleri, karınca sistemi ile bir GSP başlangıç çözümünün oluşturulması (burada Dorigo [13] kullanılmaktadır), bulunan en iyi GSP çözümün seçilmesi, topolojide istenilen güvenilirlik düzeyinin

sağlanması ve son adım olarak da algoritmanın sonlandırılmasıdır. Algoritma yapısı ayrıntılı olarak şekil 5.1'de verilmektedir.

Adım 1 Başla;

Feromon ve sezgisel matrislerine başlangıç değerlerini ata;

$\eta_{ij} = 1/d_{ij}$, $\tau_{ij} = \tau_0 = m/C^{nn}$;

T = 0;

Adım 2 Karınca Sistemi ile başlangıç GSP çözümün oluştur;

$T \leftarrow 1$;

tekrar;

$s \leftarrow 1$;

tekrar;

$\ell \leftarrow 1$;

tekrar;

p_{ij}^{ℓ} olasılık değerini hesapla;

Rulet çemberi algoritmasını kullanarak karıncanın seçeceği bir sonraki düğümü bul;

$\ell = \ell + 1$;

kadar (ℓ =m);

Feromon buharlaşmasını sağla;

Karıncaların sisteme Feromon eklemesini sağla;

$s = s + 1$;

kadar ($s = N$);

C^{ℓ} değerini hesapla;

$T = T + 1$;

kadar ($T = T^k$) veya (C^{ℓ} =GSP en iyi değer);

Adım 3 Bulunan en iyi çözüme (G_{aday}) adım 4'ü uygula;

Şekil 5.1 Melez Karınca Kolonisi Algoritması, Dorigo [21]

Adım 4 Topolojide istenilen sistem güvenilirlik seviyesini (R_0) sağla;

$T \leftarrow 1$;

tekrar;

$\ell \leftarrow 1$;

tekrar;

Aday ağ topolojisinin güvenilirliğini Jan'ın üst sınır yaklaşımına göre hesapla (R(G_{aday}^{ℓ}));

eğer R(G_{aday}^{ℓ}) < R_0) ise;

tekrar;

Turnuva seçimi mekanizmasıyla topolojide Olmayan Hatlardan η_{ij} değerine göre en iyi hattı ekle;

Yeni aday çözümlerin güvenilirliğini Jan'ın üst sınır yaklaşımına göre hesapla (R(G_{aday}^{ℓ}));

kadar (R(G_{aday}^{ℓ}) $\geq$ R_0);

Eğer bulunan ℓ . çözümün amaç fonksiyonu değeri ($f(G_{aday}^{\ell})$) daha önce bulunan $f(G_{aday}^{\ell})$ değerlerden daha küçük ise MC simülasyonu ile ağın güvenilirliğini ($R(G_{aday}^{\ell})_{MC}$) tahmin et;

eğer (R(G_{aday}^{ℓ}) < $R(G_{aday}^{\ell})_{MC}$) ise;

tekrar;

Turnuva seçimi mekanizmasıyla topolojide olmayan Hatlardan η_{ij} değerine göre en iyi hattı ekle;

Yeni aday çözümün güvenilirliğini MC ile tahmin et ($R(G_{aday}^{\ell})_{MC}$);

kadar ($R(G_{aday}^{\ell})_{MC}$) $\geq$ R_0);

Şekil 5.1 devam ediyor

Eğer aday topolojinin amaç fonksiyonu değeri ($f(G^{\ell}_{aday})$) daha önce bulunan değerlerden daha küçük ise;

$G_X = G^{\ell}_{aday}$;

$\ell = \ell + 1$;

kadar (ℓ =m);

$T = T + 1$;

kadar ($T = T^{\ell}$) veya (G_X =en iyi);

Adım 5 Dur.

Şekil 5.1 devam ediyor

5.3 Algoritma'nın Bileşenleri

Melez Karınca Kolonisi algoritmasında, oluşturulan başlangıç değerleri ve kullanılan diğer yardımcı yöntem ve metotlar aşağıdaki başlıklar altında toplanabilir;

5.3.1 Algoritmanın başlangıç değerleri

Melez Karınca Kolonisi Algoritmasının ilk bölümünde kullanılan Karınca Sistemi algoritması için başlangıç değerleri, Dorigo'nun önerdiği algoritmaya ve değerlere sadık kalınarak alınmıştır. Burada kullanılan; sezgisel matrisi, feromon matrisi, α ve β katsayıları, buharlaşma değeri olan ρ değerleri bölüm 3'de verilmektedir.

Karınca Sistemi meta sezgiselinde, diğer sezgisellerde olduğu gibi tekrar sayısının belirlenmesi gerekmektedir (T^k). Tekrar sayısı attıkça algoritmanın bulduğu sonuçlarda da bir iyileşme gözlenmektedir. Bu çalışmada, kolay karşılaştırma yapılabilmesi ve algoritmanın iyi sonuçlara ulaşabilmesi için, algoritmanın her iki aşaması için de 100 tekrar yapılmaktadır.

5.3.2 Turnuva seçim mekanizması

Turnuva seçim mekanizması, özellikle genetik algoritma içerisinde uygunluk fonksiyonu yüksel olan bireylerin seçilerek bir sonraki nesle aktarılmasında kullanılmaktadır. Bu çalışmada ise, oluşturulan başlangıç ağ tasarımının istenilen güvenilirlik düzeyine çıkartılabilmesi için, eklenecek hatların seçilmesi işleminde kullanılmaktadır. Turnuva seçim mekanizmasının algoritması şekil 5.2'de verilmektedir.

Adım 1 Başla;
Adım 2 Ağ tasarımına eklenmemiş hatlardan n tane seç;
Adım 3 Seçilen hatlardan en küçük maliyetli bir hattı tasarıma ekle
Adım 4 Dur.

Şekil 5.2 Turnuva Seçim Mekanizması'nın Algoritması

5.3.3 Güvenilirlik hesabında üst sınır yaklaşımı

Oluşturulan bilgisayar ağının güvenilirliğinin belirlenmesi için eşitlik (5.2)'de verilen Jan'ın üst sınır yaklaşımı [10] tercih edilmiştir. Bunun sebebi, ağın gerçek tüm terminal güvenilirliğinin hesaplanmasının NP-zor bir problem olması ve problem büyüklüğüne göre çözüm zamanının üstel olarak artmasıdır. Ayrıca üst limit yaklaşımı işlem zamanı açısından hızlı çalışan bir yapıya sahip olduğu için, büyük boyutlu problemlerde algoritmanın genelini yavaşlatmamaktadır.

$$R(G) \le 1 - \left\{ \sum_{i=1}^{n} q^{d_i} \times \prod_{k=1}^{m_i} \left(1 - q^{d_k - 1}\right) \times \prod_{k=m_i+1}^{i-1} \left(1 - q^{d_k}\right) \right\} \qquad \text{Eşitlik (5.2)}$$

Burada;

$R(G)$: Ağın güvenilirliği;

n : düğüm sayısı;

m_i : en küçük (d_i, i-1), tüm i değerleri için;

q : ağdaki hatların bozulma olasılığı;

d_i : i. düğümün bağlılığının derecesi.

5.3.4 Monte Carlo benzetimi

Oluşturulan bilgisayar ağının güvenilirliğinin tahmin edilmesinde kullanılan bir diğer metot da Yeh et al. [11] tarafından önerilen Monte Carlo simülasyonudur. Bu simülasyon yönteminin kullanılmasındaki temel sebep, üst limit ile tahmin edilen ağ güvenilirliği için daha kesin bir sonuç elde edebilmektir. Monte Carlo simülasyonu üst limit yaklaşımına göre işlem zamanı açısından daha yavaş çalışan bir yapıya sahiptir. Monte Carlo Benzetimi algoritması şekil 5.3'de verilmektedir.

Adım 1 Başla;

Adım 2 l_k=0, $\forall\ k\ \in$ 0,1,2,..... L;

Adım 3 j=1'den n'e kadar;

i=1'den L'ye kadar;

(0,1) arasında rasgele bir sayı üret (u_i);

Eğer $u_i > p$ ise $x_i = x_i + 1$;

Değilse x_i=0;

Eğer bozulan hatların sayısı k ise ($x_i = k$);

$l_k = l_k + 1$;

Adım 4 Ağdaki bozulan hatların dağılımını hesapla;

j=1'den n'e kadar;

$P[l = k] = l_k / n$, $\forall\ k\ \in$ 0,1,2,..... L;

Şekil 5.3 Monte Carlo Benzetimi Algoritması

Adım 5 j =1'den n'e kadar;

k =0'dan L'ye kadar;

Rasgele olarak k sayıda hat seçilir ve sistemden çıkartılır;

Ağın bağlı olup olmadığı kontrol edilir;

Eğer ağ bağlı ise; β_k = 1 değilse β_k = 0;

Güvenilirlik tahmin edicisini hesapla;.

$$\hat{R}_j(G) = \sum_{k=0}^{L} \beta_k P[l = k];$$

Adım 6 Sonuç güvenilirlik tahminini hesapla;

$$\hat{R}(G) = \sum_{j=1}^{n} \hat{R}_j(G)/n;$$

Adım 7 Dur.

