HABERLEŞME ŞEBEKELERİNİN TASARIMINDA SEZGİSEL YAKLAŞIMLAR: DEĞİŞKEN KOMŞU ARAMA, KUŞ SÜRÜSÜ OPTİMİZASYONU, KARINCA KOLONİSİ OPTİMİZASYONU

Önder BELGİN

YÜKSEK LİSANS TEZİ
ENDÜSTRİ MÜHENDİSLİĞİ

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

TEMMUZ 2007
ANKARA

Önder BELGİN tarafından hazırlanan HABERLEŞME ŞEBEKELERİNİN TASARIMINDA SEZGİSEL YAKLAŞIMLAR: DEĞİŞKEN KOMŞU ARAMA, KUŞ SÜRÜSÜ OPTİMİZASYONU, KARINCA KOLONİSİ OPTİMİZASYONU adlı bu tezin Yüksek Lisans tezi olarak uygun olduğunu onaylarım.

Prof. Dr. Berna DENGİZ
Tez Yöneticisi

Prof. Dr. Fulya ALTIPARMAK
Tez Yöneticisi

Bu çalışma, jürimiz tarafından oy birliği ile Endüstri Mühendisliği Anabilim Dalında Yüksek lisans tezi olarak kabul edilmiştir.

Başkan: : Prof. Dr. Zülal GÜNGÖR

Üye : Prof. Dr. Fevzi KUTAY

Üye : Prof. Dr. Berna DENGİZ

Üye : Prof. Dr. Fulya ALTIPARMAK

Üye : Yrd. Doç Dr. Ergün ERASLAN

Tarih :18/07/2007

Bu tez, Gazi Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygundur.

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada orijinal olmayan her türlü kaynağa eksiksiz atıf yapıldığını bildiririm.

Önder BELGİN

HABERLEŞME ŞEBEKELERİNİN TASARIMINDA SEZGİSEL YAKLAŞIMLAR: DEĞİŞKEN KOMŞU ARAMA, KUŞ SÜRÜSÜ OPTİMİZASYONU, KARINCA KOLONİSİ OPTİMİZASYONU
(Yüksek Lisans Tezi)

Önder BELGİN

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
Temmuz 2007

ÖZET

Güvenilirlik kısıtı altında minimum maliyetli haberleşme şebekelerinin topolojik eniyilemesi problemi NP-zor bir problemdir. Literatürde bu problemin çözümü için farklı genel amaçlı sezgisel yöntemler kullanılmıştır. Bu çalışmada ise güvenilirlik kısıtı altında minimum maliyetli haberleşme şebekelerinin topolojik eniyilemesi probleminin çözümünde değişken komşu arama, kuş sürüsü eniyilemesi ve karınca kolonisi eniyilemesi genel amaçlı sezgisellerine dayalı algoritmalar geliştirilmiştir. Geliştirilen algoritmaların performansları çözüm zamanı ve çözüm kalitesi açısından karşılaştırılmıştır ve kuş sürüsü eniyileme yaklaşımına dayalı olarak geliştirilen algoritmanın performansının diğerlerine göre daha iyi olduğu gözlemlenmiştir.

Bilim Kodu : 906.1.053
Anahtar Kelimeler : Şebeke tasarımı, Değişken Komşu Arama, Kuş Sürüsü Eniyilemesi, Karınca Kolonisi Eniyilemesi
Sayfa Adedi : 137
Tez Yöneticisi : Prof. Dr. Berna DENGİZ - Prof. Dr. Fulya ALTIPARMAK

HEURISTIC APPROACHES TO DESIGN OF COMMUNICATION NETWORKS: VARIABLE NEIGHBORHOOD SEARCH, PARTICLE SWARM OPTIMIZATION, ANT COLONY OPTIMIZATION

(M.Sc. Thesis)

Önder BELGİN

GAZİ UNIVERSITY
INSTITUTE OF SCIENCE AND TECHNOLOGY

July 2007

ABSTRACT

Topological optimization of minimum cost telecommunication networks subject to reliability constraint is an NP-hard problem. In literature there are diffrent metaheuristics to solve this problem. In this study, new algorithms based on variable neighborhood descent, particle swarm optimization and ant colony optimization have been developed to solve the topological optimization of communication networks under reliability constraint. When new algorithms are compared in terms of solution quality and computation burden it is seen that the algorithm based particle swarm optimization outperforms other algorithms.

Science Code : 906.1.053
Keywords : Network design, variable neighbourhood descent, particle swarm optimization, ant colony optimization
Page Number : 137
Adviser : Prof. Dr. Berna DENGİZ - Prof. Dr. Fulya ALTIPARMAK

TEŞEKKÜR

Çalışmalarımda kıymetli tecrübelerinden yararlandığım sayın Prof. Dr. Berna DENGİZ'e, değerli yardım ve katkılarıyla beni yönlendiren sayın Prof. Dr. Fulya ALTIPARMAK'a ve manevi desteklerini hiçbir zaman esirgemeyen aileme ve Milli Prodüktivite Merkezi'ndeki çalışma arkadaşlarıma sonsuz teşekkürlerimi sunarım.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

Çizelge **Sayfa**

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış bazı simge ve kısaltmalar açıklamaları ile aşağıda verilmiştir.

Simgeler	Açıklama
$\mathbf{p_{ij}}$	*i* ve *j* düğümlerineden oluşan hattın çalışma olasılığı
$\mathbf{q_{ij}}$	*i* ve *j* düğümlerineden oluşan hattın arızalanma olasılığı
$\mathbf{V}$	grafta yer alan düğümlerin kümesi
$\mathbf{E}$	grafta yer alan hatların kümesi
$\mathbf{R(G)}$	*G* grafının güvenilirliği
$\mathbf{d_i}$	*i* düğümünün derecesi
$\mathbf{N_i}$	şebekede yer alan arızalı hat sayısı
$\mathbf{N_k}$	değişken komşu arama algoritmasında kullanılan komşuluk yapılarının kümesi
$\mathbf{N_k(s)}$	*k* komşuluk yapısıyla elde edilen çözümlerin kümesi
$\mathbf{x_i(t)}$	*i* kuşunun t anında arama uzayındaki pozisyonu
$\mathbf{v_i(t)}$	*i* kuşunun t anındaki hızı
$\mathbf{c_1}$	bilişsel bileşen
$\mathbf{c_2}$	sosyal bileşen
$\mathbf{r_1}$	bilişsel bileşenin katkısını ölçekleyen rassal sayı
$\mathbf{r_2}$	sosyal bileşenin katkısını ölçekleyen rassal sayı
$\mathbf{y_i}$	*i* kuşunun en iyi pozisyonu
$\mathbf{n_s}$	sürünün büyüklüğü
$\mathbf{\hat{y}}$	global en iyi değer

Simgeler	Açıklama
$\mathbf{w}$	eylemsizlik ağırlığı
$\hat{\mathbf{y}}_{ij}$	*j* boyutundaki *i*kuşunun komşusu tarafından elde edilmiş en iyi değer
$\mathbf{V_{max}}$	parçacığın ulaşabileceği maksimum hz değeri
$\boldsymbol{\rho}$	*i* ve *j* düğümlerinden oluşan hattın feromen düzeyi feromen buharlaşma oranı
$\boldsymbol{\eta}$	sezgisel değer
$\mathbf{d_{ij}}$	*i* ve *j* düğümlerinden oluşan hattın uzunluğu
$\boldsymbol{\alpha}$	feromen düzeyinin göreceli ağırlığı
$\boldsymbol{\beta}$	sezgisel bilginin göreceli ağırlığı
$\boldsymbol{\Delta\tau_{ij}}$	*i* ve *j* düğümlerinden oluşan hatta eklenen feromen miktarı
$\mathbf{q_0}$	karınca kolonisi sistemi'nde rota seçimi karar indeksi
$\mathbf{q}$	karınca kolonisi sistemi'nde her iterasyonda rota seçiminde kullanılan rassal değer
$\boldsymbol{\sigma}$	amaç fonksiyonun göreceli ağırlığı
$\mathbf{N}$	şebekedeki toplam düğüm sayısı
$\mathbf{c_{ij}}$	*i* ve *j* düğümleri arasındaki hattın maliyeti
$\mathbf{x_{ij}}$	*i* ve *j* düğümlerinden oluşan aday şebeke
$\mathbf{R(x)}$	*x* aday şebekesinin güvenilirliği
$\mathbf{R_0}$	istenen minimum güvenilirlik düzeyi

Kısaltmalar	Açıklama
DKA	değişken komşu arama algoritması
DKİ	değişken komşu iniş algoritması

Kısaltmalar	Açıklama
TDKA	temel değişken komşu arama algoritması
İDKA	indirgenmiş değişken komşu arama algoritması
GDKA	genel değişken komşu arama algoritması
TL_DKİ	tabu listeli değişken komşu arama
KSE	Kuş Sürüsü Eniyilemesi
TB	tavlama benzetimi algoritması
KSE_D	dinamik *w* değerine sahip KSE algoritması
KSE_TB_1	her iterasyonda sürüdeki kuşlara belli olasılıkla TB uygulanan KSE algoritması
KSE_D_TB_1	her iterasyonda sürüdeki parçacıklara belli olasılıkla TB uygulanan ve dinamik *w* değerine sahip KSE
KSE_TB_{DBS}_1	her iterasyonda sürüdeki parçacıklara belli olasılıkla başlangıç sıcaklığı her defasında belli oranda düşürülek TB uygulanan KSE
KSE_D_TB_{DBS}_1	her iterasyonda sürüdeki parçacıklara belli olasılıkla başlangıç sıcaklığı her defasında belli oranda düşürülek TB uygulanan ve dinamik *w* değerine sahip KSE
KSE_TB_2	global en iyi değerde iyileşme olduğunda TB'nın uygulandığı KSE
KSE_D_TB_2	global en iyi değerde iyileşme olduğunda TB'nın uygulandığı dinamik w değerine sahip KSE
KSE_TB_{DBS}_2	global en iyi değerde iyileşme olduğunda her seferinde başlangıç sıcaklığının belli oranda düşürüldüğü TB'nın uygulandığı KSE
KSE_D_TB_{DBS}_2	global en iyi değerde iyileşme olduğunda her seferinde başlangıç sıcaklığının belli oranda düşürüldüğü TB'nın uygulandığı dinamik *w* değerine sahip KSE
KKE	karınca kolonisi optimizasyonu algoritması
DKİ	değişken komşu iniş algoritması

Kısaltmalar	Açıklama
TB_KKE$_D$	TB ile başlangıç çözümü elde edilen ve dinamik q_0 değerine sahip KKE
KKE_TB	global en iyi değerde iyileşme meydana geldiğinde TB uygulanan KKE
KKE$_D$_TB	global en iyi değerde iyileşme meydana geldiğinde TB uygulanan dinamik q_0 değerine sahip KKE
KKE_TB$_{DBS}$	global en iyi değerde iyileşme meydana geldiğinde her seferinde başlangıç sıcaklığı belli oranda düşürülerek TB uygulanan KKE
KKE$_D$_TB$_{DBS}$	global en iyi değerde iyileşme meydana geldiğinde her seferinde başlangıç sıcaklığı belli oranda düşürülerek TB uygulanan dinamik q_0 değerine sahip KKE
TB_KKE_TB	TB ile başlangıç çözümü elde edilen ve global en iyi değerinde iyileşme meydana geldiğinde TB uygulanan KKE
TB_KKE$_D$_TB	TB ile başlangıç çözümü elde edilen ve global en iyi değerde iyileşme meydana geldiğinde TB uygulanan dinamik q_0 değerine sahip KKE
TB_KKE_TB$_{DBS}$	TB ile başlangıç çözümü elde edilen ve global en iyi değerde iyileşme meydana geldiğinde her seferinde başlangıç sıcaklığı belli oranda düşürülerek TB uygulanan KKE
TB_KKE$_D$_TB$_{DBS}$	TB ile başlangıç çözümü elde edilen ve global en iyi değerde iyileşme meydana geldiğinde her seferinde başlangıç sıcaklığı belli oranda düşürülerek TB uygulanan dinamik q_0 değerine sahip KKE

1. GİRİŞ

Günümüzde bilgi teknolojileri çok hızlı bir şekilde gelişmekte, "bilgi toplumu", "enformasyon otobanı" gibi kavramlar günlük hayatımızda sıklıkla duyduğumuz kavramlar haline gelmektedir. İster kişisel kullanıcılar olsun, ister global ölçekte iş yapan firmalar olsun bilgi teknolojileri ve dolayısıyla da bilgisayar şebekeleri herkes için vazgeçilmez bir unsur haline gelmiştir. İhtiyaç duyulduğunda bilgiye minimum maliyetli ve güvenilir biçimde ulaşabilmek bilgisayar şebekeleri için temel bir performans boyutu haline gelmiştir.

Söz konusu şebekeler olduğunda birimler arasındaki haberleşmenin sağlanabilmesi için öncelikle topolojik tasarımın gerçekleştirilmesi gerekmektedir. Şebekelerin topolojik olarak tasarlanması, haberleşme amacıyla kullanılan cihazların ya da bilgisayarların birbirine nasıl bağlanacağını belirler. Bu tasarım gerçekleştirilirken maliyet, güvenilirlik, kapasite gibi performans ölçütlerinin göz önüne alınması gerekir. Bu kriterler ister omurga (backbone) olarak adlandırılan ana erişim şebekelerinin tasarımında olsun, ister daha küçük alana yayılan yerel erişim şebekelerinde olsun mutlaka dikkate alınması gereken kriterlerdir.

Bir ofis içerisinde kurulan yerel alan şebekesinde ya da ülkeler arasında yayılmış olan geniş alan şebekelerinde haberleşmenin kesintisiz bir biçimde gerçekleştirilebilmesi güvenilir şebekelerin tasarlanması gerekir. Güvenilir şebekeler tasarlarken maliyet unsurunu da göz önüne almak gerekmektedir. Bu açıdan güvenilirlik ve maliyet gibi birbirleriyle çelişen amaçların ortak bir noktada buluşturulması şebekelerin topolojik tasarımını zor bir problem haline getirmektedir. Bu tezde, güvenilirlik kısıtı altında minimum maliyetli ana erişim (omurga) şebekelerinin topolojik tasarımı problemi üzerine çalışılmıştır. Ana erişim şebekelerinde göz önüne alınan güvenilirlik kriteri belirlenen birkaç adet terminal arasında haberleşmenin gerçekleşme olasılığı (s-t terminal güvenilirliği) veya şebekede yer alan tüm terminaller arasındaki haberleşmenin (tüm terminal güvenilirliği) gerçekleşmesi olasılığı üzerine kurulabilir. Bu çalışmada ise tüm terminal güvenilirliği dikkate alınmıştır.