Şekil 5.3 devam ediyor

Burada;

n	: Algoritmanın çalışma sayısı (n =3000)
p	: Ağdaki hatların çalışma olasılığı
L	: Ağdaki toplam hat sayısı
$\hat{R}(G)$	: $R(G)$ için güvenilirlik tahmin edicisi
$\hat{R}_j(G)$	: j . tekrar için güvenilirlik tahmini; $j \in$ 1,2,3,..... n
l	: Toplam bozulan hat sayısı
$P[l = k]$	: Bozulan hatların olasılıklı dağılımı (k =0,..... L)

5.3.4.1 Bağlılığın kontrol edilmesi

Şekil 5.2'de verilen Monte Carlo simülasyonu algoritmasında beşinci adımda ağın bağlılığının kontrol edilmesi gerekmektedir. Bunun için Hopcroft and Ullman [69]'ın set birleştirme algoritması kullanılmıştır ve algoritma şekil 5.4'de verilmektedir.

Adım 1 Başla;

Adım 2 Her düğüm bir küme oluşturacak şekilde, düğüm sayısı kadar küme elde et ;

Adım 3 Ağdaki tüm hatlar için;

Şebekeden bir hat seç. Seçilen hattın düğümlerinin aynı kümede olup olmadığına bak;

Eğer aynı kümede iseler; ağdan yeni bir hat seç;

Eğer aynı kümede değilseler; düğümleri içeren kümeleri birleştir;

Adım 4 Tüm hatlar seçildikten sonra, oluşturulan yeni kümedeki eleman sayısına bak;

Eğer kümedeki eleman sayısı, ağdaki düğüm sayısına eşit ise, ağ bağlıdır;

Eğer kümedeki eleman sayısı, ağdaki düğüm sayısına eşit değil ise ağ bağlı değildir;

Adım 5 Dur.

Şekil 5.4 Set Birleştirme Algoritması

5.4 Algoritmanın Temel Yapısı

Bilgisayar ağları için geliştirilen Melez Karınca Kolonisi Algoritması iki aşamadan oluşmaktadır. Birinci aşama başlangıç çözümünün oluşturulduğu Karınca Sistemi [12] algoritmasıdır. İkinci aşama ise, oluşturulan başlangıç çözümünün istenilen güvenilirlik düzeyine çıkartılması ve güvenilirliğin kontrol edilmesidir.

Birinci aşamada Dorigo'nun önermiş olduğu Karınca Sistemi algoritması üzerinde herhangi bir değişiklik yapılmadan kullanılmıştır. Birinci aşamada Karınca Sistemi'nin kullanılmasının sebebi, başlangıç çözümü olarak elde edilmek istenilen GSP çözümünü, büyük boyutlu ve matematiksel modelleme yöntemi ile çözümü bulunamayan veya çok uzun zaman alan problemler için, hızlı ve etkin bir şekilde bulabilmesidir. İkinci aşamada ise, ilk aşamada bulunan en iyi GSP çözümü seçilir.

Seçilen topolojiye, istenilen güvenilirlik düzeyine ulaşıncaya kadar hat ekleyerek sonuç ağ topolojisi elde edilmektedir..

Bu çalışmada oluşturulan Melez Karınca Kolonisi algoritmasında elde edilen ağın 2-bağlılığı için herhangi bir ek kontrol mekanizmasına ihtiyaç duyulmamıştır. Bunun Sebebi, oluşturulan başlangıç GSP çözümünün 2-bağlılık koşulunu sağlamasıdır. Algoritmanın içerisinde, elde edilen topolojiden hat çıkarılmadığı için de oluşturulan başlangıç çözümündeki 2-bağlılık kısıtı bozulmamaktadır.

Algoritmayı adım adım inceleyecek olursak, birinci adımda algoritmanın başlangıç değerleri belirlenmektedir. Başlangıç değerleri ile ilgili bilgi 5.3.1'de anlatılmaktadır.

İkinci adımda, karınca Sistemi kullanılarak başlangıç çözümü üretilmektedir. Algoritmanın nasıl çalıştığı tezin üçüncü bölümünde detaylı olarak anlatılmaktadır. Burada elde edilmek istenilen GSP en iyi çözümdür. Algoritmanın ikinci adımının sonlanması için iki koşuldan sadece birisini sağlaması yeterlidir. Bu koşullar; tekrar sayısına ulaşılması veya GSP en iyi çözümün bulunmasıdır. Büyük boyutlu problemlerde GSP en iyi çözümü bilinmediğinden dolayı algoritmanın istenilen tekrar sayısına ulaşması yeterli olacaktır.

İkinci adımda her bir iterasyonda karınca sayısı kadar GSP çözümü elde edilmektedir. Bulunan çözümlerin her birisinin amaç fonksiyonu değeri incelenmekte ve iterasyonun en iyi amaç fonksiyonu değerine sahip çözümü daha önce bulunan en iyi amaç fonksiyonu değerleriyle karşılaştırılır. Eğer daha önce buluna çözümden daha iyi amaç fonksiyonu değerine sahip bir GSP çözümü elde edildi ise bulunan bu çözüm bulunabilen en iyi GSP çözümü olarak saklanmaktadır.

Üçüncü adımda, ikinci adımda bulunan en iyi çözüm seçilmektedir. Melez Karınca Algoritmasının ilerleyen aşamasında seçilen bu çözüm başlangıç çözümü olarak kullanılacaktır. İlerleyen adımlar sadece tek bir başlangıç çözümü üzerinden sonuca ulaşmaya çalışmaktadır.

Dördüncü adımdaki amaç başlangıç çözümünde elde edilen ağın istenilen güvenilirlik düzeyine ulaşacak şekilde hatların eklenmesidir. Bunun için Genetik algoritmada kullanılan Turnuva Seçim Mekanizması ile başlangıç çözümün hat ekleme yöntemi kullanılmaktadır. Bu adım da sezgisel bir yöntem içerdiği için, tekrarlı olarak uygulanmaktadır. Adımın başlangıcında tekrar sayısı olan (T) değeri sıfıra eşitlenmektedir. Bu adımda durma koşulu istenilen tekrar sayısına ulaşılması veya ağ için ulaşılabilecek en iyi değerin bulunmasıdır.

Karınca kolonisinde kullanılan karınca sayısı dördüncü adımda da kullanılmaktadır. Amaç yarışmacı bir yaklaşım izleyerek problemin çözümüne daha hızlı ulaşabilmektir. Her iterasyonda karınca sayısı kadar aday çözüm oluşturulmaktadır.

Eğer hesaplanan yaklaşık sistem güvenilirliği üst sınırı, istenilen sistem güvenilirliğinden daha düşük ise ağa Genetik Algoritmada kullanılan bir seçim mekanizması olan Turnuva Seçimi ile yeni bir hat eklenmektedir. Yeni hattın eklenmesinde hatların sezgisel matrisindeki değerlerine bakılmaktadır. Daha önce seçilmemiş hatların arasından rasgele hatlar seçilmekte ve bu hatlardan en küçük değere sahip olan hat ağa eklenmektedir. Daha sonra ağın tekrar güvenilirliği hesaplanmaktadır. Bu işlem Jan'ın üst limiti ile hesaplanan güvenilirlik düzeyi istediğimiz güvenilirlik düzeyine eşit veya daha fazla oluncaya kadar tekrarlanmaktadır (R(G^{ℓ}_{aday}) < R_0).

Karıncalar tarafından temsil edilen tüm aday çözümlerin Jan [10]'ın üst limit yaklaşımı ile güvenilirliği hesaplandıktan sonra, amaç fonksiyonu değeri en küçük olan aday çözüm seçilmektedir. Seçilen bu çözümün amaç fonksiyonu değeri, daha önceki çevirimlerde bulunan ağ topolojisinin değerinden daha düşük ise, aday ağ tasarımının Monte Carlo benzetimi ile güvenilirlik tahmini yapılmaktadır. Aday ağın Monte Carlo benzetimi sonucunda elde edilen güvenilirlik düzeyi istenilen güvenilirlik düzeyinden düşük ise yine turnuva seçim mekanizması yardımı ile ağa yine sezgisel matris değerleri kullanılarak yeni bir hat eklenir ve

Monte Carlo benzetimi tekrarlanır. Bu işlem aday ağın güvenilirlik düzeyi, istenilen güvenilirlik düzeyine eşit veya daha yüksek oluncaya kadar tekrarlanır (R(G^{ℓ}_{aday}) < $R(G^{\ell}_{aday})_{MC}$).