Bu problemin çözümünde karşılaşılan önemli güçlüklerden biri de güvenilirliğin hesaplanması aşamasındadır. Bir şebekenin güvenilirliğini hesaplama güçlüğü şebeke boyutu arttıkça artmaktadır. Bu nedenle şebeke güvenilirliğinin tam değerinin hesaplanması yerine güvenilirliğin alt ve üst sınırlarının hesaplanması ya da şebeke güvenilirliğinin tahmin edilmesi yöntemlerinden birinin kullanılması çözüm zamanının azaltılmasına yardımcı olmaktadır. Bu çalışmada şebeke güvenilirliğinin tahmininde Monte Carlo benzetimi kullanılmıştır.

Güvenilirlik kısıtı altında minimum maliyetli haberleşme şebekelerinin topolojik tasarımı problemi NP-zor sınıfa giren bir problem türüdür. Bu problemin çözümünde probleme özgü geliştirilen sezgisel yöntemlerin yanında genel amaçlı sezgisel yöntemler yoğunlukla kullanılmaktadır. Genel amaçlı sezgisel yöntemler, kombinatoryal optimizasyon problemlerinde akıllı arama stratejileri kullanarak çözüm uzayında arama yapan yöntemlerdir. Genel amaçlı sezgisel yöntemlere örnek olarak rassal arama, tepe tırmanma, tabu arama, tavlama benzetimi, genetik algoritma, parçacık kuş sürüsü eniyilemesi karınca kolonisi eniyilemesi, değişken komşu arama gibi yöntemler verilebilir. Bu yöntemlerin NP-zor sınıfa giren problemlerin çözümünde başarılı sonuçlar verdiği çeşitli araştırmalar tarafından gösterilmiştir. Bu genel amaçlı sezgisel yöntemler, problemlerin çözümünde tek başına kullanılmalarının yanı sıra birkaç yöntemin aynı algoritma içerisinde kullanıldığı veya algoritmalardaki çeşitli yapıların birleştirilerek karma algoritmaların da geliştirilmesi söz konusu olabilmektedir. Bu çalışmada, değişken komşu arama, kuş sürüsü eniyilemesi ve karınca kolonisi eniyilemesi algoritmalarına dayalı yeni algoritmalar geliştirilmiştir. Bu algoritmaların seçilmesinin nedeni ise güvenilirlik kısıtı altında minimum maliyetli haberleşme şebekelerinin topolojik tasarımı probleminde bu yöntemlerin kullanılmamış olması ve farklı problemlerde iyi performans göstermiş olmalarıdır.

Bu tez, 9 ana bölümden oluşmaktadır. Birinci bölümde yapılan kısa bir girişten sonra ikinci bölümde bilgisayar şebekeleri ve şebeke tasarımı ve şebeke güvenilirliği konularında bilgiler verilmekte, güvenilirlik kısıtı altında minimum maliyetli şebekelerin topolojik tasarımı probleminin karmaşıklığı ele alınmaktadır. Üçüncü

bölümde ise literatür incelemesi gerçekleştirilmiştir. Dördüncü, beşinci ve altıncı bölümlerde bu çalışmanın konusu olan problemin çözümünde kullanılan genel amaçlı sezgisellere ilişkin açıklamalara yer verilmiştir. Yedinci bölümde ise ele alınan genel amaçlı sezgisellerin güvenilirlik kısıtı altında minimum maliyetli şebekelerin topolojik tasarımı probleminde nasıl kullanıldığı ele alınmıştır. Sekizinci bölümde kullanılan genel amaçlı sezgiseller için uygun parametrelerin seçimi gerçekleştirilmiş ve bu genel amaçlı sezgisellerle elde edilen sonuçların değerlendirilmesi yapılmıştır. Sonuç bölümünde ise çalışmaya ilişkin genel bir değerlendirmeye yer verilmiştir.

2. BİLGİSAYAR ŞEBEKELERİNİN TOPOLOJİK TASARIMI

Bulunduğumuz çağda en hızlı gelişmeler bilginin elde edilmesi, işlenmesi ve dağıtımında kullanılan bilgi teknolojileri alanında yaşanmaktadır. Gelişen teknolojiye bağlı olarak hızlı ve güvenilir şekilde bilgiye ulaşma ve bu bilgiyi paylaşmada kullanılan teknolojiler alanında sürekli bir gelişme söz konusudur. Bilginin elde edilmesinde ve paylaşımında kullanılan bilgisayar şebekeleri artık günümüzün vazgeçilmez unsurları haline gelmişlerdir. Bireysel kullanıcılardan organizasyonlara kadar bilgi iletişimi hiyerarşisinde yer alan tüm kademeler bu şebekelerle bütünleşmektedirler.

2.1. Bilgisayar Şebekeleri

Bilgisayar şebekeleri, birden fazla bilgisayarın iletişim kurmak veya bilgi paylaşmak amacıyla bir haberleşme sistemi kullanarak birbirine bağlanmasıyla meydana gelirler. Bilgisayarları bir şebekeye bağlı olan kişiler bu şebeke üzerinden hem bilgi paylaşımı hem de çevre birimlerinin paylaşımını gerçekleştirebilirler. Farklı noktalardaki ağlar, bir telefon servis sağlayıcısı ile birbirlerine bağlandıklarında kullanıcılar elektronik posta gönderebilir, global internet bağlantılarını paylaşabilir veya ağı paylaşanlar arasında video konferans gibi uygulamalar gerçekleştirebilirler. Bir bilgisayar şebekesinin bileşenleri:

- en az iki adet bilgisayar,
- her bilgisayarda bir şebeke ara yüzü,
- bir bağlantı aracı (genellikle kablolar- fakat şebekeye dahil olmuş bilgisayarlar arasında kablosuz iletişim de mümkün olabilmektedir),
- şebeke işletim sistemi yazılımı [1].

2.2. Bilgisayar Şebekelerinin Türleri

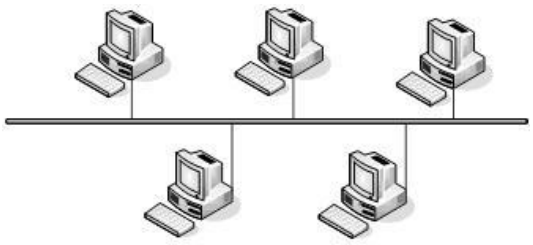
Genel kabul görmüş bir sınıflandırma biçimi bulunmasa da bilgisayar şebekelerini iki boyutu göz önüne alarak sınıflandırmak mümkündür: *İletim teknolojisi* ve *ölçek* [2].

Bilgisayar Şebekelerinin İletim Teknolojisine Göre Sınıflandırılması

Genel olarak iki tür iletim teknolojisinden bahsetmek mümkündür:

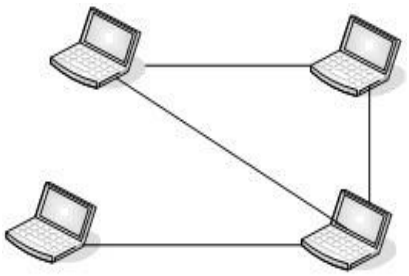
- Yayın (broadcast) bağlantıları
- Noktadan noktaya bağlantılar

Yayın ağları, şebeke üzerinde yer alan tüm makineler tarafından paylaşılan tek bir haberleşme kanalına sahiptir. Her hangi bir bilgisayardan gönderilen kısa mesajlar (paketler) diğer bilgisayarlar tarafından alınır. Gönderilen paket içinde bu paketi alması istenen bilgisayara ait bir adres alanı da yer alır. Bir paket alındığında bilgisayara adres alanını kontrol eder ve eğer paket ulaşması istenen bilgisayara gönderilmişse alınan paket bu bilgisayar tarafından işlenir, tersi durumda paket göz ardı edilir [3].



Şekil 2.1. Yayın şebekesi

Noktadan noktaya ağlar, kişisel bilgisayar çiftleri arasında bir çok bağlantıdan meydana gelirler. Bu tür şebekelerde bir paket kaynaktan hedefe ulaşabilmek için bir veya daha fazla ara bilgisayarı ziyaret etmek zorundadır. Bu şebekelerde sıklıkla çoklu rotalar, farklı uzunluklar mümkündür. Bu nedenle iyi rotalar elde edebilmek noktadan noktaya şebekelerde önemli bir problemdir [2].



Şekil 2.2. Noktadan noktaya şebeke

Bir çok istisnası olsa da genel bir kural olarak daha küçük, coğrafi olarak konumlandırılmış şebekeler yayın şebekeleri şeklinde, buna karşın daha büyük şebekeler noktadan noktaya şebekeleri kullanmaktadırlar [2].

Bilgisayar Şebekelerinin Ölçeklerine Göre Sınıflandırılması

Bilgisayar ağlarını sınıflandırmada kullanılan diğer bir kriter ölçektir. Bu kısımda bahsedilecek olan bilgisayar şebeke türleri şunlardır [2]:

- Kişisel Alan Şebekeleri (Personal Area Networks – PAN)
- Yerel Alan Şebekeleri (Local Area Networks – LAN)
- Metropolitan Alan Şebekeleri (Metropolitan Area Networks – MAN)
- Geniş Alan Şebekeleri (Wide Area Networks – WAN)
- Internet

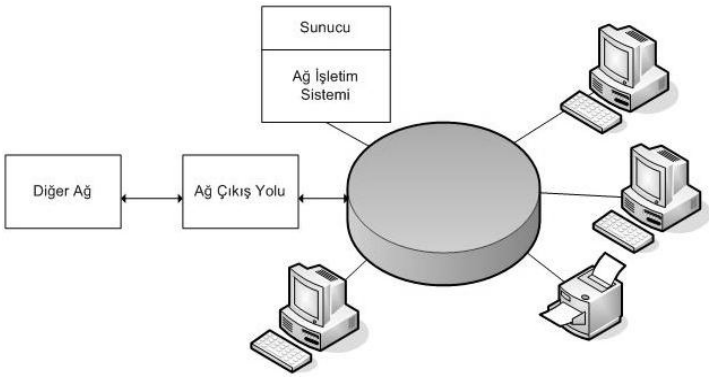
Kişisel Alan Şebekeleri

Bu şebekeler bir kişiyi kapsayan şebekelerdir. Örneğin bir bilgisayarla kablosuz şebekeye bağlanan kişinin oluşturduğu şebeke kişisel alan şebekesidir.

Yerel Alan Şebekeleri

Yerel alan şebekeleri (Local Area Networks-LAN), bilgisayarları, çevre birimlerini veya diğer cihazları bir bina içerisinde veya sınırlı bir alanda birbirine bağlayan

şebekelerdir. Bu şebekeler, genellikle ofislerde ya da fabrikalarda bulunan kişisel bilgisayarlar ve iş istasyonları arasında kaynak paylaşımı ve bilgi alışverişi gerçekleştirmek amacıyla kullanılırlar. Bu tür şebekeler boyut bakımından sınırlı olduklarından yönetilmeleri kolaydır.



Şekil 2.3. Yerel alan Şebekesi [3]

Metropolitan Alan Şebekeleri

Bir ağ, birçok durumda karmaşık bir fiziksel yerleşime dağılmış olabilir. Metropolitan alan şebekeleri (MAN) uygulaması genelde bu tür geniş bir alana yayılmış olan sistemi alt şebekeler ayrıştırarak entegre etme yaklaşımına dayanmaktadır.

MAN, LAN'ın kapsadığı alandan daha geniş, fakat WAN'ın kapsadığından daha dar mesafeler arası iletişimi sağlamakta ve genellikle şehir içi şebekelerin birbirine bağlanmasıyla oluşturulmaktadır [3].

Geniş Alan Şebekeleri

Bir geniş alan şebeke (wide Area Network-WAN), bir ülke veya bir kıta gibi geniş bir coğrafi alana yayılır. Geniş alan şebekeleri; yerel alan şebekeleri ve diğer şebeke türlerini birbirlerine bağlamakta kullanılırlar. Böylelikle farklı yerlerdeki bilgisayarlar ve kullanıcılar birbirleriyle haberleşebilmektedirler.

Internet

Dünyada farklı yazılım ve donanıma sahip birçok şebeke bulunmaktadır. Bir şebekeye bağlanan kullanıcılar diğer şebekelere bağlı kullanıcılarla da haberleşmek istemektedirler. Yazılım ve donanım bakımından birbirinden farklı bu şebekeler arasında haberleşme talebini karşılamak amacıyla geçit (gateway) adı verilen cihazlar kullanılır. Birbiriyle bağlanmış şebekelerden meydana gelen bu yapılara internet adı verilir.

2.3. Şebeke Topolojileri

Bilgisayar şebekelerinde topoloji, şebekedeki cihazların birbirlerine nasıl bağlandığını tanımlar. Şebeke topolojileri *fiziksel* ve *mantıksal* olarak ikiye ayrılırlar. *Fiziksel topoloji*, bir şebekenin cihazlar, yer ve kablo kurulumunu içeren fiziksel tasarımıdır. *Mantıksal topoloji* ise şebekenin tasarımından bağımsız olarak verinin bu şebekede nasıl iletildiğini ifade eder. Topoloji bir şebekenin sanal şekli veya yapısı olarak da düşünülebilir.

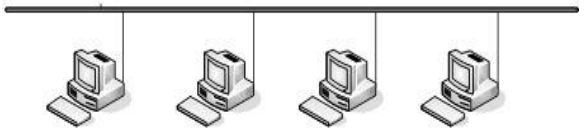
Fiziksel açıdan bilgisayar şebeke topolojileri aşağıdaki biçimde sınıflandırılabilir:

1. Doğrusal Topoloji
2. Yıldız Topoloji
3. Halka Topoloji
4. Ağ Topoloji
5. Ağaç Topoloji

2.3.1. Doğrusal topoloji

Doğrusal topolojiler, bütün şebeke cihazlarının omurga (backbone) adı verilen bir kabloya doğrusal bir biçimde bağlandığı topolojilerdir İletişim kurmak isteyen cihaz şebekedeki tüm cihazlara paylaşılan kablo aracılığıyla veri yollar ve bu verinin ulaşması istenen cihaz veriyi alarak işler. Şebekeyi kontrol eden merkezi bir

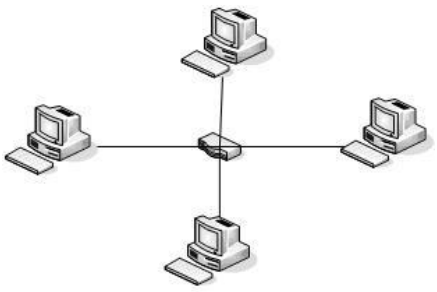
bilgisayar bulunmamaktadır [3]. Bu tür topolojilerde ağın toplam uzunluğu 185 metreyi geçememektedir ve standartları 30 düğümden fazlasına imkan vermemektedir. Tek bir kablo üzerinden iletişim kurmasından dolayı veri iletimi sırasında bir noktada sorun meydana geldiğinde tüm ağ etkilenmekte ve arızanın bulunması da zor olmaktadır. Bunların yanında doğrusal topolojilerin kurulumu daha kolaydır ve daha az kablo gerektirmektedir.