Beşinci adımda ise algoritma bulabildiği en iyi çözümü üretmiştir ve algoritma sonlanmaktadır.

6. UYGULAMA

Bu bölümde, güvenilirlik kısıtı altında en küçük maliyetli bilgisayar ağlarının topolojik en iyilenmesi problemi için Melez Karınca Kolonisi Algoritması geliştirilmiştir. Bu algoritmanın etkinliği, literatürde bulunan test problemleri ve yeni oluşturulan büyük boyutlu test problemleri üzerinde çözüm zamanı ve kalitesi açısından incelenmiştir. Elde edilen sonuçlar diğer yöntemlerle bulunan sonuçlarla ve var olan en iyi değerler ile karşılaştırılmıştır.

6.1 Etkinlik Ölçütleri

Geliştirilen Melez Karınca Kolonisi Algoritmasının etkinliğinin belirlenmesi ve karşılaştırılması amacı ile aşağıdaki ölçütler dikkate alınmıştır;

Çözümlerin Optimumdan Sapma Oranı: Geliştirilen algoritma, optimum çözümleri bilinen test problemleri üzerinde, algoritmanın bulduğu çözümlerin, problemlerin en iyi (optimum) çözümünden sapma oranına göre karşılaştırılmıştır. Test problemleri ve bu problemlerin optimum çözümleri Altıparmak [9]'dan alınmıştır.

Herhangi bir problem için, algoritmanın bulduğu çözümün optimum çözümden sapma oranı eşitlik (6.1)'deki gibi hesaplanmaktadır.

$$OS = \frac{MKKAÇ - OPT}{OPT} * 100 \qquad (6.1)$$

Burada;

MKKAÇ : Melez Karınca Kolonisi Algoritması ile elde edilen çözüm
OPT : Problemin optimum çözümü
OS : Optimumdan sapma oranı (%)

olarak kullanılmıştır.

Aranan Nokta Sayısı: Geliştirilen algoritmanın karşılaştırılması amacı ile algoritmanın çözüm uzayında aradığı toplam çözüm sayısını gösteren değer etkinlik ölçütüdür.

6.2 Test Problemleri

Bu çalışmada kullanılan test problemleri çizelge 6.1'de sınıflandırılmaktadır. Bu problemlerindeki ağ tipi tam bağlı olarak seçilmiştir.

Çizelge 6.1 Test Problemleri

Düğüm Sayısı	Ağ Tipi	En İyi Sonuç	Literatürde Daha Önce Çözüm Aranmış Mı?
5	Tam Bağlı	Biliniyor	Evet
6	Tam Bağlı	Biliniyor	Evet
7	Tam Bağlı	Biliniyor	Evet
8	Tam Bağlı	Biliniyor	Evet
9	Tam Bağlı	Biliniyor	Evet
10	Tam Bağlı	Biliniyor	Evet
15	Tam Bağlı	Bilinmiyor	Evet
20	Tam Bağlı	Bilinmiyor	Evet
22	Tam Bağlı	Bilinmiyor	Evet
25	Tam Bağlı	Bilinmiyor	Evet
30	Tam Bağlı	Bilinmiyor	Hayır
35	Tam Bağlı	Bilinmiyor	Hayır
40	Tam Bağlı	Bilinmiyor	Hayır
45	Tam Bağlı	Bilinmiyor	Hayır
50	Tam Bağlı	Bilinmiyor	Hayır

Bu çalışmada 6 ile 10 düğüm arasında olan küçük boyutlu her bir problem için çizelge 6.2'deki hat güvenilirliği ve güvenilirlik düzeyi değerleri dikkate alınmıştır. Böylece 25 farklı problem için 3 farklı hat güvenilirliği ve güvenilirlik düzeyi durumu için toplam 75 (25*3) farklı problem elde edilmiş ve çözülmüştür. Ayrıca en iyi sonucu bilinen 1 adet 11 düğümlü problemin çözümüne yer verilmiştir. Böylece toplam 76 adet küçük boyutlu problem incelenmiştir.

Çizelge 6.2 Test Problemleri için Hat Güvenilirliği ve Güvenilirlik Düzeyi Kombinasyonları

Hat Güvenilirliği	Güvenilirlik Düzeyi
0.90	0.90
0.90	0.95
0.95	0.95

Literatürde 15, 20, 22 ve 25 düğümlü orta büyüklükteki problemler için homojen ağlar, yani aynı hat güvenilirliğine sahip ağlar dikkate alınmıştır.

30 ve üzerinde düğüm sayısına sahip yeni oluşturulan ağlarda kullanılan hat güvenilirliği, p, ve istenilen güvenilirlik düzeyi değeri olan R_0 değerleri, çizelge 6.3'de gösterildiği gibi 3 farklı kombinasyon olarak dikkate alınmıştır. Böylece 5 farklı problem için 3 farklı hat güvenilirliği ve güvenilirlik düzeyi kombinasyonu dikkate alındığında toplam 15 (5 * 3) test problemi elde edilmektedir.

Çizelge 6.3 Büyük Boyutlu Problemler için Hat Güvenilirliği ve Güvenilirlik Düzeyi Kombinasyonları

Hat Güvenilirliği	Güvenilirlik Düzeyi
0.90	0.90
0.95	0.90
0.95	0.95

Geliştirilen Melez Karınca Kolonisi Algoritması, VISUAL BASIC dilinde kodlanmış ve denemelerin hepsi Intel Centrino Mobile 1.7 işlemcili bilgisayarda yapılmıştır.

Geliştirilen algoritma için herhangi bir deney tasarımına ihtiyaç duyulmamıştır. Bunun sebebi, algoritmanın içinde kullanılan Karınca Sistemi Algoritmasında Marco Dorigo'nun Karınca Sistemi için kendi çalışmalarında önerdiği parametrelerin aynen kullanılmasıdır.

6.3 Geliştirilen Melez Karınca Kolonisi Algoritması'nın Çalışma Uzunluğunun Belirlenmesi

Algoritmanın çalışma uzunluğunun belirlenmesi iki başlık halinde incelenebilir. Birinci olarak geliştirilen algoritmanın Karınca Sistemi Algoritması ile Gezgin Satıcı Problemini çözen ve başlangıç çözümünü elde eden kısmı ve ikinci olarak da elde edilen bu çözüme Turnuva Seçim Mekanizması ile hat ekleyerek ağ tasarımını istenilen güvenilirlik seviyesine ulaştıran bölümü.

Literatürde Karınca Sistemi Algoritması ile yapılan çalışmalarda, çalışma uzunluğunun belirlenmesinde aşağıdaki kriterler göz önüne alınmaktadır:

1. Önceden belirlenen iterasyon sayısının tamamlanması
2. Bir optimum çözüm bulunması

Karınca Sistemi Algoritmasında Dorigo [13] tarafından kullanılan uzunluk koşulları kullanılmış, buna ek olarak sadece optimum değeri bilinmeyen problemlerde, elde edilen çözümde daha fazla bir iyileşmenin sağlanamaması durumunda algoritmanın sonlandırılması kısıtı eklenmiştir.

Bu tez çalışmasında geliştirilen algoritman, en iyi çözümü bilinen problemler için, en iyi çözümü bulduğu anda durdurulmaktadır. Algoritmanın en iyi çözümü bulamadığı durumlarda veya en iyi çözümün bilinmediği problemlerde, algoritmanın çalışma uzunluğu iterasyon sayısının kısıtlanması ile belirlenmiştir.

Karınca Kolonisinin çok aracılı yapısından dolayı, problem büyüklüğü arttıkça, aracı sayısı da problem büyüklüğüne bağlı olarak artmaktadır. Yani 10 düğümlü bir problemin Karınca Sistemi Algoritması ile GSP en iyi değerini bulmak için 10 tane aracı kullanılırken, 20 düğümlü bir GSP probleminin çözümü için 20 aracı kullanılmaktadır. Bu da her bir iterasyonda incelenen çözüm sayısını problemin zorluğuna göre otomatik olarak artması anlamına gelmektedir. Böylece iterasyon sayısını her problem için sabit tutarak algoritmanın çalışması sağlanmıştır. Yapılan ön çalışmalarda ve denemelerde, iterasyon sayısı 100 olarak alındığında, hem küçük boyutlu problemlerde belirlenen iterasyon sayısına ulaşmadan çok önce algoritmanın en iyi değeri bulduğu gözlenmiş hem de büyük problemlerde iyi sonuçlar elde edilmiştir.