Şekil 2.4. Doğrusal topoloji

2.3.2. Yıldız topoloji

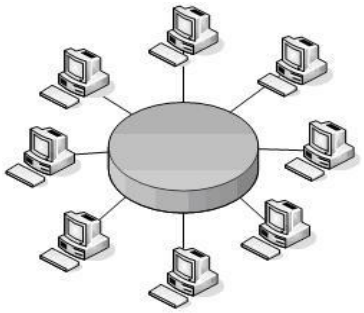
Yıldız topoloji, yerel alan şebekelerinde en sık kullanılan türdür. Bir yıldız topolojide tüm bilgisayarlar yönlendirici (router), anahtar (switch) ya da tekrarlayıcı (hub) olarak isimlendirilen merkezi bir aygıta bağlanır. Ağda bulunan her bilgisayar merkezi aygıta bağımsız şekilde bağlanır. Merkezi aygıt, şebekeye bağlı bilgisayarların iletişimini kontrol ettiğinden bu aygıtın arızalanması tüm şebekedeki iletişimin kopmasına neden olacaktır. Diğerlerinin çalışmasını aksatmadan her aygıtın ayrı olarak bağlanabilmesi, hata giderme kolaylığı, ortam hatalarının oluştuğu bölümde izole edilmesi gibi avantajlarının yanı sıra, kurulumunun zor olması ve genellikle diğer topolojilerden daha fazla kablo gerektirmesi gibi bir takım dezavantajlara sahiptir [3].



Şekil 2.5. Yıldız topoloji

2.3.3. Halka topoloji

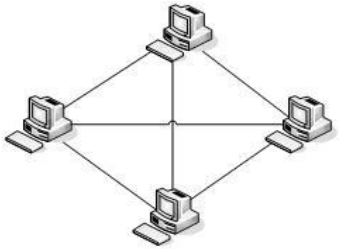
Halka topolojide şebekedeki tüm aygıtlar diğer aygıtlara kapalı bir çevrim şeklinde bağlıdır, böylece her aygıt her iki yanındaki aygıta direkt olarak bağlanmış olur. Mesajlar bilgisayardan bilgisayara kapalı döngü boyunca genelde tek yönde gönderilirler. Bu topolojide merkezi bir aygıta ihtiyaç duyulmamaktadır. Şebekedeki her bilgisayar bağımsız çalışmakta ve ağdaki bilgisayarların arızalanması şebekeyi etkilememektedir [3].



Şekil 2.6. Halka topoloji

2.3.4. Ağ topoloji

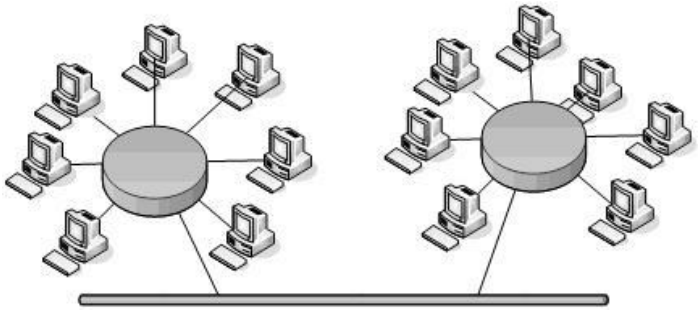
Şebekede bulunan aygıtlar diğerlerine kablolarla direkt olarak bağlanırlar. İdeal bir ağ topolojide bir düğüm şebekedeki diğer tüm düğümlerle bağlantıya sahiptir. Ağ topolojide kaynaktan gönderilen mesaj hedefe ulaşmak için en kısa, en kolay yolu seçebilir. İnternet, ağ topolojileri kullanır. Şebekedeki tüm aygıtların birbirlerine direkt olarak bağlandığı topolojiye *tam ağ topoloji*, aygıtların endirekt olarak birbirlerine bağlandığı topolojiye *kısmi ağ topoloji* adı verilir.



Şekil 2.7. Ağ topoloji

2.3.5. Ağaç topoloji

Ağaç topoloji, birden fazla yıldız topolojinin doğrusal hat üzerine konumlandırılmasıdır. Bu topolojinin karma yapısı şebekenin genişletilebilirliğine yıldız veya doğrusal topolojiden daha iyi olanak sağlar.



Şekil 2.8. Ağaç topoloji

2.4. Şebeke Tasarımı

Şebeke tasarımı, coğrafik olarak dağıtılmış yerler, bu yerler arasındaki trafiğe ait bilgi ve bu yerlerin nasıl bağlanabileceklerine ilişkin bilgi doğrultusunda aday bağlantıların seçilmesidir [7].

Genel olarak şebeke tasarım problemi üç temel bileşenden oluşur:

1. Çevresel Durumlar: Sunucular, servis sağlayıcılar, terminaller ve diğer bitiş düğümlerinin konumunu; çevrenin tasarlanan trafiği ve farklı servis seviyelerinin tahmin edilen maliyetlerini içerir.
2. Performans kısıtları: Şebeke güvenilirliği, trafik akışı, sunucu/istemci bilgisayarlarının hızlarını içine alır.
3. Şebekeler arası değişkenler: Şebeke topolojisi ve hat kapasitelerini içerir.

Şebeke tasarımında amaç, elementlere bağlı maliyetleri minimize ederek hizmet sunarken belirlenen var olma gerekliliklerine çelişmeyen bir tasarım elde etmektir. Bu noktada iki önemli kavramla karşı karşıya kalınmaktadır: *var olma (güvenilirlik)*

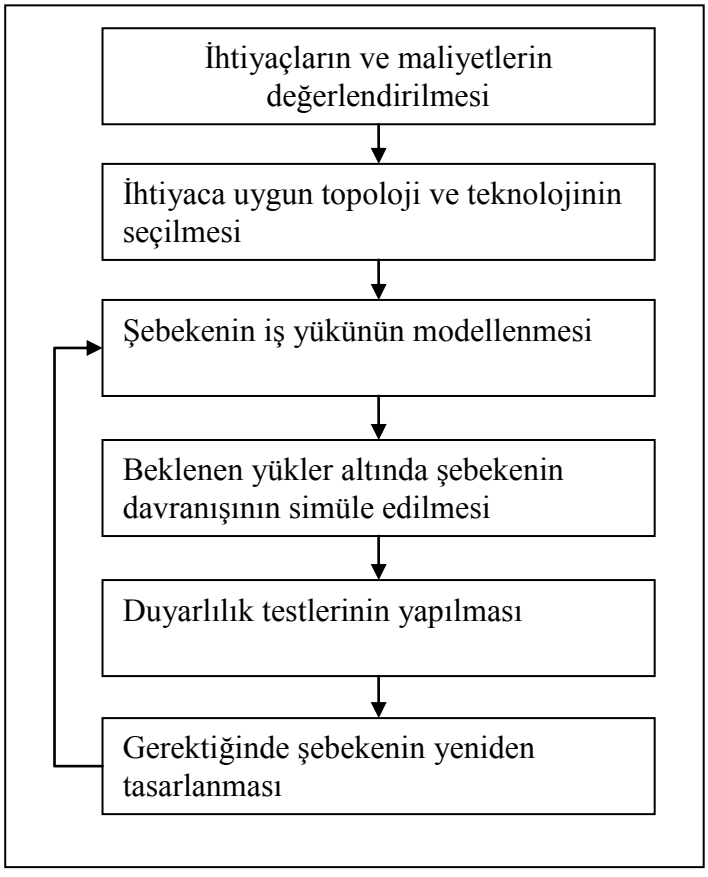
ve *maliyet*. Güvenilirlilikteki herhangi bir artış maliyetlerde artış şeklinde yansıma bulacaktır. Sonuç olarak, güvenilirlilik ve maliyet dikkatli biçimde dengelenmelidir [5].

Şebeke tasarımının adımları Şekil 2.9'da özetlenmektedir [5].

Genel bir şebeke tasarımı yerel ve ana erişim şebekelerinin tasarımı olmak üzere iki ana grupta incelenebilir.

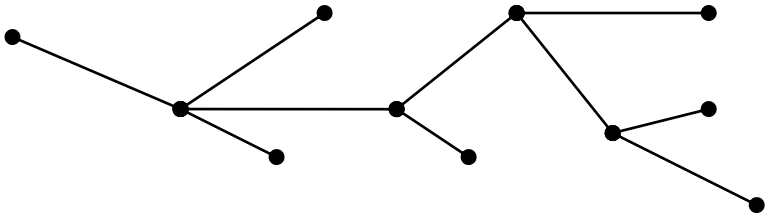
2.4.1. Yerel erişim şebekelerinin tasarımı

Yerel erişim şebekelerinde bilgisayar terminalleri çok noktalı hatlarla merkezi bir noktaya bağlanırlar. Bu şebekelere merkezi şebekeler de denir. Bu çok noktalı şebeke



Şekil 2.9. Genel şebeke tasarım süreci [5]

formları graf teorisinde *ağaç* olarak adlandırılır. Bir ağaç hiçbir çevrimin bulunmadığı ve her hangi bir düğüm çiftini bağlayan yalnızca bir hattın bulunduğu şebekedir. Şekil 2.10'da bir ağaç şebeke görünmektedir.



Şekil 2.10. Ağaç şebeke

Bir ağacın yine ağaç meydana getiren alt parçasına *dal* adı verilir. Güvenilir ağaç şebekelerin tasarımında kullanılan yöntemlerden birisi *minimum yayılan ağaç* elde etmektir. Minimum yayılan ağaç verilen bir düğüm seti için en az toplam uzunluğa sahip bir ağaçtır. Minimum yayılan ağaç elde etmek için Prim ve Kruskal algoritmaları geliştirilmiştir. Fakat güvenilir bir şebekenin tasarımında minimum yayılan ağaç noktalar arasındaki uzaklık verileri yerine bir düğüm çiftleri arasındaki hatların bozulma olasılığı verileriyle çalışır. Hatların birbirlerinden bağımsız şekilde arızalandığını düşünürsek ağacın çalışmaması olasılığı $1-\prod_{l=1}^{L}(1-p_l)$'dir. Bu ifadede L, şebekedeki hatların sayısı; p_l ise l hattının arızalanma olasılığıdır. Bu durumda problem, şebekenin arızalanması durumunu minimize edecek tasarımı elde etmektir.
Merkezi şebekelerin tipik uygulamalarında veri trafiği merkezi bir işlemci ve uzak noktalardaki istasyonlar arasında akar. Yalnızca hat uzunlukları ele alındığında minimum yayılan ağaç algoritması herhangi bir merkezi düğüm için en iyi tasarımı elde eder. Bunun yanında elde edilen konfigürasyon trafiği taşıyacak hat kapasitelerini garanti etmez.

Herhangi bir hattın aşırı yüklenmesini önlemek için bir hatta akacak trafik hacmi için kapasite kısıtı konabilir. Bir şebeke için güvenilirlik göz önüne alındığında ağacın bir dalındaki düğüm sayısının kısıtlanması gerekebilir. Bir hat üzerindeki toplam trafik ve bir daldaki düğüm sayısı kısıtı birlikte dikkate alınabilir. Kısıtlı en iyi yayılan

ağaç algoritmaları *dal sınır* tekniklerinin varyasyonudur. Kısıtlı en iyi yayılan ağaç algoritmaları en iyi çözümü elde etmelerine karşın düğüm sayısı arttıkça çözüm için gereken zamanda artış çok fazladır. Bu nedenle birkaç adımda en iyiye yakın çözüm üreten algoritmalar geliştirilmiştir. Bunlardan biri *Easu-Williams Algoritması*'dır [6].

2.4.2. Ana erişim şebekelerinin tasarımı

Buraya kadar anlatılan haberleşme şebekeleri trafiğin merkezi bir noktadan sınırlı sayıdaki terminale aktığı şebekelerdi. Bu kısımda bahsedilecek olan şebeke türleri geniş bir alana yayılan şebekelerdir. Bu tür şebekeler yerel dağıtım ağlarının kullandığı merkezi işlemcilerin birbiriyle bağlanmasıyla oluşan omurga şebekelerdir. Bu şebekeler de, geniş bir alana yayılmış kaynaklardan gelen verilerin trafiğinin yönlendirilmesi söz konusu olduğu için güvenilirlik önemli bir kriterdir. Güvenilir bir şebeke için kaynak hedef çiftleri arasında birden fazla yol bulunması gerekir.

Temel topolojik tasarım problemi kaynak hedef düğüm çiftleri arasındaki akışı gösteren trafik matrisiyle başlar. Buradaki amaç, bağlılık ve performans kısıtları altında maliyeti minimize edecek hat ve hat kapasitelerini seçmektir. Geçerli bir performans ölçütü olarak ortalama gecikme örnek verilebilir.

Şebekedeki herhangi bir düğüm, şebekedeki diğer düğümlerden herhangi birine bağlanacağından dolayı bir şebeke tasarımı elde etmek zordur. Bu tür şebekelerin tasarımı için bilinen en iyileme teknikleri sadece küçük boyutlu problemler için kullanılabilmektedir [6]. Dolayısıyla, bu tür problemlerin çözümü için literatürde sezgisel yaklaşımlar önerilmektedir. Bunlar;

- Dal Değiştirme Metodu
- Dış Bükey Dal Eleme Metodu
- Kesme Doyurma Metodu

olarak sıralanabilir.

Dal Değiştirme Metodu

Dal değiştirme metodu geçerli bir topolojiyle çözüme başlar ve hat ekleme, hat kaldırma ya da hat değiştirme hareketleri ile belirlenen amaca doğru ilerlenir. Yapılan hat değişimleri sırasında şebekenin 2-bağlılık ölçüsünün bozulmaması gerekir. Hatların kullanım oranı ve maliyeti elde edilecek topolojiye karar vermede önemli rol oynar. Kısıtlar sağlanıyorsa yeni tasarım mevcut tasarımın yerine geçer ve metot tekrarlanır. Kısıtlar sağlanmıyor ise mevcut tasarıma geri dönülerek yeni bir hat değişimi gerçekleştirilir ve metot tekrarlanır [7].

Dış Bükey Dal Eleme Metodu

Dış bükey dal eleme metodu, başlangıç şebekesi olarak tam bağlı şebekeyi kullanır ve yerel minimuma ulaşıncaya kadar şebekeden hat çıkarır. Bu metodun uygulanabilmesi için tanımlanan maliyet fonksiyonunun dış bükey olması gerekir. Kesikli maliyet fonksiyonu söz konusu ise bu fonksiyonun dış bükey maliyet fonksiyonuna dönüştürülmesi gerekir [7].

Kesme Doyurma Metodu

Bu metot, devre teorisindeki bazı ana kavramlar temel alınarak geliştirilmiştir. İki düğüm arasındaki bir kesme düğümü bağlı olmayan duruma gelinceye kadar kaldırılması gereken hatların kümesidir. Bu hatların kapasitelerinin toplamı da kesmenin kapasitesine eşittir. En düşük kapasiteye sahip olan kesme minimum kesme olarak adlandırılır. Devre teorisinin temel bir teoremi olan *minimum kesme-maksimum akış teorem*ine göre düğüm çiftleri arasındaki maksimum akış düğüm çiftleri arasındaki minimum kesmenin kapasitesini aşamaz.

Bu teorik devre kavramları işlem hacmi kısıtı altında kesme doyurma algoritmasını temellendirir. Bu algoritmadaki temel varsayım, hatların hepsinin aynı kapasiteye sahip olduğudur. Algoritmanın başlangıç iterasyonunda geçerli bir şebeke çözümü mevcuttur. İlk adım doyurulacak kesmenin belirlenmesidir. Şebekenin hatları

taşıdıkları trafiğin hacmine bağlı olarak sıralanırlar. Hatlar en fazla doyurulmuş hattan başlanarak şebeke iki ayrı parçaya ayrılana kadar çıkarılır. Bu hat çıkarma işlemi şebekenin işlem hacminin sınırını belirlemek için yapılır. Algoritmanın bir sonraki adımı doyurulmuş kesme setinden ayrılmış parçaları arasında hat ekleme ve çıkarma işlemidir. Ekleme ve çıkarmalar tamamlandıktan sonra şebeke için en iyi tasarım bulunmuş olur.

Algoritma, yayılan ağaçla veya tam bağlı bir şebekeyle başlatılabilir. Algoritmanın karmaşıklığı son şebeke tasarımındaki düğüm sayısının karesi ve hat sayısının karesinin toplamına eşittir [6].

2.5. Şebekelerin Tasarımında Güvenilirlik

Bir haberleşme şebekesini tasarlarken maliyet, işlem hacmi, performans, bağlılık ve güvenilirlik gibi birbirleriyle çelişen kısıtların göz önüne alınması gerekmektedir. Son on yılda bu kriterler arasında güvenilirlik önemli derecede göz önüne alınır hale gelmiştir. Yüksek kapasiteli dağınık bir şebeke, bileşenlerin (düğümler ve hatlar) çalışmamasına karşı savunmasızdırlar. Tek bir parçanın bile arıza yapması durumunda servis kalitesi büyük zarar görebilir ya da şebekeyi kullananların çoğunun şebekeyle bağlantısının kesilmesine neden olabilir. Bu nedenle, haberleşme şebekelerinin bağımlılığı arttıkça ve topolojiler daha dağınık hale gelmeleri şebeke güvenilirliğini önemli bir kriter haline getirmiştir.

Genel olarak şebeke güvenilirliği, şebeke bileşenlerinin arızası durumunda tüm şebekenin hizmet vermeye devam edebilme yeteneğini tanımlar. Bu problemin en önemli yönü şebekenin güvenilirliğinin hesaplanmasıdır. Şebeke güvenilirliğinin tam değerinin hesaplanması NP-zor bir problemdir. Bu nedenle araştırmacılar şebeke güvenilirliğinin hesaplanmasında etkin algoritmaların geliştirilmesi üzerine yoğunlaşmışlardır. Bu teknikler *tam değer, teorik sınırlar* ve *benzetim* olmak üzere 3 ana başlıkta sınıflandırılabilirler [8].

2.5.1. Şebeke güvenilirliğini modellenmesi

Haberleşme şebekeleri kusursuz olmayan bileşenlerden oluşur. Hatlar ve düğümler arızaya maruz kalabilirler. Bileşenlerin bozulma oranları geçmiş verilere dayanarak elde edilebilir. Güvenilir olmayan bileşenlere sahip haberleşme şebekeleri yönlendirilmemiş olasılıklı şebekeler olarak modellenebilirler: G = (V,E). V, düğümler setini; E ise p_{ij} ve $1\text{-}p_{ij}$ olasılıklarıyla çalışan veya arızalı hatların oluşturduğu kümeyi ifade eder. Bu modelde düğümler yönlendiriciler, bilgisayarlar gibi bileşenleri; hatlar ise bu bileşenleri birbirine bağlayan bağlantıları ifade eder. Bu modelin genel varsayımları şu şekilde sıralanabilir:

- Hatların arızalanmaları bağımsızdır,
- Düğümler tam güvenilirdir,
- Hiçbir tamire izin verilmez.

İlk iki varsayım mantıksız olduğu gerekçesiyle eleştirilmiştir. Gerçekte bileşenlerin arızalarının beraber görülmesine karşın bileşenlerin bağımsız olarak arızalanmaları varsayımı hesaplama izlenebilirliği açısından önemlidir [8].

2.5.2. Şebeke güvenilirlik ölçüleri

Şebeke güvenilirliği ölçülerini temel yaklaşımları iki ana sınıfta toplayabiliriz:

- Belirli (deterministik) ölçüler
- Olasılıklı ölçüler

Belirli ölçüler için sorulan temel soru bir şebekenin bağlı sayılmaması için kaç adet hattın arızalı olması gerekir sorusudur. Olasılıklı yaklaşımda ise hat ve düğümlerin belirli olasılıklarla arıza yapmaları söz konusudur. Burada sorulan temel soru ise şebekenin bağlı sayılmama olasılığı nedir sorusudur [6]. Belirli ölçülerin şebekelerin

tam ölçüsünün olmadığı Wilkov (1972) ile Soi ve Aggarwal (1981) tarafından belirtilmektedir [7].

Güvenilirlikte Belirli (Deterministik) Ölçüler

Bu sınıfa giren güvenilirlik ölçülerini bağlılık, birleşme, çap, çevre ve birleştirme düzeyi olarak sıralayabiliriz [7]:

Bağlılık: Bağlılık, grafın herhangi bir düğüm çifti arasındaki yolları kırmak için kaldırılması gereken hatların ya da düğümlerin minimum sayısıdır. Maksimum güvenilirliğe sahip bir şebeke elde etmek için bağlılık ölçüsünün yüksek olması gerekir. Şebekelerin topolojik tasarımında 2-bağlılık yaygın olarak kullanılan bir güvenilirlik ölçüsüdür.

Birleşme: Birleşme δ(m), bir graftan m düğümlü bir alt graf elde etmek için kaldırılması gereken hatların ya da düğümlerin minimum sayısıdır. Yüksek güvenilirliğe sahip bir şebekenin tasarımında tüm m değerleri için δ(m) değerinin mümkün olduğunca yüksek olması gerekir.

Çap: Bir grafta herhangi iki düğüm arasındaki uzaklık bu iki düğüm arasındaki en kısa yola eşittir. Bir grafın tüm düğüm çiftleri arasındaki en kısa yolların en uzunu bu grafın çapına eşittir. Yüksek güvenilirliğe sahip bir haberleşme şebekesinde çap değerinin olabildiğince küçük olması gerekir.

Çevre: Bir grafta çevrim, başlangıç ve bitiş düğümü aynı olan yoldur. Çevrimin uzunluğu kapalı yoldaki hatların sayısına eşittir. Bir grafın çevresi t(G), graftaki çevrimlerin minimum uzunluğuna eşittir. Yüksek güvenilirliğe sahip haberleşme şebekesinin tasarımı için t(G) değerinin olabildiğince yüksek olması gerekir.

Şebekenin birleştirme (articulation) düzeyi: Bir grafın düğüm ya da hat birleştirme düzeyi bir grafı en az iki alt grafa ayırmak için graftan kaldırılması için gereken düğüm ya da hatların minimum sayısıdır. Birleştirme düzeyinin m değerine eşit

olması grafı en az iki alt grafa ayırmak için graftan kaldırılması gereken m adet düğüm ya da hattın minimum kümesidir. Yüksek güvenilirliğe sahip bir haberleşme şebekesinde düğüm ve hat birleştirme düzeyinin mümkün olduğu kadar düşük olması gerekmektedir.

Güvenilirlikte Olasılıklı Ölçüler

Literatürde, şebeke güvenilirliği ile ilgili 3 ana güvenilirlik ölçüsü vardır. Bunlar, bağlılık, esneklik ve başarabilirliktir (performability). Şebeke güvenilirliği üzerine yapılan çalışmaların büyük çoğunluğu bağlılık üzerine odaklanmıştır. Bunun nedeni bir haberleşme şebekesinin temel fonksiyonunun bağlılık ile sağlanmasıdır. [8].

Yönlendirilmemiş olasılıklı şebekeler için temel şebeke güvenilirlik ölçüleri şunlardır:

- *İki terminal güvenilirliği:* Seçilen düğüm çiftlerinin (kaynak düğümü *s* ve hedef düğümü *t*) bağlı olması olasılığıdır. (T = {*s*,*t*})
- *Tüm terminal güvenilirliği:* Her düğüm çiftinin diğer tüm düğümlerle haberleşebilme olasılığıdır. (T=V)
- *K terminal güvenilirliği:* V'nin bir alt kümesi olan K'da yer alan düğümlerin K'da yer alan diğer düğümlerle haberleşebilme olasılığıdır.

İki terminal ölçüsü bir şebekedeki belli iki düğüm arasındaki haberleşmenin kritik olduğu durumlarda (metropolitan alanlar, farklı sitelerdeki dosya ve sunucu) kullanılır. Tüm terminal güvenilirliği paket anahtarlı (packet switched) şebekelerin omurga seviyelerinde kullanılır. Eğer bu şebekede bir hat bozulursa trafik şebekedeki alternatif bir rotaya yönlendirilir. K terminal güvenilirliği ise şebeke düğümlerinin bir alt kümesinin bağlılığı söz konusu olduğunda kullanılır (sanal yerel ağlar ve dağıtık hesaplama uygulamaları). Yukarıda bahsedilen bu güvenilirlik ölçüleri birbirleriyle ilişkilidir. Biri için uygulanan bir yöntem diğerleri için de uygulanabilir. Genel şebekeler için bu güvenilirlik ölçülerinin hesaplanması NP-zor bir problemdir [8].

Şebeke esneklik (resiliance) ölçüleri bağlılık ölçülerinin özel bir türüdür. Yukarıda verilen bağlılık ölçüleri istenen bağlılık sağlanmadığında şebekenin çalışmayacağını varsayar. Gerçek durumda bir veya birkaç düğüm devre dışı kaldığında bile bağlı düğümler arasında haberleşme devam eder. Bir şebekenin en yıkıcı bozulmalarla baş etmesi ve bağlı olmayan kısımlardaki haberleşmenin düzeltilmesi yeteneklerini değerlendirmek için birkaç şebeke esneklik ölçüsü geliştirilmiştir. Şebekenin esnekliğinin hesaplanması güvenilirliğin hesaplanmasından daha zordur [8].

Cevap zamanı ve işlem hacmi gibi performans ölçüleri haberleşme şebekelerinin tasarım ve analizinde yaygın bir biçimde kullanılır. Şebeke başarabilirliği (performability) ölçüleri şebekenin farklı durumlardaki performansıyla ilgilenir. (Ω) bir şebekenin verilen performans ölçüsünü göstermek üzere başarabilirlik ölçüleri genellikle üç durumda ifade edilirler [8]:

- $\Pr(\Omega \le \alpha)$;
- $E|\Omega|$;
- $E|\Omega|$ (en fazla k bileşen arızalandığında).

Son grup başarabilirlik ölçüsü k (genellikle bir ya da iki) değeri kullanarak hesaplama karmaşıklığını azaltma amacı gütmektedir [8].

2.5.3. Güvenilirliğin değerlendirilmesi

Şebeke güvenilirliğinin değerlendirilmesinde 3 yöntemden bahsedilebilir:

1. Tam Değerin Hesaplanması
2. Sınırlar
3. Benzetim ve Diğer Tahmin Yöntemleri

Tam Değerin Hesaplanması

Şebeke güvenilirliğinin tam değerinin hesaplanmasında kullanılan metotlar ikiye ayrılır:

- Kesme/yol kümesi sayımı metotları,
- Durum sayma metotları.

Kesme/yol kümesi sayımı metotları, şebekenin tüm kesme/yol kümelerinin sayılmasını gerektirir. Örneğin, bir şebekenin verilen tüm yol kümelerinin kümesi $P_1,....,P_h$ olsun ve E_i, P_i'deki tüm hatların çalışıyor olması durumunu ifade ediyor olsun. Bu durumda şebeke güvenilirliği Eş. 2.1 kullanılarak hesaplanır.

$$R(G) = \Pr\left[E_1 \cup E_2 \cup ....E_h\right] \tag{2.1}$$

Fakat bu eşitlikte E_i'ler ayrık olaylar olmadığından ve bir şebekenin yol kümelerinin sayısı hatların sayısına bağlı olarak üstel biçimde arttığından hesaplama yapmak çok zordur.

En temel durum temelli metot, m adet hattan oluşan bir şebekede tüm 2^m adet durumun üretilmesini gerektirir. Bu tür durumlarda güvenilirlik korumalı şebeke indirgeme ve şebeke ayrıştırma teknikleri hesaplama zamanında önemli derecede azalmalar sağlayabilir. Şebeke indirgemede G şebekesi daha az düğüm ve hattan oluşan G' şebekesine indirgenir.

$$R(G) = \lambda R(G') \tag{2.2}$$

Bu eşitlikte λ güvenilirlik koruyan sabit çarpandır. Şebeke ayrıştırmada bir şebeke iki veya daha fazla alt şebekeye ayrılır ve tüm şebekenin güvenilirliği ayrıştırmayla oluşan şebekelerin güvenilirlikleri kullanılarak hesaplanır. Bu yöntem, iki terminal yönlendirilmiş şebekelerde başarılı şekilde uygulanmaktadır. Güçlü bir şebeke

ayrıştırma tekniği *çarpanlara ayırma (factoring)* olarak bilinen metottur. (i,j) arkının durumuna bağlı olarak R(G) güvenilirlik değeri şu şekilde ifade edilir:

$$R(G) = R(G \cdot (i, j))\, p_{ij} + R(G - (i, j))(1 - p_{ij}) \tag{2.3}$$

Bu ifadede $G \cdot (i, j)$, G şebekesinden i ve j düğümlerinin yeni bir düğüm şeklinde birleştirilmesiyle her hattı bu düğüme bağlayarak elde edilen şebekeyi; $G - (i, j)$, G şebekesinden (i,j) hattının silinmesiyle elde edilir. Eğer bu ayrıştırma yöntemi doğru hatlar seçilerek uygulanırsa şebekenin güvenilirliğinin elde edilmesinde önemli geliştirmeler sağlayabilir.