Taranan çözüm uzaylarının hesaplanmasında ise, başlangıç çözümünün bulunması için taranan çözüm uzayı ve ağın istenilen güvenilirlik seviyesine çıkartılması için taranan çözüm uzayı değerlerinin toplanması ile toplam taranan çözüm uzayı değerlerine erişilmektedir.

6.4 Küçük Boyutlu Test Problemleri

Literatürden bulunan 76 adet küçük boyutlu tam bağlı test problem için elde edilen detaylı sonuçlar daha kolay anlaşılabilmesi açısından düğüm sayılarına göre farklı çizelgelere bölünmüştür. Altı düğümlü test problemleri Çizelge 6.4'de, yedi düğümlü test problemleri Çizelge 6.5'de, sekiz düğümlü test problemleri Çizelge 6.6'da, dokuz düğümlü test problemleri Çizelge 6.7'de, on ve on bir düğümlü test problemleri Çizelge 6.8'de verilmiştir. Bu problemler için Melez Karınca Kolonisi Algoritması ile bulunabilen en iyi sonuçlar, taranan çözüm uzayı, en iyi değerden sapma yüzdeleri ve değişim katsayısı değerleri detaylı bir şekilde listelenmiştir. Önerilen MKKA algoritması, küçük boyutlu 76 test probleminde 45 kere en iyi sonuç elde etmiştir. Önerilen algoritma ile en iyi sonucun elde edilme oranı %59,21'dir.

Çizelge 6.4 Altı Düğümlü Test Problemleri İçin MKKA İle Elde Edilen Sonuçlar

Problem Numarası	Düğüm Sayısı	Hat Sayısı	p	R_0	En İyi Sonuç	MKKA İle Bulunabilen En İyi Sonuç	Taranan Çözüm Uzayı	En İyi Sonucun (Best) En İyi (Optimum)'den Sapması (%)	Değişim Katsayısı
1	6	15	0.90	0.90	231	239	17	3,46320346	0
2	6	15	0.90	0.90	239	260	13	8,78661088	0
3	6	15	0.90	0.90	227	227	12	0	0
4	6	15	0.90	0.90	212	224	16	5,66037736	0
5	6	15	0.90	0.90	184	184	43	0	0
6	6	15	0.90	0.95	254	258	19	1,57480315	0
7	6	15	0.90	0.95	286	288	20	0,6993007	0
8	6	15	0.90	0.95	275	275	131	0	0,0076384
9	6	15	0.90	0.95	255	255	28	0	0
10	6	15	0.90	0.95	198	204	141	3,03030303	0,0102838
11	6	15	0.95	0.95	227	227	14	0	0
12	6	15	0.95	0.95	213	213	12	0	0
13	6	15	0.95	0.95	190	190	13	0	0
14	6	15	0.95	0.95	200	200	13	0	0
15	6	15	0.95	0.95	179	179	19	0	0

Çizelge 6.5 Yedi Düğümlü Test Problemleri İçin MKKA İle Elde Edilen Sonuçlar

Problem Numarası	Düğüm Sayısı	Hat Sayısı	p	R_0	En İyi Sonuç	MKKA İle Bulunabilen En İyi Sonuç	Taranan Çözüm Uzayı	En İyi Sonucun (Best) En İyi (Optimum)'den Sapması (%)	Değişim Katsayısı
16	7	21	0.90	0.90	189	189	26	0	0
17	7	21	0.90	0.90	184	184	1150	0	0,0478532
18	7	21	0.90	0.90	243	264	15	8,64197531	0
19	7	21	0.90	0.90	129	129	18	0	0
20	7	21	0.90	0.90	124	131	196	5,64516129	0,0287045
21	7	21	0.90	0.95	205	205	95	0	0
22	7	21	0.90	0.95	209	209	1007	0	0,0223632
23	7	21	0.90	0.95	268	269	22	0,37313433	0
24	7	21	0.90	0.95	143	143	17	0	0
25	7	21	0.90	0.95	153	153	119	0	0
26	7	21	0.95	0.95	185	185	182	0	0,0068226
27	7	21	0.95	0.95	182	182	1152	0	0,0445391
28	7	21	0.95	0.95	230	230	15	0	0
29	7	21	0.95	0.95	122	122	16	0	0
30	7	21	0.95	0.95	124	131	104	5,64516129	0

Çizelge 6.6 Sekiz Düğümlü Test Problemleri İçin MKKA İle Elde Edilen Sonuçlar

Problem Numarası	Düğüm Sayısı	Hat Sayısı	p	R_0	En İyi Sonuç	MKKA İle Bulunabilen En İyi Sonuç	Taranan Çözüm Uzayı	En İyi Sonucun (Best) En İyi (Optimum)'den Sapması (%)	Değişim Katsayısı
31	8	28	0.90	0.90	208	208	30	0	0
32	8	28	0.90	0.90	203	219	37	7,8817734	0
33	8	28	0.90	0.90	211	211	401	0	0,003989
34	8	28	0.90	0.90	291	299	39	2,74914089	0
35	8	28	0.90	0.90	178	180	194	1,12359551	0
36	8	28	0.90	0.95	247	247	164	0	0,0025585
37	8	28	0.90	0.95	247	248	92	0,4048583	0
38	8	28	0.90	0.95	245	245	65	0	0
39	8	28	0.90	0.95	336	339	31	0,89285714	0
40	8	28	0.90	0.95	202	203	742	0,4950495	0,0116122
41	8	28	0.95	0.95	179	179	28	0	0
42	8	28	0.95	0.95	194	197	39	1,54639175	0
43	8	28	0.95	0.95	197	197	208	0	0,0032072
44	8	28	0.95	0.95	276	290	28	5,07246377	0
45	8	36	0.95	0.95	173	180	186	4,04624277	0

Çizelge 6.7 Dokuz Düğümlü Test Problemleri İçin MKKA İle Elde Edilen Sonuçlar

Problem Numarası	Düğüm Sayısı	Hat Sayısı	p	R_0	En İyi Sonuç	MKKA İle Bulunabilen En İyi Sonuç	Taranan Çözüm Uzayı	En İyi Sonucun (Best) En İyi (Optimum)'den Sapması (%)	Değişim Katsayısı
46	9	36	0.90	0.90	239	239	43	0	0
47	9	36	0.90	0.90	191	194	34	1,57068063	0
48	9	36	0.90	0.90	257	257	79	0	0
49	9	36	0.90	0.90	171	171	109	0	0,0036942
50	9	36	0.90	0.90	198	200	1292	1,01010101	0,0095275
51	9	36	0.90	0.95	286	286	775	0	0,0030519
52	9	36	0.90	0.95	220	221	49	0,45454545	0
53	9	36	0.90	0.95	306	306	230	0	0,0015355
54	9	36	0.90	0.95	219	219	649	0	0,0041732
55	9	36	0.90	0.95	237	238	628	0,42194093	0
56	9	36	0.95	0.95	209	209	60	0	0
57	9	36	0.95	0.95	171	174	43	1,75438596	0
58	9	36	0.95	0.95	233	244	101	4,72103004	0
59	9	36	0.95	0.95	151	151	20	0	0
60	9	36	0.95	0.95	185	187	1163	1,08108108	0,0082055

Çizelge 6.8 On ve On Bir Düğümlü Test Problemleri İçin MKKA İle Elde Edilen Sonuçlar

Problem Numarası	Düğüm Sayısı	Hat Sayısı	p	R_0	En İyi Sonuç	MKKA İle Bulunabilen En İyi Sonuç	Taranan Çözüm Uzayı	En İyi Sonucun (Best) En İyi (Optimum)'den Sapması (%)	Değişim Katsayısı
61	10	45	0.90	0.90	131	131	289	0	0
62	10	45	0.90	0.90	154	154	242	0	0
63	10	45	0.90	0.90	267	267	176	0	0
64	10	45	0.90	0.90	263	266	585	1,14068441	0,0019777
65	10	45	0.90	0.90	293	294	44	0,34129693	0
66	10	45	0.90	0.95	153	153	120	0	0
67	10	45	0.90	0.95	197	197	202	0	0,0016044
68	10	45	0.90	0.95	311	311	265	0	0
69	10	45	0.90	0.95	291	293	986	0,68728522	0,0021497
70	10	45	0.90	0.95	358	358	172	0	0
71	10	45	0.95	0.95	121	121	1093	0	0,0025942
72	10	45	0.95	0.95	136	136	153	0	0
73	10	45	0.95	0.95	236	236	84	0	0
74	10	45	0.95	0.95	245	249	134	1,63265306	0
75	10	45	0.95	0.95	268	269	23	0,37313433	0
76	11	55	90	90	246	246	57	0	0

Melez Karınca Kolonisi Algoritmasının, küçük boyutlu test problemleri için NGA, LS-NGA, TA_udh, TB, KSE_TB_2 ve TB_KKE_TB algoritmaları ile arama uzayına göre karşılaştırılması ve elde edilen değerlerin çözüm uzayının tamamına oranı çizelge 6.9'da verilmektedir. Bu tablodan anlaşılacağı üzere, önerilen algoritma, diğer algoritmalara kıyasla oldukça küçük bir arama uzayı ile sonuca ulaşmıştır.

Çizelge 6.9 Küçük Boyutlu Test Problemleri İçin MKKA'nın Arama Uzayına Göre Karşılaştırılması

Algoritma/(N,L)	(6,15)	(7,21)	(8,28)	(9,36)	(10,45)	(11,55)	Ortalama
Çözüm Uzayı	*3,28E+04*	*2,10E+06*	*2,68E+08*	*6,87E+10*	*3,52E+13*	*3,60E+16*	*6,01E+15*
NGA	2378	6254	11638	28166	62783	83833	32509
NGA/ÇÖ	*7,26E-02*	*2,98E-03*	*4,34E-05*	*4,10E-07*	*1,78E-09*	*2,33E-12*	*5,41E-12*
LS-NGA	1596	4190	7811	12922	34168	43566	17376
LS-NGA/ÇÖ	*4,87E-02*	*2,00E-03*	*2,91E-05*	*1,88E-07*	*9,71E-10*	*1,21E-12*	*2,89E-12*
TA_udh	168	509	837	989	2372	N/A	975
TA_udh/ÇÖ	*5,11E-03*	*2,43E-04*	*3,12E-06*	*1,44E-08*	*6,74E-11*	N/A	*1,62E-13*
TB	3995	3991	4473	4687	5905	N/A	4610
TB/ÇÖ	*1,22E-01*	*1,90E-03*	*1,67E-05*	*6,82E-08*	*1,68E-10*	N/A	*7,67E-13*
KSE_TB_2	22501	14625	14340	17621	16404	23070	18093
KSE_TB_2/ÇÖ	*6,87E-01*	*6,97E-03*	*5,34E-05*	*2,56E-07*	*4,66E-10*	*6,40E-13*	*3,01E-12*
TB_KKE_TB	8425	6350	7436	10482	11104	17527	10221
TB_KKE_TB/ÇÖ	*2,57E-01*	*3,03E-03*	*2,77E-05*	*1,53E-07*	*3,16E-10*	*4,86E-13*	*1,70E-12*
MKKA	34	276	152	352	305	57	196
MKKA/ÇÖ	*1,04E-03*	*1,32E-04*	*5,66E-07*	*5,12E-09*	*8,67E-12*	*1,58E-15*	*3,26E-14*

Melez Karınca Kolonisi Algoritmasının, küçük boyutlu test problemleri için NGA, LS-NGA, TA_udh, TB, DKİ, TL_DKİ, KSE_TB_2 ve TB_KKE_TB algoritmaları ile en iyi değerden sapma oranına göre karşılaştırılması çizelge 6.10'da verilmektedir. Bu tabloda görüldüğü üzere, önerilen algoritmanın ortalama en iyi değerden sapması %1'in altındadır. Diğer algoritmalara bakıldığında, bu değer oldukça iyidir.

Çizelge 6.10 Küçük Boyutlu Problemler İçin MKKA'nın En İyi Değerden (%) Sapma Oranına Göre Karşılaştırılması

Algoritma/(N,L)	(6,15)	(7,21)	(8,28)	(9,36)	(10,45)	(11,55)	Ortalama (%)
NGA	0,472	1,068	1,176	2,957	3,509	4,675	2,310
LS/NGA	0,400	0,777	0,889	1,050	1,094	1,094	0,884
TA_udh	0,000	0,000	0,000	0,000	0,038	N/A	0,008
TB	2,818	0,716	0,553	0,345	0,962	N/A	1,079
DKİ	4,596	2,244	1,830	3,820	2,682	0,610	2,630
TL_DKİ	5,616	1,997	0,126	4,418	3,684	0,000	2,640
KSE_TB_2	0,126	0,085	0,065	0,191	0,166	0,000	0,106
TB_KKE_TB	0,193	0,012	0,284	0,770	0,554	0,000	0,302
MKKA	1,548	1,354	1,614	0,734	0,278	0,000	0,921

Melez Karınca Kolonisi Algoritmasının, küçük boyutlu test problemleri için NGA, LS-NGA, TA_udh, TB, DKİ, TL_DKİ, KSE_TB_2 ve TB_KKE_TB algoritmaları ile değişim katsayısı değerlerine göre karşılaştırılması çizelge 6.11'de verilmektedir. Önerilen algoritmanın sonuçlarının çizelgedeki en iyi üçüncü değer olduğu görülmektedir ve en iyi ikinci değer ile arasında sadece 0,001 birimlik bir fark vardır. MKKA ile araştırılan test problemlerinde, değişim katsayısı değerleri türünden oldukça kaliteli sonuçlar elde edilmiştir.

Çizelge 6.11 Küçük Boyutlu Test Problemleri İçin MKKA'nın Değişim Katsayısı Değerlerine Göre Karşılaştırılması

Algoritma/(N,L)	(6,15)	(7,21)	(8,28)	(9,36)	(10,45)	(11,55)	Ortalama
NGA	0,025	0,016	0,014	0,039	0,035	0,050	0,030
LS/NGA	0,008	0,012	0,011	0,017	0,014	0,000	0,010
TA_udh	0,000	0,000	0,000	0,000	0,001	N/A	0,0002
TB	0,001	0,000	0,007	0,003	0,007	N/A	0,004
DKİ	0,040	0,017	0,022	0,044	0,031	0,010	0,027
TL_DKİ	0,041	0,018	0,018	0,042	0,028	0,000	0,024
KSE_TB_2	0,002	0,002	0,000	0,002	0,002	0,000	0,002
TB_KKE_TB	0,002	0,000	0,004	0,007	0,005	0,000	0,003
MKKA	0,001	0,010	0,001	0,002	0,001	0,000	0,003

Melez Karınca Kolonisi Algoritmasının, küçük boyutlu test problemleri için NGA, LS-NGA, TA_udh, TB, DKİ, TL_DKİ, KSE_TB_2 ve TB_KKE_TB algoritmaları ile elde edilen ortalama değerlere göre karşılaştırılması çizelge 6.12'de verilmektedir. Bu karşılaştırma çizelgesinde, her bir kriter için bulunan en iyi değer işaretlenmiştir. Buna göre, MKKA diğer algoritmaların tümünden çok daha az arama uzayı tarayarak çözüme ulaşmıştır. Ayrıca MKKA ile en iyi değerden sapma ve değişim katsayısı değerlerine göre de genel ortalamanın altında sonuçlar elde edilmiştir.

Çizelge 6.12 Küçük Boyutlu Problemler İçin MKKA ile Diğer Algoritmaların Ortalamalarının Karşılaştırılması

Algoritma/Kriter	Arama Uzayı	En İyi Değerden Sapma (%)	Değişim Katsayısı
NGA	32509	2,310	0,030
LS/NGA	17376	0,884	0,010
TA_udh	975	0,008	0,0002
TB	4610	1,079	0,004
DKİ	N/A	2,630	0,027
TL_DKİ	N/A	2,640	0,024
KSE_TB_2	18093	0,106	0,002
TB_KKE_TB	10221	0,302	0,003
MKKA	196	0,921	0,003
Genel Ortalama:	*11997*	*1,209*	*0,011*

6.5 Orta Büyüklükteki Problemler

Geliştirilen algoritma ile literatürde daha önce araştırılmış ve en iyi sonucu bilinmeyen 15, 20, 22 ve 25 düğümlü orta büyüklükteki problemler çözülmüştür. Elde edilen sonuçlar GA, TA, TB ve Opti-Net algoritmalarının sonuçları ile çizelge 6.13'de karşılaştırılmaktadır.