Gerçekte şebeke güvenilirliğinin hesaplanması çok büyük hesaplama zamanları gerektirir. Yönlendirilmemiş hatlar için şebeke indirgemesi kullanarak çarpanlara ayırma en iyi mümkün zamanı verir [8].

Sınırlar

Güvenilirliğin tam değerinin hesaplanmasının zorluğu nedeniyle güvenilirliğin teorik sınırları kullanılarak güvenilirlik hesaplanmaktadır. Şebeke güvenilirliğine yaklaşmada sınırların kullanılması 3 grupta incelenebilir:

- Güvenilirlik polinomuna bağlı sınırlar,
- Kesme veya yol kümesi yoluyla hat paketlemeye dayalı sınırlar,
- En olası durumlara bağlı sınırlar.

İlk grup sınırlar operasyonel şebeke durumlarının küçük bir parçasının sayılmasına dayanır. Bu yaklaşımın zayıf yanı yalnızca eşit hat olasılıklarının olduğu şebekelere uygulanabilmesidir.

İkinci grup sınırlar bir şebekenin tüm kesme/yol setlerinin yalnızca bir parçasının dikkate alırlar. Bu sınırlar farklı hat olasılıklarına sahip şebekelerde kullanılabilir.

Bununla birlikte, bu sınırlar yalnızca 0–1 yapı fonksiyonlu bağlılık temelli güvenilirlik ölçülerinde kullanılabilir.

En olası durum sınırları, tek tek bileşenlerin güvenilirliklerinin çok yüksek olması ve bu nedenle tüm durumların küçük bir parçasının olasılığın büyük bir bölümünü temsil edebileceği temeline dayanır. Mümkün şebeke durumlarının sayısının çok fazla olmasına rağmen birçoğu düşük oranda ortaya çıkma olasılığına sahip olduğundan göz ardı edilebilirler. En olası durumlar metodu k adet en olası durumun $\mathbf{X}^1$, $\mathbf{X}^2$,……,$\mathbf{X}^k$ olmak üzere $\Pr\{\mathbf{X}^1\} \geq \Pr\{\mathbf{X}^2\} \geq \ldots\ldots \geq \Pr\{\mathbf{X}^k\}$ şeklinde numaralandırılmasını gerektirir. k adet en olası duruma bağlı olarak güvenilirliğin üst ve alt sınırları şu şekilde elde edilir:

$$R_U(G) = \sum_{i=1}^{k} \Phi(X^i) \Pr\{X^i\} + (1 - \sum_{i=1}^{k} \Pr\{X^i\} \tag{2.4}$$

$$R_L(G) = \sum_{i=1}^{k} \Phi(X^i) \Pr\{X^i\} \tag{2.5}$$

Bu sınırların aralığı göz önüne alınan durumların sayısına bağlıdır [12].

Bu çalışmada, şebekelerin güvenilirlik üst sınırının belirlenmesinde Jan [13] tarafından önerilen yaklaşım kullanılmıştır. Bu yöntem, ℓ =n+2, n+3,….[n(n-1)]/2 hatta sahip ve düğüm dereceleri d_1, d_2,…..,d_n ifade edilen şebekenin güvenilirlik üst sınırını hesaplamaktadır. Düğüm derecesi, i düğümüyle ilişkili olan hatların sayısının toplamıdır. Buna göre her şebeke $\sum_{i=1}^{n} d_i = 2\ell$ ile ifade edilen benzersiz bir derece dizisine sahiptir. Bu ifadede ℓ, şebekedeki hatların toplam sayısını göstermektedir. G şebekesindeki her hat q = 1-p güvenilirliğine sahip olduğunda G şebekesinin güvenilirliği Eş. 2.5 kullanılarak hesaplanmıştır.

$$R(G) \leq 1 - \left[\sum_{i=1}^{n} q^{d_i} \prod_{k=1}^{m_i} (1 - q^{d_i - 1}) \prod_{k=m_i+1}^{i-1} (1 - q^{d_i}) \right] \tag{2.6}$$

$m_i = min(d_i, i-1),\ i = 1,2,....,n$ ifade etmektedir [9].

Benzetim ve Diğer Tahmin Teknikleri

Şebeke güvenilirliğinin tam değerinin hesaplanmasının zorluğu ve dar sınırların bulunmaması nedeniyle şebeke güvenirliliğinin tahmininde benzetim önemli bir seçenektir [8]. Bu tekniklerin en yaygın olarak kullanılanı *Monte Carlo Benzetimi*dir. Monte Carlo benzetimi, örnek uzayından rassal olarak örneklemler alarak güvenilirliğin gerçek değeri tahmin eder. Monte Carlo benzetiminin uygulanma aşamasında performans ölçülerinde ve hat arızalarında bazı varsayımlar söz konusudur. Bu durum, metodun hem güçlü hem de zayıf tarafıdır. Örnekleme planının geniş bir başarabilirlik performansı yelpazesine hizmet etmesi ve tam değer hesaplamaya göre daha fazla uygulanabilir bir yapıya sahip olması Monte Carlo benzetiminin güçlü yönü iken, ilgilenilen problemle ilgili yapısal bilgiyi kullanmaması zayıf yanıdır. Bu yöntemde çözüm için ayrılan zaman arttıkça daha güvenilir tahminler yapılabilmektedir. Monte Carlo benzetiminde amaç, makul bir sürede doğru tahmini gerçekleştirebilmektir [4].

Yukarıda da belirtildiği gibi bu metot şebeke durumlarının rassal olarak örneklenmesi ve örneklenen durumlar için çalışan durumların belirlenmesinden oluşmaktadır.

$1 \leq i \leq m$ olmak üzere n düğümlü bir şebekeden elde edilen m tane rassal S_i durumu için şebekenin çalışıp çalışmadığı belirlenir. S_i durumuna bağlı olarak $r(S_i)$ fonksiyonu için aşağıdaki durumlar geçerlidir.

$$r(S_i) = \begin{cases} 1, & S_i \text{ çalışıyorsa} \\ 0, & \text{d.d.} \end{cases} \qquad (2.7)$$

Bu durumda şebeke güvenilirliğinin tahmini değeri Rel (G) aşağıdaki gibi elde edilir.

$$\mathrm{Re}\,l(G) = \frac{1}{m}\sum_{i=1}^{m} r(S_i) \qquad (2.8)$$

Bu çalışmada şebeke güvenilirliği Yeh ve arkadaşları [10] tarafından geliştirilen Monte Carlo benzetimi ile tahmin edilmiştir. Bu yönteme ilişkin ayrıntılı açıklama aşağıda yer almaktadır.

$0 \le k \le l$ olmak üzere C_k, şebekedeki k tane hattın arızalı olduğu alt kümelerin sınıfını göstersin. Bu durumda şebekede $C_0, C_1,...,C_l$ için $l+1$ tane sınıf ve her sınıfın da $l!/[k!(l-k)!]$ tane alt kümesi vardır. Her tekrarlamada her bir C_k'dan $l+1$ tane örnek alınır. Bu durumda $j = 1,2,...,m$ olmak üzere j. tekrarlamada güvenilirliğin tahmini

$$\begin{aligned} \bar{r}_j &= \beta_0 P[N=0] + \beta_1 P[N=1] + \ldots + \beta_1 P[N=t] \\ &= \sum_{k=0}^{l} \beta_k P[N=k] \end{aligned} \tag{2.9}$$

eşitliği ile tahmin edilir. Bu eşitlikte Ni şebekedeki arızalı hat sayısını ve β_k, $N=k$ için şebekenin çalışıp çalışmadığını göstermektedir. Eğer şebeke bağlı ise $\beta_k = 1$, aksi halde $\beta_k = 0$ değerini alır. m tane tekrar sonucunda şebeke güvenilirliğinin tahmini

$$\mathrm{Re}\, l'(G) = \sum_{j=1}^{m} \bar{r}_j / m \tag{2.10}$$

eşitliği ile yapılır.

Yeh ve arkadaşları [14], β_k ve $P[N=k]$ değerlerini sırasıyla NFA ve BETA adını verdikleri iki örnekleme planıyla hesaplamaktadırlar. p, hatların çalışma olasılığını; Xi i hattının durumunu ($X_i = 1$, hat çalışıyorsa; $X_i = 0$ hat arızalı ise) göstermek üzere Monte Carlo metodunun işleyişi aşağıda verilmektedir [11].

NFA algoritmasının adımları aşağıdaki şekildedir:

Adım 1. $k = 0,1,2,\ldots..,1$ için $N_k = 0$
Adım 2. for $j = 1$ to m do
Adım 2.1. for $i=1$ to l do

Adım 2.1.1. U (0,1) dağılımından bir u_i rassal sayısı üret

Adım 2.1.2. Eğer $u_i < p$ ise $X_i = 1$, değilse $X_i = 0$

Adım 2.2. k=0,1,2,….., l için arızalı hat sayısı k ise N_k sayacını bir artır:N_k=N_k+1

Adım 3. k = 0,1,2,….., l için, $P[N=k] = N_k/m$ değerlerini hesapla.

BETA algoritmasının adımları aşağıdaki gibidir.

Adım 1. for k=0 to l do

Adım 1.1. Şebekeden rassal olarak k tane hat seç: C_k'dan bir A_k örneği al.

Adım 1.2. Şebekeden seçilen k tane hat çıkarıldığında şebeke bağlı ise $\beta_k = 1$, değilse $\beta_k = 0$.

Monte Carlo yönteminin adımları aşağıdaki gibidir.

Adım 1. k = 0,1,2,….., l olmak üzere arızalı hat sayısının dağılımı $P[N=k]$'yı simüle etmek için NFA algoritmasını kullan.

Adım 2. for j =1 to m do

Adım 2.1. Tüm k = 0,1,2,….., l için BETA algoritmasını kullanarak β_k değerini simüle et

Adım 2.2. $\bar{r}_j = \sum_{k=0}^{l} \beta_k P[N = k]$ değerini hesapla

Adım 3. $\mathrm{Re}\, l'(G) = \sum_{j=1}^{m} \bar{r}_j / m$ şebeke güvenilirliğini hesapla.

Göreli hassasiyet düzeyi ε ve güven aralığı %(1-2c) alındığında gerekli tekrar sayısı m aşağıdaki eşitlik kullanılarak hesaplanır.

$$m \geq \frac{\sum_{k=0}^{l} P[N = k]^2}{4\varepsilon\sqrt{Var[\mathrm{Re}\, l'(G)]}} \qquad (2.11)$$

Bu çalışmada, ε 0.005 ve güven aralığı %95 alınarak tüm şebeke boyutları için gerekli tekrar sayısının 3000 olmasına karar verilmiştir. Dengiz ve arkadaşları şebeke güvenilirliğinin tahmininde anlatılan metodu kullanmışlar ve elde edilen tahmini güvenilirlik düzeyi ile geri izleme (backtracking) algoritması kullanılarak elde edilen güvenilirlik düzeyi arasında %0.5'lik bir sapma olduğunu göstermişlerdir [12].

2.6. Güvenilirlik Kısıtı Altında Minimum Maliyetli Şebekenin Tasarımı Problemi ve Karmaşıklığı

Bir şebekenin güvenilirliği şebeke birleşenleriyle (hatlar, terminaller) şebekenin topolojik yapısına da bağlıdır. Minimum maliyetli ve bağlı bir şebeke elde etmenin en kolay yolu minimum yayılan ağaç topolojisine sahip bir şebeke tasarlamaktır. Böyle bir şebekenin maliyetinin düşük olmasına karşın güvenilirliği düşük olacaktır. Yüksek güvenilirliğe sahip bir şebeke elde etmenin en kısa yolu ise tüm düğümlerin birbirine direk bağlanmasıdır. Bu yaklaşım yüksek güvenilirliğe sahip bir şebeke elde edilmesini sağlayacaktır. Ancak, böyle bir şebekenin de maliyeti çok yüksek olacaktır. Güvenilirlik kısıtını sağlayan minimum maliyetli şebeke bu durumda yayılan ağaç ile tam bağlı topoloji arasında olacaktır. n adet düğüme ve $\ell = [n(n-1)]/2$ adet hatta sahip bir şebekede mümkün topolojilerin sayısının üst sınırı 2^{ℓ} tanedir. Düğüm sayısı arttıkça mümkün topolojilerin sayısı ve en iyi topolojiyi elde etmek için gereken çözüm zamanı üstel olarak artmaktadır.

Bu problemi zor kılan noktalardan biri de şebeke güvenilirliğinin hesaplanmasıdır. Önceki bölümlerde de değinildiği gibi şebeke güvenilirliğinde deterministik ve olasılıklı ölçüler vardır. Deterministik ölçüler bir şebekede haberleşmenin gerçekleşmemesi için arızalanması gereken hat ve düğümlerin sayısıyla ilgilenir. Bu çalışmada şebeke güvenilirliğine ilişkin deterministik ölçü olarak 2-bağlılık göz önüne alınmıştır. Buna göre, şebekedeki tüm düğümlerin derecelerinin en az 2 olması gerekmektedir. Fakat, deterministik ölçüler tek başlarına şebekenin güvenilirliği için yeterli değildir. Bu çalışmada güvenilirlik ölçüsü olarak olasılıklı ölçü kullanılmıştır. Olasılıklı yaklaşımda ise hat ve düğümlerin belirli olasılıklarla arıza yapmaları söz

konusudur. Fakat bu çalışmada düğümlerin tam güvenilir olduğu kabul edilmiştir ve olasılıklı güvenilirlik ölçülerinden tüm terminal güvenilirliği göz önüne alınmıştır. Yani tüm düğüm çiftlerinin diğer düğümlerle haberleşebilme olasılıkları göz önüne alınmıştır. Şebeke güvenilirliğinin hesaplanması aşamasında ise Yeh ve arkadaşları [10] tarafından önerilen Monte Carlo benzetimi kullanılmıştır. Tüm şebeke boyutları için benzetimin tekrar sayısı 3000 olarak alınmıştır. Fakat çözümün her iterasyonunda şebekelerinin hepsinin güvenilirliği tahmin edilmemiştir. Yalnızca güvenilirlik üst sınırı R_0 değerine eşit veya bu değerden büyük olan şebekelerin güvenilirlikleri tahmin edilmiştir. Bu aşamada, Jan [9] tarafından geliştirilmiş şebeke güvenilirliğinin üst sınırı hesaplaması kullanılmıştır.