Çizelge 6.13 Orta Büyüklükteki Problemler İçin MKKA İle Elde Edilen Sonuçların Karşılaştırılması Ve Sağlanan (%) İyileşme

Problem Numarası	Düğüm Sayısı	Hat Sayısı	p	R_0	MKKA	GA	TA_udh	TB	Opti-Net	Sağlanan % İyileşme
1	15	105	0.90	0.95	253	397	262	266	304	3,557312253
2	20	190	0.95	0.95	129	419	154	159	286	19,37984496
3	22	231	0.90	0.90	276	415	298	344	-	7,971014493
4	25	300	0.95	0.90	238	1606	-	-	402	68,90756303

En iyi sonucu bilinmeyen orta büyüklükteki problemlerde MKKA, diğer algoritmalar ile bulunan çözümlerden daha iyi sonuçlar elde etmiştir. Geliştirilen algoritma,

çizelge 6.13'deki bir numaralı problemle MKKA'dan sonra en iyi sonucu bulan TA' ya göre %3,55 daha iyi bir sonuç bulmuş, iki numaralı problemde %19,37 oranında bir iyileşme sağlamış, üç numaralı problemde ise %7,97 oranında daha iyi bir çözüme ulaştığı gözlemlenmiştir. Dört numaralı problemde Opti-Net algoritmasına göre %68,91 oranında bir iyileşme sağlanmıştır.

Orta büyüklükte ve en iyi sonucu bilinmeyen 15, 20, 22 ve 25 düğümlü problemlerin 10 kere çalıştırılmasının sonucu MKKA'nın taradığı ortalama arama uzayı değerleri, GA ile taranan arama uzayı değerleri ve bu değerlerin çözüm uzayına göre oranları Çizelge 6.14'de gösterilmektedir. Bu tezde önerilen algoritma, orta büyüklükteki problemler için oldukça az bir uzayı tarayarak sonuca ulaşmıştır. Genetik algoritmanın taradığı uzayın %1'inden daha az bir uzayı araştırdığı görülmektedir.

Çizelge 6.14 Orta Büyüklükteki problemler için Melez Karınca Kolonisi Algoritmasının Arama Uzayına Göre Karşılaştırılması

Algoritma/(N,L)	(15,105)	(20,190)	(22,231)	(25,300)	Ortalama
Çözüm Uzayı	*4,06E+31*	*1,57E+57*	*3,45E+69*	*2,04E+90*	*5,09E+89*
GA	1,40E+05	2,00E+05	N/A	4,00E+05	2,47E+05
GA/ÇÖ	*3,45E-27*	*1,27E-52*	*N/A*	*1,96E-85*	*1,15E-27*
MKKA	1,33E+03	4,52E+02	2,80E+03	3,60E+03	2,04E+03
MKKA/ÇÖ	*3,27E-29*	*2,88E-55*	*8,10E-67*	*1,76E-87*	*4,01E-87*
MKKA/GA	0,009	0,002	N/A	0,009	0,008
MKKA/GA(%)	0,948214	0,226	N/A	0,89875	0,828111

Değişim katsayıları açısından KSE_TB_2, TB_KKE_TB ve MKKA arasında yapılan karşılaştırma çizelge 6.15'de verilmektedir. Bu sonuçlara göre önerilen algoritmanın, orta büyüklükteki problemlerde çok düşük değişim katsayılarına sahip olduğu ve oldukça kaliteli sonuçlar elde ettiği görülmektedir.

Çizelge 6.15 Orta Büyüklükteki problemler için Melez Karınca Kolonisi Algoritmasının Değişim Katsayısına Göre Karşılaştırılması

Algoritma/(N,L)	(15,105)	(20,190)	(25,300)	Ortalama
KSE_TB_2	0,079	0,071	0,062	0,071
TB_KKE_TB	0,092	0,089	0,056	0,079
MKKA	0,000	0,041	0,004	0,015

6.6 Büyük Boyutlu Problemler

Bu çalışmada daha önce literatürde çalışılmış test problemlerinin yanı sıra, yeni test problemleri oluşturulmuş ve oluşturulan test problemlerinin çözümleri aranmıştır. Yeni test problemlerine ihtiyaç duyulmasının sebebi, bilinen problemlerin MKKA ile çok hızlı ve etkin bir şekilde çözülebilmesidir. Geliştirilen algoritmanın daha büyük boyutlu problemleri de çözebileceği düşünülmüş ve 50 düğümlü problemlere kadar yeni test problemleri oluşturulmuştur.

Yeni problem setinin oluşturulmasında, daha önceki problem setlerinin oluşturulmasında kullanılan bir ile yüz sayıları arasında rasgele uzaklık değeri atanması yöntemi kullanılmıştır.

En iyi sonucu bilinmeyen ve yeni oluşturulan 30, 35, 40, 45 ve 50 düğümlü test problemleri için, 10 tekrar sonucunda bulunan sonuçlar çizelge 6.16'de verilmektedir. Her deneme için elde edilen detaylı sonuçlar ise Ek-1'de verilmektedir.

Çizelge 6.16 Büyük Boyutlu Problemler İçin MKKA İle Elde Edilen Sonuçlar

Problem Numarası	Düğüm Sayısı	Hat Sayısı	p	R_0	MKKA İle Bulunabilen En İyi Sonuç	Taranan Çözüm Uzayı	Çözüm Uzayı	Değişim Katsayısı
1	30	435	0.90	0.90	346	3060	8,87E+130	0,00000
2	30	435	0.95	0.90	249	3327	8,87E+130	0,00381
3	30	435	0.95	0.95	249	3942	8,87E+130	0,00666
4	35	595	0.90	0.90	369	3294	1,30E+179	0,00000
5	35	595	0.95	0.90	237	5754	1,30E+179	0,00931
6	35	595	0.95	0.95	237	5383	1,30E+179	0,00934
7	40	780	0.90	0.90	438	4864	6,36E+234	0,00509
8	40	780	0.95	0.90	267	6020	6,36E+234	0,00393
9	40	780	0.95	0.95	267	7548	6,36E+234	0,00356
10	45	990	0.90	0.90	390	8436	1,05E+298	0,01080
11	45	990	0.95	0.90	246	8555	1,05E+298	0,01085
12	45	990	0.95	0.95	246	8118	1,05E+298	0,01071
13	50	1225	0.90	0.90	440	9955	N/A	0,01532
14	50	1225	0.95	0.90	260	9898	N/A	0,02033
15	50	1225	0.95	0.95	260	9918	N/A	0,02054

7. SONUÇ

Bu tezde, bilgisayar ağlarının topolojik en iyilenmesi probleminin çözümü için sürü yaklaşımına dayalı Karınca Kolonisi Algoritması ile melez bir algoritma önerilmiştir. Önerilen algoritma, güvenilirlik kısıtı altında en küçük maliyetli ağ tasarımını bulmayı amaçlamaktadır. Bu problem türü, literatürdeki NP-zor problemlerden bir tanesidir. Önerilen algoritma 76 küçük boyutlu, 4 Orta boyutlu ve 15 büyük boyutlu olmak üzere toplam 95 problem üzerinde test edilmiştir.

Çalışmanın uygulama kısmında bahsedildiği üzere, önerilen Melez Karınca Kolonisi Algoritması ile Literatürdeki diğer algoritmalardan Genetik Algoritma, Tabu Arama, Tavlama Benzetimi, Değişken Komşu Arama, Kuş Sürüsü eniyilemesi ve Karınca Kolonisi Algoritması ile yazılmış diğer algoritmaların sonuçları ile karşılaştırmalar yapılmıştır. Bu karşılaştırmalarda algoritmalar en iyi değerden sapma oranları, arama uzayları ve değişim katsayıları değerleri dikkate alınarak incelenmiştir.

Küçük boyutlu ve en iyi değeri bilinen problemler için Melez Kasrınca Kolonisi Algoritması ile diğer algoritmaların ortalama sonuçları incelendiğinde, çözüm uzayı açısından oldukça büyük bir iyileşme kaydedildiği gözlenmektedir. En iyi değerden sapma ve değişim katsayısı açısından, literatürdeki algoritmaların ortalama değerlerinden daha iyi sonuçlar elde edilmiştir.

Orta büyüklükteki en iyi sonucu bilinmeyen problemler için ise diğer algoritmaların bulduğu sonuçlara göre %3,5 ile %68,9 arasında bir iyileşme sağlanmıştır. Genetik Algoritma ile karşılaştırılan çözüm uzayı değerlerinde küçük boyutlu problemlerde olduğu gibi oldukça büyük bir azalma kaydedilmiştir.

Tez kapsamında, literatürde daha önce bulunmayan 30, 35, 40, 45 ve 50 düğümlü yeni test problemleri önerilmiş ve bu problemlere çözüm aranmıştır. Test sonuçları incelendiğinde, Melez Karınca Kolonisi Algoritmasının bu problemler için oldukça kaliteli sonuçlar elde ettiği söylenebilir. Büyük boyutlu problemlerde, çözüm

uzayının büyüklüğü de dikkate alınırsa, önerilen algoritmanın taradığı çözüm uzayı değerleri oldukça azdır. Ayrıca değişim katsayısı değerlerinin düşük olması, algoritma ile elde edilen sonuçların kalitesinin bir göstergesidir.