Sonuç olarak, güvenilirlik kısıtı altında minimum maliyetli haberleşme şebekelerinin topolojik tasarımı problemi NP-zor bir problemdir. Bunun nedeni şebeke boyutu arttıkça arama uzayının üstel olarak artması ve şebeke güvenilirliğinin tahmini için gereken sürenin şebeke boyutu arttıkça üstel olarak artmasıdır. Bu problem için en iyi şebeke tasarımını klasik eniyileme yoluyla elde etmek küçük boyutlu problemlerde bile oldukça zor olmaktadır. Bu nedenle sezgisel optimizasyon tekniklerinin kullanılması kaçınılmaz olmaktadır. Sezgisel optimizasyon teknikleri klasik eniyileme tekniklerinin aksine global en iyi bulmayı garanti etmezler. Son yıllarda geliştirilen sezgisel optimizasyon teknikleri çözüm uzayında arama yaparken global en iyi ya da global en iyi değere yakın değerler üretebilmektedirler. Bu problemin çözümünde daha önce genetik algoritma, tabu arama, tavlama benzetimi, yapay sinir ağları sezgiselleri kullanılmış ve başarılı sonuçlar elde edilmiştir.

Bu çalışmada ise güvenilirlik kısıtı altında minimum maliyetli şebeke tasarımı probleminin çözümünde değişken komşu arama (variable neighborhood search), kuş sürüsü eniyilemesi (particle swarm optimization) ve karınca kolonisi eniyilemesi (ant colony optimization) sezgisel teknikleri kullanılmıştır.

3. LİTERATÜRÜN İNCELENMESİ

Çalışmanın bu bölümünde öncelikle şebekelerin topolojik tasarımı ile ilgili yapılmış olan çalışmalara yer verilecek, daha sonra bu çalışmada şebeke tasarımı probleminin çözümünde kullanılan sezgisel metotlarla yapılmış çalışmalara ilişkin bilgilere yer verilecektir.

3.1. Şebekelerin Topolojik Tasarımı İle İlgili Çalışmalar

Broostyn ve Frank, çalışmalarında şebeke tasarımı problemi türlerinden olan işlemci yerleşimi problemi, terminal atama problemi, terminal yerleşimi, dağıtık şebekelerin topolojik yerleşimi ve ana şebeke düğüm yerleşimi problemleri ile ilgilenmişleridir. Bu problemlerin çözümünde kullanılan kesin (exact) ve sezgisel yöntemler hakkında bilgiler verilmiş ve bu yöntemlerle elde edilen çözüm kaliteleri değerlendirilmiştir. Bilgi trafiği ihtiyaçlarının dikkate alındığı bir ana şebeke problemi için dal değiştirme ve kesme doyurma sezgisel yöntemlerinin açıklandığı çalışmada kesme doyurma yönteminin daha etkin olduğu sonucuna varılmıştır [13].

Aggarwal ve Bajwa, maliyet kısıtı altında maksimum s-t güvenilirliğini verecek bir sezgisel yöntem geliştirmişlerdir [14].

Chopra ve arkadaşları, haberleşme şebekelerinin güvenilirlik ve maliyet kısıtı altında topolojik optimizasyonunda kullanılan bir metot geliştirmişlerdir. Geliştirilen metodun avantajı, kısıtlarda yapılacak değişikliklerin modelde değişiklik gerektirmemesidir [15].

Aggarwal ve Suresh, şebeke güvenilirliğini değerlendirmede yayılan ağaç temelli bir yaklaşım önermişlerdir. Her hattın eşit arızalanma olasılığına sahip olduğu varsayımı altında şebeke güvenilirliği ve s-t terminal güvenilirliği ifadeleri karşılaştırılmıştır [16].

Venetsanopoulos ve Singh, haberleşme şebekelerinin güvenilirlik kısıtı altında topolojik optimizasyonu problemini ele almışlardır. Problemin çözümünde tüm terminal güvenilirliğinin dikkate alındığı bir sezgisel algoritma geliştirmişlerdir. Bu algoritma marjinal analiz ve dal sınır tekniğine dayanmaktadır. Araştırmacılar geliştirdikleri metodun çözüm hızını azalttığını göstermişlerdir [17].

Jan ve arkadaşları, güvenilirlik kısıtı altında minimum maliyetli şebekenin topolojik tasarımında dal sınır algoritmasına dayanan bir ayrıştırma metodu kullanmışlardır. Çözümü hızlandırmak için de düğüm dereceleri kullanılarak güvenilirlik üst sınırı hesaplaması yapılmıştır [9].

Atiqullah ve Rao, maliyet kısıtı altında tüm şebekenin güvenilirliğinin maksimizasyonu probleminde tavlama benzetimi kullanmışlardır. Algoritma, global en iyi değere ulaşmada hiyerarşik stratejiyi dahil eden tavlama benzetiminin bir varyasyonudur. Önerilen algoritmanın karmaşık şebekelerde kullanmanın oldukça avantajlı olduğu belirtilmiştir [18].

Kumar, Pathak ve Gupta, var olan bilgisayar şebekelerinde güvenilirlik kısıtını bozmadan yeni düğüm ve hatlar eklenerek yapılan şebeke genişletme (network expansion) problemi üzerinde durmuşlardır. Şebeke genişletmede maliyet kısıtı altında şebeke güvenilirliğinin maksimizasyonu probleminin çözümü için genetik algoritma temelli bir metot geliştirmişlerdir. Bu yaklaşımda amaç fonksiyonunun değerlendirilmesinde yapılacak küçük değişikliklerle problemin farklı türlerinde de kullanılabileceğini belirtmişleridir. Araştırmacılar genetik algoritma temelli geliştirdikleri metodu kullanarak elde ettikleri sonuçları birerleme (exhaustive search) metodu ile elde ettikleri en iyi çözümleri karşılaştırmışlardır. Kullandıkları yöntem doğruluk ve hesaplama zamanı açısından etkin olmasına karşın her zaman en iyi çözümü garanti etmemektedir [19].

Kumar ve arkadaşları, bu çalışmalarında dağıtık sistemlerin topolojik optimizasyonu için genetik algoritma kullanmışlardır. Bu şebekelerin optimizasyonunda çap, uzaklık ve şebeke güvenilirliği kısıtları göz önüne alınmıştır. Araştırmacılar

önerdikleri yaklaşımın şebeke optimizasyonunun birçok türünde kolaylıkla uygulanabileceğini de belirtmişlerdir. Genetik algoritmayla elde ettikleri sonuçların birerleme yöntemi ile elde edilen çözümler kadar kaliteli olduğunu göstermişlerdir [20].

Deeter ve Smith, tüm terminal güvenilirliği kısıtı altında haberleşme şebekelerinin topolojik eniyilemesinde genetik algoritma kullanmışlardır. Araştırmacılar problemi şebekeye eklenecek hatların farklı güvenilirlik düzeyine sahip hatlar arasından seçilmesi şeklinde genişletmişlerdir. Kullandıkları algoritma esnek ve etkin çözüm sağlamıştır [21].

Dengiz, Altıparmak ve Smith, güvenilirlik kısıtlı minimum maliyetli haberleşme şebekelerinin topolojik optimizasyonunda genetik algoritma kullanmışlardır. Bu çalışmada kromozomların kodlamasında değişken boyutlu tamsayı değerleri kullanmışlar ve kullanılan kodlama yapısına uygun çaprazlama ve mutasyon operatörü geliştirmişlerdir. Genetik algoritmanın başlangıç çözümünde yüksek güvenilirlikli çözümlerle başlamışlar ve geliştirilen genetik operaörler ile iyi çözümler elde etmişlerdir. Ayrıca uygunluk fonksiyonunun değerlendirilmesinde ceza fonksiyonu kullanmışlardır [12].

Ahuja, bilgisayar şebekelerinin performans tabanlı güvenilirlik optimizasyonunda genetik algoritma kullanmıştır. Bu çalışmada hatlara kapasite tahsisleri yapılarak şebeke güvenilirliğinin optimizasyonu gerçekleştirilmiştir [22].

Costamagna, Fanni ve Giacinto, çok katlı haberleşme şebekelerinde tasarımında tavlama benzetimi yöntemi kullanmışlardır. Elde ettikleri sonuçları genetik algoritma ve sezgisel bir yöntemle elde ettikleri sonuçlarla karşılaştırmışlar ve tavlama benzetimiyle daha kaliteli sonuçlar elde ettiklerini göstermişlerdir [23].

Pierre ve Legault, gecikme ve güvenilirlik kısıtı altında dağıtık bilgisayar şebekelerinin topolojik tasarımında genetik algoritma kullanmışlardır. Elde edilen

sonuçlar orta büyüklükteki problemlerde bilinen metotlara göre daha kaliteli sonuçlar üretmiştir [24].

Dengiz, Altıparmak ve Smith, genetik algoritma kullanarak güvenilirlik kısıtı alında minimum maliyetli şebekenin topolojik optimizasyonu üzerine çalışmışlardır. Şebeke gösteriminde 0 – 1 kodlama kullanmışlardır. Bu çalışmada genetik algoritmanın uygunluk fonksiyonunu hesaplamada ceza fonksiyonu kullanılmıştır. Bunun yanında şebekenin güvenilirliğini tahmin etmek için Monte Carlo benzetiminden yararlanılmıştır [25].

Pierre ve Elgibaoui, güvenilirlik kısıtı altında minimum maliyetli şebeke topolojisinin elde edilmesinde tabu arama metodunu kullanmışlardır. Her iterasyonda birden fazla komşu üretilerek içlerinden güvenilirlik kısıtını sağlayan minimum maliyetli şebeke seçilerek bir sonraki iterasyona geçilmiştir. Bu yaklaşımı da genelleştirilmiş yerel arama metodu denemesi olarak ifade etmişlerdir. Önerilen algoritma kesme doyurma, tavlama benzetimi ve genetik algoritma yöntemlerine göre daha iyi sonuçlar elde etmiştir [26].

Costamagna, Fanni ve Giacinto, geniş bant haberleşme şebekelerinde çok katlı merkezlerin sayısını belirleme ve yerleşimi probleminde tabu arama algoritması kullanılmıştır. Şebeke mimarisi, fiber optik kabloların yüksek maliyeti nedeniyle yayılan ağaçtır. Düğümlerdeki çok katlı merkezlerin aktif olup olmadıklarını belirtmekte 0–1 kodlama kullanılmıştır. Kullanılan tabu arama algoritmasında dinamik tabu listesi, frekans temelli uzun dönemli hafıza ve durdurma koşulu kullanılmıştır. Elde edilen sonuçların ekle/çıkar, genetik algoritma ve tavlama benzetimi algoritmalarıyla elde edilen çözümlere göre çözüm kalitesi ve çözüm süresi bakımından daha iyi oldukları görülmüştür [27].

Cheng, omurga (backbone) bilgisayar şebekelerinin topolojik tasarımında minimum toplam hat maliyeti ve 1 hattın bozulmasında hata toleransı (fault tolerant to 1 link-failure 1 FT) kısıtlarını dikkate almıştır. Cheng bu problemin çözümünde genetik algoritma kullanmıştır [28].

Deeter ve Smith, maliyet ve güvenilirlik kriterlerini göz önüne alarak haberleşme şebekelerinin optimal tasarımında genetik algoritma kullanmışlardır [29].

Altıparmak, Dengiz ve Smith, maliyet kısıtı altında şebeke güvenilirliğinin maksimizasyonu probleminde genetik algoritma kullanmışlardır. Hatların ve düğümlerin genetik algoritmada kodlarken tamsayılı gösterimden yararlanmışlardır [30].

Liu ve Iwamura, çalışmalarında çoklu güvenilirlik amacına sahip haberleşme şebekelerinin topolojik optimizasyonu probleminin çözümünde yeni yöntemler önermektedir. Bu çalışma bağımlı-şans (dependent-chance) çok amaçlı programlama, bağımlı-şans (dependent-chance) amaç programlama yöntemlerini ve bir stokastik benzetim tabanlı genetik algoritma yaklaşımı önermektedir [31].

Dengiz ve Alabaş, modifiye edilmiş seçme mekanizması kullanan tavlama benzetimi algoritmasıyla güvenilirlik kısıtı altında minimum maliyetli haberleşme şebekelerinin topolojik tasarımı problemi üzerine çalışmışlardır. Elde ettikleri sonuçları genetik algoritmadan elde edilen sonuçlarla karşılaştırmışlar ve tavlama benzetimi algoritmasıyla daha etkin sonuçların elde edildiğini belirtmişleridir [32].

AboElFotoh ve Al-Sumait, tüm terminal güvenilirliği kısıtı altında minimum maliyet amaç fonksiyonuna sahip şebekenin topolojik optimizasyonu probleminde yapay sinir ağları temeline dayanan bir yaklaşım kullanmışlardır. Araştırmacılar çalışmalarında ksııtlı komninatoryal problemler için kullanılan ve ilk olarak 1985'te ortaya atılan OPTI-net algoritmasını kullanmışlardır. Bu çalışmada güvenilirliğin karmaşık yapısından dolayı alt ve üst sınırlar kullanılmıştır. Algoritma 50 düğüme kadar oldukça hızlı ve kaliteli çözümler elde etmiştir [33].

Fard ve Lee, bir şebekeye hat eklenmesinde eklenecek hattın hangi düğüm çiftleri arasında ekleneceğini belirlenmede kullanılacak olan bir yöntem sunmuşlardır. Düğüm çiftleri yeni hatların eklenmesiyle oluşan yayılan ağaçların sayısından yararlanılarak karşılaştırılmakta ve en fazla yayılan ağacı sağlayan düğüm çifti

seçilmektedir. Yayılan ağaçların sayılmasında da şebekenin derece matrisinden yararlanılmaktadır [34].

Koide, Shinmori ve Ishii, güvenilirlik kısıtı altında şebeke optimizasyonu probleminde Jan ve arkadaşları tarafından güvenilirlik üst sınırının hesaplanmasında için önerilen algoritmada geliştirmeler yapmışlardır. Jan ve arkadaşları hat olasılıklarının aynı olduğu problemler için bir algoritma geliştirmişlerdir. Fakat Koide ve arkadaşları farklı hat olasılıkları durumu için Jan ve arkadaşlarının algoritmasında geliştirme yapmışlardır [35].

Altıparmak, Dengiz ve Smith, maliyet kısıtı altında şebeke güvenilirliğinin maksimizasyonu probleminin çözümünde önce tepe tırmanma, tavlama benzetimi ve genetik algoritma ile bu algoritmaların karma versiyonlarını kullanarak genel amaçlı sezgisellerin problem üzerindeki performanslarını karşılaştırmışlardır. Sonuç olarak genetik algoritmanın karma versiyonu olan memetik algoritma (MA) en iyi performansı göstermiştir [36].

Kumar ve arkadaşları, yaptıkları çalışmada haberleşme şebekelerinin tasarımının karmaşık, çok kısıtlı ve çok kriterli bir optimizasyon problemi olduğunu belirtmişlerdir. Araştırmacılar çalışmalarında Pareto Yakınsayan Genetik Algoritma kullanmışlardır ve elde ettikleri sonuçları dal değiştirme sezgiseliyle elde edilen sonuçlarla karşılaştırmışlardır [37].

Srivaree-ratana, Konak ve Smith, tüm terminal güvenilirliğini hesaplamada yapay sinir ağlarını kullanmışlardır. Güvenilirlik tahmininde kullandıkları bu yöntemi Monte Carlo simülasyonu, geri izleme algoritması ve alt ve üst sınır yöntemleriyle karşılaştırmışlardır. Bu karşılaştırmayı topoloji tasarımında kullandıkları tavlama benzetimi algoritması içinde gerçekleştirmişler ve yapay sinir ağları kullanılarak yapılan güvenilirlik tahminin daha hızlı sonuçlar verdiğini görmüşlerdir [38].

Mandal, Saha, Mukherjee ve Roy, haberleşme şebekelerinin omurga topoloji tasarımında birerleme tekniğinin bir varyasyonu olan ve koşulu sağlandığında global

ekstermumu garanti eden RAS algoritması kullanmışlardır. Elde ettikleri sonuçları genetik algoritmayla elde edilen sonuçlarla karşılaştırmışlar ve daha iyi sonuçlar elde etmişlerdir [39].

Altıparmak, Dengiz ve Smith, yapay sinir ağlarını kullanarak tüm terminal şebeke güvenilirliğini tahmin etmişlerdir. Geliştirdikleri algoritmayı homojen ve heterojen hat güvenilirlikleri için test etmişleridir. Bu çalışmada yapay sinir ağlarının eğitilmesinde rassal tasarım ve deneysel tasarımın etkileri de incelenmiş; deney tasarımıyla elde edilen verilerin daha doğru tahminler ürettiği gözlenmiştir [40].

Altıparmak, Gen, Dengiz ve Smith, güvenilirlik kısıtı altında bilgisayar şebekelerinin topolojik optimizasyonu için bulanık mantık kontrollü bir şebeke tabanlı genetik algoritma (flc-NBGA) geliştirmişlerdir. Geliştirilen flc-NBGA algoritmasında Prufer sayıları kullanılarak kodlama yapılmış, iki noktalı çaprazlama kullanılmış ve yerel arama operatörü kullanılarak mutasyon gerçekleştirilmiştir. Elde edilen sonuçlar 0-1 gösterim ve bulanık mantık kontrolsüz şebeke tabanlı genetik algoritma ile elde edilen sonuçlarla karşılaştırılmıştır [41].

Shao ve Shen, dağıtık erişim şebekelerinin toplam maliyet kısıtı altında güvenilirlik optimizasyonu üzerine çalışmışlardır. Yazarlar öncelikle maliyet kısıtlı güvenilirlik problemini kombinatoryal bir ağaçta bir arama prosesi olarak formüle etmişler böylece tüm mümkün çözümleri birerleme yoluyla elde etmişlerdir. Fakat elde edilen çözümler için güvenilirlik değerlerinin hesaplanması zaman alıcı olacağından arama prosesini hızlandırmak amacıyla *daraltma ve arama algoritması* (shrinking and searching algorithm – SSA) kullanılmıştır [42].

Xiong ve Gong, çalışmalarında şebeke güvenilirliği tahmini için oransal yaklaşım algoritması kullanmışlardır. Bu algoritmanın hem tüm terminal güvenilirliği problemlerine hem de çeşitli koruma algoritmalarına sahip şebekelerin güvenilirliği problemlerine uygulanabileceğini belirtmişlerdir [43].

Konak ve Bartolacci, şebeke tasarımı problemlerinin geleneksel kısıtı olan güvenilirlik yerine şebeke esnekliğini ele almışlardır. Yazarlar şebeke esnekliğinin bir göstergesi olan trafik esnekliğinin tahmininde kullanılmak üzere bir yöntem geliştirmişler ve şebekenin tasarımında da karma bir genetik algoritma kullanmışlardır. Karma genetik algoritma aşamasında özelleştirilmiş yerel arama operatörleri ve basit adaptif ceza fonksiyonları kullanmışlardır [44].

Cancela ve Petingi, geleneksel güvenilirlik hesaplamalarında yalnızca düğümler arasındaki arkların varlığının dikkate alındığı, fakat düğümler arası uzaklığında (çap) önemli bir kriter olduğunu belirtmişlerdir. Bu nedenle araştırmacılar çalışmalarında *çap kısıtlı güvenilirlik* yaklaşımını önermişlerdir. Ayrıca bu çalışmada çap kısıtıyla birlikte baskı (domination) durumunun etkisi de gözlenmiştir [45].

Khan ve Engelbrecht, dağıtık yerel alan şebekelerinin topolojik tasarımında birbiriyle çelişen maliyet, güvenilirlik, gecikme ve kaynak ile hedef arasındaki çevresel birim sayısı amaçlarını bulanık operatör kullanarak tek bir amaç fonksiyonunda birleştirmişlerdir. Araştırmacılar çalışmalarında UAO (birleştirilmiş VE-VEYA - unified AND-OR) bulanık operatörünü kullanmışlar ve evrimsel bir algoritmada kullandıkları geleneksel OWA(sıralı ağırlıklı ortalama – ordered weighted averaging) bulanık operatörünün performansı ile karşılaştırmışlar ve UAO operatörüyle elde edilen sonuçların daha iyi olduğunu belirtmişlerdir [46].

Reichelt, Gmilkowsky ve Linser, güvenilirlik kısıtı altında haberleşme şebekelerinin topolojik tasarımında yerel arama (local search – LS) ve ötelenmiş yerel arama (iterated local search – ILS) algoritmasını kullanmışlardır. LS algoritmasının genellikle yerel en iyi değere takılması nedeniyle bu algoritmaya bu algoritmaya ILS metotunda bir mutasyon operatörü eklenerek çözümler iyileştirilmiştir. Sonuçlar genetik algoritmayla elde edilen sonuçlarla karşılaştırmış ve ötelenmiş yerel arama algoritmasıyla elde edilen sonuçların daha iyi olduğu görülmüştür [47].

Lucio, Reed ve Hanning, haberleşme şebekelerinin optimizasyonunda geliştirilmiş tepe tırmanma algoritması olarak hızlı yerel arama (fast local search – FLS) ve yerel

minimumdan kurtulmak için yönlendirilmiş yerel arama (guided local search – GLS) algoritmalarını birleştirerek FLS+GLS tekniğini kullanmışlardır. Bu tekniğin genetik algoritma ve tavlama benzetimine göre oldukça basit olduğunu ve bu tekniklerdeki manuel parametre değişimine karşılık kendi kendine düzenleme avantajının bulunduğunu belirtmişlerdir. Çalışmanın sonucunda FLS+GLS tekniğinini genetik algoritma ve tavlama benzetimine göre daha hızlı ve daha kaliteli sonuçlar elde etmişlerdir [48].

Hui, şebeke topolojik optimizasyon problemlerinde geleneksel güvenilirlik tahmini ve sıralama (ranking) yönteminin çok zaman alıcı olması nedeniyle bazı hat yerleştirme problemlerinde şebekenin güvenilirlik sıralamasını tahmin edecek bir yaklaşım geliştirmiştir. Önerilen Synchronous Construction Ranking yönteminin geleneksel metotlara göre 30000 kat daha hızlı olduğu görülmüştür [49].

Marseguerra ve arkadaşları, haberleşme şebekelerinin topolojik tasarımında çok amaçlı optimizasyondan yararlanmışlar ve sistem güvenilirliğinin tahmininde Monte Carlo benzetiminden yararlanmışlardır. Araştırmacılar çalışmalarında maliyeti minimize ederken şebeke için istenen güvenilirlik düzeyinin sağlanmasına çalışmışlardır. Fakat sistem güvenilirliğini tahmin ederken düğüm ve arkların belirsizlikten etkilenmelerini de göz önüne almışlardır. Bu yaklaşımla oldukça farklı tasarımlarda şebekeler elde edebilmişlerdir [50].

3.2. Şebekelerin Topolojik Tasarımında Değişken Komşu Arama, Kuş Sürüsü Eniyilemesi ve Karınca Kolonisi Optimizasyonu'nu Kullanan Çalışmalar

Literatür incelendiğinde şebekelerin topolojik tasarımında Değişken Komşu Arama Kuş Sürüsü Eniyilemesi sezgisellerini kullanan bir çalışmaya rastlanmamıştır.

Qoubutra ve arkadaşları, ekonomi ve güvenilirlik kısıtlarını birlikte göz önüne alan haberleşme şebekelerinin topolojik optimizasyonu probleminin çözümünde Geliştirilmiş Karınca Kolonisi Optimizasyonu (Improved Ant Colony Optimization – IACO) kullanmışlardır. IACO yöntemi, paralel aramaya olanak sağlayabilen bir

yöntemdir. Bu yöntemin geleneksel yerel arama ve tabu arama yöntemlerine göre daha iyi sonuç verdiği belirtilmiştir [51].

4. DEĞİŞKEN KOMŞU ARAMA

Değişken Komşu Arama (Variable Neighborhood Search – DKA), aramada kullanılan komşuluk yapılarının sistematik biçimde değiştirilmesi esasına dayanan bir algoritmadır [52]. Bu algoritma 1997 yılında Pierre Hansen ve Nenad Mladenovic tarafından geliştirilmiştir.

Değişken Komşu Arama'da (DKA) N_k, ($k = 1,2,...,k_{max}$) sınırlı sayıdaki komşuluk yapılarının kümesini ve $N_k(s)$ s çözümünün k. komşuluk yapısıyla elde edilen çözümlerinin kümesini ifade eder. Genellikle, her s çözümü için bir komşuluk yapısından iç içe geçmiş komşuluklar serisi elde edilir. Bu k. komşuluğa doğru yapılacak hareket başlangıç komşuluğuna k defa yapılacak hareketle gerçekleşir. $s' \in S$ (çözüm uzayı) çözümü N_k'ya göre bir yerel optimumdur. Bu durum s' değerinden daha iyi bir çözüm olmadığında geçerlidir [52].

Değişken Komşu Arama (DKA) 3 basit gerçeğe dayanır:

1. Bir komşuluk yapısına göre elde edilen yerel minimum diğer komşuluk yapılarına göre bir yerel en küçük olmak zorunda değildir.
2. Bir global en küçük değeri tüm mümkün komşuluk yapılarına göre bir yerel en küçük değeridir.
3. Bir veya birkaç komşuluk yapısına göre yerel en küçük genellikle birbirine benzerdir.

Bu algoritmayı uygularken, farklı komşuluk yapılarını elde etmek için aşağıdaki yaklaşımlardan birisi kullanır:

a) Bir problem için farklı komşuluk yapılarının kullanılması,
b) Varolan yerel arama algoritmalarının parametrelerinin değiştirilmesi,
c) Birçok komşuluk yapısı uzaklık ölçüsüyle ilgilidir; bu durumda uzaklığın artırılması.

Değişken Komşu Arama'da 4 versiyondan bahsetmek mümkündür. Bunlar;

1. Değişken Komşu İniş (Variable Neighborhood Descent – DKİ)
2. Temel Değişken Komşu Arama (Basic Variable Neighborhood Search – TDKA)
3. İndirgenmiş Değişken Komşu Arama (Reduced Variable Neighborhood Search – İDKA)
4. Genel Değişken Komşu Arama (General Variable Neighborhood Search – GDKA)

DKA'nın temel prensiplerinin geliştirici yerel aramaya uygulanmasıyla Değişken Komşu İniş (DKİ) elde edilir. Bu metot, bir yerel aramada komşuluk yapılarının sistematik olarak değiştirilmesine dayanır. DKİ'ye ait adımlar aşağıda verilmektedir [53]:

N_k komşuluk yapıları kümesini seç ($k=1,2,..k_{max}$)
s başlangıç çözümü bul
Bir durdurma koşulu seç
Durdurma koşulu sağlanıncaya kadar aşağıdaki adımları tekrarla
Adım 1. $k=1$ olarak ayarla
Adım 2. $k=k_{max}$ oluncaya kadar aşağıdaki adımları tekrarla
s'den k uzaklığında bir komşu elde et.
$f(s') < f(s)$ ise $s=s'$ olarak kaydet
$k=1$ olarak devam et
d.d. $k=k+1$ olarak ayarla

Elde edilen son çözüm bütün k_{max} komşuluk yapılarına göre yerel en küçük olmalıdır ve böylece global en küçüğe ulaşma olasılığı tek bir komşuluk yapısı kullanma durumuna göre daha yüksek olmaktadır.

Birçok yerel arama sezgiseli iniş (descent) prosedürlerinde tek veya en fazla iki komşuluk yapısı kullanmaktadır. ($k_{max} \leq 2$) Düzensizlik (perturbation) stratejisi, birinci komşuluk yapısını kullanarak çözüm elde etmek ve yeni bir yerel aramaya

başlamak için ikinci komşuluk yapısından rassal olarak bir çözüm seçerek mevcut çözümü düzensizleştirmeyi içermektedir. Temel Değişken Komşu Arama (TDKA) düzensizleştirme için komşuluk yapılarında deterministik değişimler kullanır. TDKA'ya ilişkin adımlar aşağıda verilmektedir.

N_k komşuluk yapıları kümesini seç ($k=1,2,..k_{max}$)
s başlangıç çözümü bul
Bir durdurma koşulu seç
Durdurma koşulu sağlanıncaya kadar aşağıdaki adımları tekrarla
Adım 1. k=1 olarak ayarla
Adım 2. $k=k_{max}$ oluncaya kadar aşağıdaki adımları tekrarla
(a) Karıştırma
s'nin *k*. komşuluğundan rassal olarak bir *s'* noktası üret
(b) Yerel Arama
s' başlangıç çözümüne yerel arama metodunu uygula ve elde edilen yerel minimumu *s''* olarak kaydet
(c) Hareket et ya da dur
Eğer yerel en iyi daha iyi bir sonuç ise $s = s''$ olarak ayarla ve $k=1$ olarak devam et; eğer iyileşme olmuyorsa $k=k+1$ olarak ayarla

Durdurma koşulu olarak çözüm zamanı, iterasyon sayısı veya her iki iyileşme arasındaki geçen iterasyon sayısı olabilir.

İDKA ise $N_k(s)$'ten iniş kullanmadan rassal olarak noktalar seçildiğinde gerçekleşir. Bu algoritmanın yerel aramanın çok maliyetli olduğu problemlerde kullanılması daha doğrudur.