İleride bu konu ile ilgili olarak yapılabilecek çalışmalarda, Melez Karınca Kolonisi Algoritması ile elde edilen sonuçların yakın komşulukları incelenebilir ve elde edilen sonuçlar daha da iyileştirilebilir. Ayrıca 50 düğümden daha büyük boyutlu problemlerin çözümü için başlangıç çözümünün oluşturulmasında, literatürdeki diğer Karınca Kolonisi algoritmaları denenebilir.

KAYNAKLAR LİSTESİ

[1] AGGARWAL K.K., CHOPRA Y.C., BAJAWA J.S., Topological Layout of Links for Optimizing The S-T Reliability in Computer Communication System, Microelectronics & Reliability, vol.22, no.3, s341-345, 1982

[2] JAN, R., H., HWANG, F., J., CHENG, S., T., Topological Optimization of a Communication Network Subject to Reliability Constraint, IEEE Transaction on Reliability, vol.42-1, s.63-70, 1993

[3] DENGIZ, B., ALTIPARMAK, F., SMITH, A., E., Efficient Optimization of All-Terminal Reliable Networks, IEEE Transactions on Reliability, vol.46, no.1, s.18-26, 1997a.

[4] DENGIZ, B., ALTIPARMAK, F., SMITH, A., E., Local Search Genetic Algorithm for Optimal Design of Reliable Networks, IEEE Transactions on Evolutionary Computation, vol.1, no.3, s.179-188, 1997b.

[5] DENGIZ, B., ALABAS, C., A Tabu Search Algorithm for Design of Computer Networks, 4th World Multi-Conference on Circuits, Systems, Communications and Computers, s.363-368, 2000

[6] DENGIZ, B., ALABAS, C., A Simulated Annealing Algorithm for Design of Computer Communication Networks, 5th Multi-Conference on Systemics, Cybernetics, and Informatics, vol.5, Part I. s.188 -193, 2001

[7] HOSAM, M., F., ABOELFOTOH AND LOULWA S. AL-SUMAIT, A Neural Approach to Topological Optimization of Communication Networks, With Reliability Constraints, IEEE Transactions On Reliability, vol.50, no.4, 2001

[8] KONAK, A., SMITH, A., E., Handbook of Optimization in Telecommunications, Springer Science + Business Media, s.735-761, 2006

[9] ALTIPARMAK, F., Genetik Algoritma ile Haberleşme Şebekelerinin Topolojik Optimizasyonu, Doktora Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 1996

[10] JAN, R, H., Design of Reliable Networks, Computers and Operations Research, vol.20-1, s.25-34, 1993

[11] YEH, M., LIN, J., YEH, W., A New Monte Carlo Method for Estimating Network Reliability, IEEE Transactions on Reliability, vol.29-1, s.27-32, 1994

[12] COLORNI, A., DORIGO, M. and MANIEZZO, V., Distributed Optimization by Ant Colonies, Proceedings of the First European Conference on Artificial Life, s.134-142, 1992

[13] COLORNI, A., DORIGO, M. and MANIEZZO, V., An Investigation of Some Properties of an Ant Algorithm, Proceedings of PPSN-II, Second International

Conference on Parallel Problem Solving from Nature, Amsterdam, s.509-520, 1992

[14] DORIGO, M., MANIEZZO, V., COLORNI, A., Positive Feedback as a Search Strategy, Technical Report 91-016, Dipartimento di Elettronica, Politecnico di Milano, Milan, 1991

[15] DORIGO, M., MANIEZZO, V., COLORNI, A., The Ant System: An Autocatalytic Optimizing Process, Technical Report 91-016 Revised, Dipartimento di Elettronica, Politecnico di Milano, Milan, 1991

[16] DORIGO, M., MANIEZZO, V., COLORNI, A., Ant System: Optimization by a Colony of Cooperating Agents, IEEE Transactions on Systems, Man and Cybernetics, vol.26-1, s.29-41, 1996

[17] KRISHNAIYER, K., CHERAGHI, S.H., Ant Algorithms: Review and Future Applications in IERC´02, Industrial Engineering Research Conference, Orlando, Florida, USA, 2002

[18] DORIGO, M., STUTZLE, T., The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances, Technical Report, IRIDIA/2000-32, IRIDIA, University Libre de Bruxelles, Belgium, 2000

[19] GOSS, S., ARON, S., DENEUBOURG, J., L., PASTEELS, J., M., Self Organized Shortcuts in the Argentine Ant, Naturwissenschaften, vol.76, s.579-581, 1989

[20] DORIGO, M., Optimization, Learning and Natural Algorithms, PhD. Thesis, Department of Electronics, Milano, 1992

[21] DORIGO, M. and STÜTZLE, T., Ant Colony Optimization, MIT Press, 2004

[22] GOLDBERG, D., E., Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley, 1989

[23] BULLNHEIMER, B., HARTL, R., F. and STRAUSS, C., A New Rank Based Version of the Ant System: A Computational Study, Central European Journal for Operations Research and Economics, vol.7-1, s.25-38, 1999

[24] STÜTZLE, T., and HOOS, H., H., The Max-Min Ant System and Local Search for the Traveling Salesman Problem, IEEE International Conference on Evolutionary Computation, pp.309-314, 1997

[25] DORIGO, M. and GAMBARDELLA, L., M., Ant Colony System: A Cooperative Learning approach to the Traveling Salesman Problem, IEEE Transactions on Evolutionary Computation, vol.1, s.53-66, 1997

[26] BOORSTYN, Robert R. and FRANK Howard, Large Scale Network Topological Optimization, IEEE, Transactions on Communications, vol.25, no.1, s29-37, 1977

[27] AGGARWAL, K. K. and SURESH R., Reliability Evaluation in Computer Communication Networks, IEEE Transactions on Reliability. Vol. R-30, no. 1, s. 32-35, 1981

[28] CHOPRA, Y., C., SOHI, B., S., TIWARI, R., K., Network Topology For Maximizing The Terminal Reliability In A Computer Communication Network, Microelectronics and Reliability, vol.24, s.911-913, 1984

[29] VENETSANOPOULOS, A. N. and SINGH, I., Topological Optimization Of Communication Networks Subject To Reliability Constraints, Prob. Control Info. Theory. Vol.15, no.1, s.63-78, 1986

[30] KUMAR, Anup, PATHAK, Rakesh, M., GUPTA, Yash, P., Genetic Algorithm - Based Reliability Optimization for Computer Network Expansion, IEEE Transactions On Reliability, vol. 44, no.1, s.63-72, 1995

[31] KUMAR, Anup, PATHAK, Rakesh, M., GUPTA, Yash, P., PARSAEI, Hamid, R., A Genetic Algorithm for Distributed System Topology Design, Computers and Industrial Engineering, vol.28, no.3, s.596-670, 1995

[32] COSTAMAGNA, Eugenio, FANNI, Alessandra, GIACINTO, Giorgio, Simulated Annealing Algorithm for The Optimization Of Communication Networks, International Symposium on Signals Systems and Electronics, San Francisco, s.405-408, 1995

[33] DEETER, Darren, L. and SIMITH, Alice, E., Heuristic Optimization of Network Design Considering All - Terminal Reliability, Annual Reliability and Maintainability Symposium, Philadelphia, s.194-199, 1997

[34] AHUJA, Sanjay, P., Performance Based Reliability Optimization for Computer Networks, Southeastcon '97, s.121-125, 1997

[35] DENGİZ, Berna, ALTIPARMAK, Fulya, SIMITH, Alice, E., Efficient Optimization of All - Terminal Reliable Networks, Using an Evolutionary Approach, IEEE Transactions of Evolutionary Reliability, vol.46 no.1, s.18-26, 1997

[36] PIERRE, Samuel and LEGAULT, Gisele, A Genetic Algorithm for Designing Distributed Computer Network Topologies, IEEE Transactions On Systems, Man, And Cybernetics - Part B: Cybernetics, vol.28, no.2, s.249-258, 1998

[37] PIERRE, Samuel and ELGIBAOUI, Ali, A Tabu-Search Approach for Designing Computer-Network Topologies with Unreliable Components, IEEE Transactions On Reliability, vol. 46, no. 3, 1997

[38] COSTAMAGNA, Eugenio, FANNI, Alessandra, GIACINTO, Giorgio, A Tabu Search Algorithm for the Optimization of Telecommunication Networks, European Journal of Operational Research, vol.106, s.357-372 , 1998

[39] CHENG, Sheng, Tzong, Topological Optimization of a Reliable Communication Network, IEEE Transactions On Reliability, vol.47, no.3, 1998

[40] DEETER, Darren, L., and SMITH, Alice, E., Economic Design of Reliable Networks, IIE Transactions, vol.30, s.1161-1174, 1998

[41] ALTIPARMAK, Fulya, DENGIZ, Berna, SMITH, Alice, E., Reliability Optimization of Computer Communication Networks Using Genetic Algorithms, International Conference on Systems, Man and Cybernetics, vol.5, s.4676-4681, 1998

[42] KONAK, Abdullah and SMITH, Alice, E., A Hybrid Genetic Algorithm Approach for Backbone Design of Communication Networks, Proceedings of the 1999 Congress on Evolutionary Computation, Washington, D.C. (USA), s. 1817–1823, 1999

[43] LIU, Baoding, IWAMURA, K., Topological Optimization Models for Communication Network with Multiple Reliability Goals, Computers and Mathematics with Applications, vol.39, s.59-69, 2000

[44] ABOELFOTOH, Hosam, M., F. and AL-SUMAIT, Loulwa, S., A Neural Approach to Topological Optimization of Communication Networks With Reliability Constraints, IEEE Transactions on Reliability, vol. 50, no. 4, 2001

[45] FARD, N., LEE, Tea, Han, Spanning Tree Approach in All-Terminal Network Reliability Expansion, Computer Communications, vol.24, s.1348-1353, 2001

[46] KOIDE, T., SHINMORI, S., ISHII, H., Topological Optimization With a Network Reliability Constraint, Discrete Applied Mathematics, vol.115, s.125-149, 2001

[47] KUMAR, Rajeev, PARIDA, Pranja, P., GUPTA, Mohit, Topological Design of Communication Networks using Multi Objective Genetic Optimization, Proceedings of the 2002 Congress on Evolutionary Computation, vol.1, s.425-430, 2002

[48] SRIVAREE-RATANA, Chat, KONAK, Abdullah, SMITH, Alice E., Estimation Of All-Terminal Network Reliability Using An Artificial Neural Network, Computers & Operations Research, vol.29, s.849-868, 2002

[49] ALTIPARMAK, Fulya, and DENGIZ, Berna, Optimal Design of Reliable Computer Networks: A Comparison of Metaheuristics, Journal of Heuristics, vol.9, s.471-487, 2003

[50] MANDALL, Swamp, SAHA, Debashis, MUKHEJEE, Rajarshi, ROY, Anandarup, An Efficient Algorithm for Designing Optimal Backbone Topology for a Communication Networks, International Conference on Communication Technology, Beijing, China, vol.1, s.103-106, 2003

[51] ALTIPARMAK, Fulya, DENGIZ, Berna, SIMITH, Alice E., Reliability Estimation of Computer Communication Networks: ANN Models, Eighth IEEE International Symposium on Computers and Communication (ISCC'03), Kemer, Antalya, vol.2, s.1-6, 2003

[52] ALTIPARMAK F., GEN, M., DENGIZ, B., SMITH, A., E., A Genetic Algorithm with Fuzzy Logic Controller for Design of Communication Networks, Transactions on Electronics, Information and Systems, vol.24, no.10, s.1979-1985, 2004

[53] SHAO, Fang-Ming, SHEN, Xuemin, and HO, Pin-Han, Reliability Optimization of Distributed Access Networks With Constrained Total Cost, IEEE Transactions On Reliability, vol.54, no.3, s.421-430, 2005

[54] XIONG Jintao, GONG Weibo, A Novel Algorithm on Network Reliability Estimation, Mathematical And Computer Modeling, vol.41, no.2-3, s.119-133, 2005

[55] REICHELT, Dirk, GMILKOWSKY, Peter, LINSER, Sebastian, A Study of an Iterated Local Search on the Reliable Communication Networks Design Problem, EvoWorkshop 2005, Lausanne, s.156-165, 2005

[56] MARSEGUERRA Marzio, ZIO Enrico, PODOFILLINI Luca, and COIT David W., Optimal Design of Reliable Network Systems in Presence of Uncertainty, IEEE Transactions On Reliability, vol.54, no.2, s.243-253, 2005

[57] GEN, Mitsuo, KUMAR, Anup, KIM, Jong, Ryul, Recent Network Design Techniques Using Evolutionary Algorithms, International Journal of Production Economics, s.251–261, 2005

[58] REICHELT, Dirk, ROTHLAUF, Franz, Reliable Communication Network Design with Evolutionary Algorithms, International Journal of Computational Intelligence and Applications, vol.5, No.2, s.251–266, 2005

[59] KONAK, Abdullah, BARTOLACCI, Michael, R., Designing Survivable Resilient Networks: A Stochastic Hybrid Genetic Algorithm Approach, The International Journal of Management Science, vol.35, s.645-658, 2007

[60] CANCELA, Hector, PETINGI, Louis, Properties of a Generalized Source to All Terminal Network Reliability Model with Diameter Constraints, The International Journal of Management Science, vol.35, s.659-670, 2007

[61] KHAN, Salman, A., ENGELBRECHT, Andries, P., A New Fuzzy Operator and Its Application to Topology Design of Distributed Local Area Networks, Information Sciences, vol.177, s.2692-2711, 2007

[62] LUCIO, Gilberto, Flores, REED, Martin, J., HENNING, Ian, D., Guided Local Search as a Network Planning Algorithm That Incorporates Uncertain Traffic Demands, Computer Networks, vol.51, s.3172-3196, 2007

[63] HUI, Kin-Ping, Monte Carlo Network Reliability Ranking Estimation, IEEE Transactions on Reliability, vol.56, no.1, 2007

[64] MARQUEZA, Jose, Emmanuel, Ramirez, ROCCO, Claudio, M., All-Terminal Network Reliability Optimization via Probabilistic Solution Discovery, Reliability Engineering and System Safety 93, s.1689–1697, 2008

[65] SEM, K., MALHOTRA, S., Multi Criteria Network Design Using Genetic Algorithm, Wireless, Mobile and Multimedia Networks, IET International Conference, s.56-60, 2008

[66] PREMPRAYOON, Patompom, and WARDKEIN, Paramote, Topological Communication Network Design Using Ant Colony Optimization, Advanced Communication Technology, vol.2, s.1147-1151, 2005

[67] QOUBUTRA, M., PREMPRAYOON, P., and WARDKEIN, P., Topological Communication Network Design using Improved Ant Colony Optimization, Networks and Communication Systems, Krabi, Thailand, s.118-128, 2005

[68] DENGIZ, B., ALTIPARMAK, F., BELGIN, O., A Hybrid Ant Colony Optimization Approach for the Design of Reliable Networks, Evolutionary Computation, s. 1118-1125, 2007

[69] HOPCRAFT, J., E., ULLMAN, J., D., Set Merging Algorithms, SIAM Journal of Computers, vol.2, s.296-303, 1973

EK 1 BÜYÜK BOYUTLU PROBLEMLER İÇİN MKKA İLE ELDE EDİLEN SONUÇLAR

Problem Numarası	MKKA ile Bulunabilen En İyi Sonuç	Değişim Katsayısı	Deneme Numarası									
			1	2	3	4	5	6	7	8	9	10
1	346	0	**346**	**346**	**346**	**346**	**346**	**346**	**346**	**346**	**346**	**346**
2	249	0,00381	**249**	**249**	**249**	**249**	**249**	**249**	**249**	**249**	252	**249**
3	249	0,00666	253	**249**	253	**249**	**249**	**249**	**249**	**249**	**249**	250
4	369	0	**369**	**369**	**369**	**369**	**369**	**369**	**369**	**369**	**369**	**369**
5	237	0,00931	244	**237**	238	238	238	238	**237**	238	**237**	241
6	237	0,00934	239	240	**237**	240	244	240	240	**237**	**237**	**237**
7	438	0,00509	**438**	**438**	440	**438**	**438**	445	**438**	**438**	**438**	**438**
8	267	0,00393	269	**267**	269	**267**	**267**	269	**267**	269	**267**	269
9	267	0,00356	**267**	269	269	269	**267**	269	269	269	269	269
10	390	0,01080	399	396	**390**	399	**390**	**390**	396	**390**	399	**390**
11	246	0,01085	249	250	251	250	252	**246**	252	253	255	247
12	246	0,01071	**246**	253	249	249	248	247	249	254	**246**	249
13	440	0,01532	451	446	459	448	453	448	453	**440**	465	452
14	260	0,02033	**260**	265	266	267	274	266	**260**	275	**260**	269
15	260	0,02054	269	**260**	274	265	266	275	265	**260**	269	274