Temel Komşu Arama'nın 2 (b) kısmında yer alan yerel arama Değişken Komşu İniş ile yer değiştiğinde Genel Değişken Komşu Arama (GDKA) elde edilir. Bu algoritmaya ilişkin adımlar aşağıda verilmektedir.

N_k komşuluk yapıları kümesini seç ($k=1,2,..k_{max}$)

İniş için N_j komşuluk yapıları kümesini seç ($j = 1,2,...,j_{max}$)

s başlangıç çözümü bul

Bir durdurma koşulu seç

Durudurma koşulu sağlanıncaya kadar aşağıdaki adımları tekrarla

Adım 1. k=1 olarak ayarla

Adım 2. $k=k_{max}$ oluncaya kadar aşağıdaki adımları tekrarla

(a) Karıştırma

s'nin *k.* komşuluğundan rassal olarak bir s' noktası üret

(b) İniş

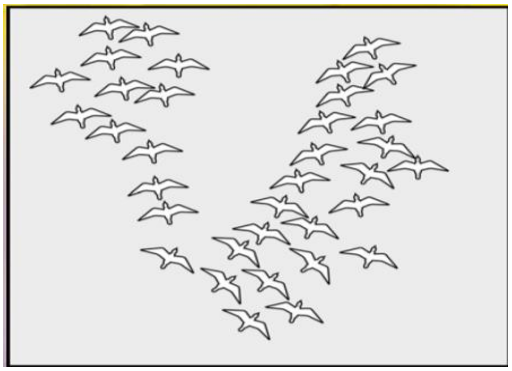
s' başlangıç çözümüne N_j ($j = 1,2,...,j_{max}$) kümesini kullanarak Değişken Komşuluk İniş (DKİ)'i uygula ve *s''* değerini elde et.

(c) Hareket et ya da dur

Eğer yerel en iyi daha iyi bir sonuç ise $s = s''$ ve k=1 olarak devam et; değilse $k=k$+1

5. KUŞ SÜRÜSÜ ENİYİLEMESİ

Kuş Sürüsü Eniyilemesi (Particle Swarm Optimization – KSE) algoritması, her parçacığın potansiyel bir çözümü temsil ettiği kuşların bir araya gelmesinden oluşur. Evrimsel algoritmalarla bir bağ kurulduğunda *sürü* populasyona, *kuş* ise bireye benzetilebilir. Basit bir ifadeyle kuşların çok boyutlu bir arama uzayında kendilerinin ve komşularının deneyimlerinden yararlanarak hareket ettirildiği bir algoritmadır. Kennedy ve Eberhart tarafından 1995 yılında geliştirilmiştir.



Şekil 5.1. Bir kuş sürüsünün hareketi

KSE'nin daha iyi anlaşılabilmesi için şöyle bir senaryo düşünülebilir: Bir grup kuş bir alanda rasgele bir biçimde yiyecek arıyor olsun. Arama yapılan alanda sadece bir parça yiyecek bulunmaktadır. Bütün kuşlar, yiyeceğin nerede olduğunu bilmemektedirler, fakat yiyeceğe her iterasyonda ne kadar uzakta olduklarını bilmektedirler. Bu durumda yiyeceğe ulaşmak için en uygun strateji yiyeceğe en yakın kuşu izlemektir. Bu senaryoda sürüde yer alan her bir kuş bir çözüm olarak düşünülür ve *kuş* adını alır.

i. kuşun *t* anında arama uzayındaki pozisyonu $x_i(t)$ olsun. Tersine belirtilmedikçe *t* kesikli zaman adımlarını ifade eder. Kuşun pozisyonu mevcut konumuna $v_i(t)$ hızının eklenmesiyle değiştirilir.

$$x_i(t+1) = x_i(t) + v_i(t+1),\ x_i(0) \sim U(x_{\min}, x_{\max}). \tag{5.1}$$

KSE'de eniyileme işlemini hız vektörü yönlendirir ve kuşun deneyimsel bilgisi ile kuşun komşularından sosyal değişimle elde ettiği bilgiyi yansıtır. Kuşun deneyimsel bilgisi, algoritmanın birinci adımından beri kuşun en iyi pozisyonundan (kişisel en iyi – personal best) olan uzaklığının şimdiki pozisyonundan olan uzaklığına oranı olan *bilişsel bileşeni (cognitive component)* ifade eder. Sosyal olarak elde edilen bilgi ise hız eşitliğinin *sosyal bileşeni (social component)* olarak ifade edilir.

Bu noktadan yola çıkarak kuşların komşularının büyüklüğüne bağlı olarak iki türlü KSE algoritması geliştirilmiştir: *Global En İyi KSE* ve *Yerel En İyi KSE*.

5.1. Global En İyi KSE

Global En İyi KSE'de her bir kuşun komşusu tüm sürüdür. Global En İyi KSE tarafından kullanılan şebeke yıldız topolojiye sahiptir. Yıldız komşu topolojisinde kuşun hız güncellemesi sürüdeki tüm kuşlardan elde edilen bilgiyi yansıtır. Bu durumda, sosyal bilgi sürü tarafından elde edilen en iyi pozisyondur ve $\hat{y}(t)$ ile gösterilir. Global En İyi KSE için *i* kuşunun hızı aşağıdaki eşitliğe göre hesaplanır.

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}[\hat{y}_j(t) - x_{ij}(t)] \qquad (5.2)$$

Bu eşitlikte $v_{ij}(t)$, $j= 1,...,n$ boyutundaki *i* kuşunun *t* zamanındaki hızı, $x_{ij}(t)$, *j* boyutundaki *i* kuşunun *t* zamanındaki konumu, c_1 ve c_2 bilişsel ve sosyal bileşenlerin katkılarını ölçekleyen pozitif hızlanma sabitleri, r_{1j} ve r_{2j} ise düzgün dağılımdan örneklenen ve [0,1] aralığında rassal sayılardır. Bu rassal değerler algoritmaya stokastiklik katmaktadır.

Kişisel en iyi pozisyon y_i, *i* kuşunun algoritmanın birinci adımından bu yana ziyaret ettiği en iyi pozisyondur. En küçükleme problemi göz önüne alındığında $t+1$ zamanındaki en iyi pozisyon şu şekilde hesaplanır:

$$y_i(t+1) = \begin{cases} y_i(t) & if \ \ f(x_i(t+1) \geq f(y_i(t)) \\ x_i(t+1) & if \ f(x_i(t+1) < f(y_i(t)) \end{cases} \tag{5.3}$$

Bu eşitlikteki *f,* uygunluk fonksiyonudur. Evrimsel algoritmalarda olduğu gibi uygunluk fonksiyonu mevcut çözümün en iyi çözüme ne kadar yakın olduğunu değerlendirmeye yarar [54].

n_s sürüdeki toplam kuşların sayısını ifade eder. Global en iyi KSE'ye ait adımlar aşağıda verilmektedir:

n_x boyutlu bir sürü *S* oluştur ve başla;
repeat
for her $i=1,...,n_s$ kuşu için *do*
// kişisel en iyi pozisyonu ayarla
if $f(Sx_i) < f(Sy_i)$ then $Sy_i=Sx_i$;
end
if $f(Sy_i)<f(S\hat{y})$ then $S\hat{y}=Sy_i$ olarak ayarla;
end
end
for $i=1,...,n_s$ paçacığı için *do*
Hızı güncelle
Pozisyonları güncelle
end
until durdurma koşulu

5.2. Yerel En İyi KSE

Yerel en iyi KSE, daha küçük komşuların tanımlandığı halka sosyal şebeke topolojisi kullanır. Sosyal bileşen, kuşun komşuları içerisinde değiştirilen çevrenin yerel bilgisini yansıtır. Hız eşitliğinde kuşun hızına sosyal katkı bir kuşla kuşların

komşularının bulduğu en iyi pozisyon arasındaki uzaklığın oranıdır. Hız aşağıdaki şekilde hesaplanır:

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}[\hat{y}_{ij}(t) - x_{ij}(t)] \quad (5.4)$$

$\hat{y}_{ij}$, j boyutundaki i kuşunun komşusu tarafından elde edilmiş en iyi pozisyondur. Yerel en iyi kuş pozisyonu $\hat{y}_i$, örneğin N_i komşusu tarafından bulunan en iyi pozisyon

$$\hat{y}_i(t+1) \in \{N_i \mid f(\hat{y}_i(t+1)) = \min\{f(x)\}, \ \forall x \in N_i\} \quad (5.5)$$

şeklinde ifade edilir. n_{Ni} büyüklüğündeki komşular için N_i

$$N_i = \{y_{i-n_{N_i}}(t), y_{i-n_{N_i}+1}(t), \ldots, y_{i-1}(t), y_i(t), y_{i+1}(t), \ldots, y_{i+n_{N_i}-1}(t), y_{i+n_{N_i}}(t) \quad (5.6)$$

şeklinde ifade edilir. Yerel en iyi pozisyon ayrıca komşu en iyi pozisyon olarak da kullanılır [54].

Yerel en iyi KSE'ye ait algoritma adımları aşağıda verilmektedir:

n_x boyutlu bir sürü S oluştur ve başla;
repeat
for her $i=1,\ldots,n_s s$ paçacığı için *do*
// kişisel en iyi pozisyonu ayarla
if $f(Sx_i) < f(Sy_i)$ then $Sy_i = Sx_i$;
end
// *if* $f(Sy_i) < f(S\hat{y})$ then $S\hat{y} = Sy_i$ olarak ayarla;
end
end
for $i=1,\ldots,n_s$ paçacığı için *do*

Hızı güncelle
Pozisyonları güncelle
end
until durdurma koşulu

5.3. Kuş Sürüsü Eniyilemsi'nde Temel Durumlar

Bu bölümde, hız eşitliğinde yer alan bileşenler daha ayrıntılı biçimde incelenecektir. Bunun yanında kuşların başlatılması, durdurma koşulu ve global en iyi KSE ile yerel en iyi KSE'ye ilişkin karşılaştırmalı bilgiler verilecektir.

5.3.1. Hız bileşenleri

Daha önce verilen hız eşitliklerinde de görüleceği gibi bu eşitlik üç bileşenden meydana gelmektedir.

Önceki hız $v_i(t)$, önceki hareket yönünün hafızasıdır. Bu hafıza bileşeni kuşun ani biçimde yön değiştirmesini ve mevcut yönün tersine doğru yönlenmesini önlemek amacı taşır. Bu bileşen ayrıca, *eylemsizlik (inertia)* bileşeni olarak da adlandırılır.

Bilişsel bileşen (cognitive component) $c_1r_1(y_i-x_i)$, *i* kuşunun geçmiş performansına göre şu anki performansını sayısallaştırır. Bilişsel bileşen, kuş için şimdiye kadarki en iyi pozisyon hafızasına benzer. Bu terimin etkisi, en iyi pozisyonlarına geri çekilmelerini sağlamaktır. Kennedy ve Eberhart bu bileşeni kuşun *nostalji*si olarak tanımlamışlardır.

Sosyal bileşen (social component) global en iyi KSE'de $c_2r_2(\hat{y}-x_i)$, yerel en iyi KSE'de $c_2r_2(\hat{y}_i-x_i)$ olarak ifade edilen bu bileşen *i* kuşunun sürüdeki diğer komşulara göre performansını belirtir. Bu bileşenin etkisi, her kuşun komşuları tarafından bulunmuş en iyi pozisyona yönlendirilmesini sağlamaktır.

Bilişsel ve sosyal bileşenlerin katkısı r_1 ve r_2 [0,1] rassal sayılarıyla ağırlıklandırılır. Bu ağırlıkların etkisi ileriki kısımlarda açıklanacaktır [55].

5.3.2. Algoritmanın bakış açısı

KSE'de eniyileme işlemi iteratiftir. Tekrarlı iterasyonlar durdurma koşulu sağlanıncaya kadar devam ettirilir. Böyle bir iterasyon kişisel en iyi pozisyonların, global en iyi pozisyonun belirlenmesi ve her bir kuşun hızının ayarlanması bir döngü içinde gerçekleştirilir. Her iterasyonda bir dizi fonksiyon değerlendirme gerçekleştirilir. Fonksiyon değerlendirme, eniyileme problemini karakterize eden uygunluk fonksiyonunun hesaplanmasıdır. KSE'de her iterasyonda sürüde yer alan kuş sayısı n_s kadar fonksiyon değerlendirme işlemi gerçekleştirilir.

KSE algoritmasının ilk adımı, sürünün ve kontrol parametrelerinin başlatılmasıdır. c_1 ve c_2 hızlanma sabitleri, başlangıç hızları, başlangıç pozisyonları ve kişisel en iyi pozisyonlar belirlenmelidir. Bunun yanında yerel en iyi KSE komşuların büyüklüklerinin belirlenmesini de gerektirir.

Genellikle, kuşların başlangıç pozisyonları arama uzayını düzgün kaplayacak biçimde belirlenir. KSE'nin etkinliği sürünün başlangıç çeşitliliğinden etkilenir. Arama uzayında başlangıç sürüsü tarafından kapsanmayan bölgeler bulunduğunda KSE en iyi değeri bulmakta zorlanır. KSE böyle bir en iyiyi eğer bir kuş kapsanmayan alana girerse ve bu alandaki değer kişisel en iyi veya global en iyi değer olursa bulabilir.

En iyi değerin x_{min} ve x_{max} gibi iki vektörle sınırlandırılması gerektiği durumlarda bu değerler sırasıyla her bir boyutun en küçük ve en büyük değer aralıklarını belirtir. Kuş pozisyonları için etkin başlangıç metodu

$$x(0) = x_{\min,j} + r_j (x_{\max,j} \quad x_{\min,j}),\ \forall_j = 1,..., n_s \qquad r_j \sim U(0,1) \tag{5.7}$$

ile belirlenir.

Başlangıç hızları ise $v_i = 0$ olarak belirlenebilir. Başlangıç hızlarını gerekli olmasa da rassal olarak belirlemek de mümkündür fakat bunu yaparken çok dikkatli olmak gerekir. Gerçekte kuşların başlangıçlarını düşündüğümüzde hızları sıfırdır, kuşlar durağandır. Kuşlar belirli bir başlangıç hızına sahip olursa bu fiziksel benzeşme ortadan kalkar. Pozisyon vektörlerinin rassal başlangıçları rassal pozisyon ve hareket yönlerini garanti eder. Eğer rassal başlangıç hızları kullanılacaksa bu değerler fazla büyük olmamalıdır. Büyük değerler kullanılırsa kuşun başlangıç hızlılığı (momentum) da büyük olur ve bu büyük başlangıç pozisyonu güncellemeleriyle sonuçlanır. Bu durum kuşların arama uzayının dışına çıkmasına ve kuşların tek bir çözüm üzerinde durmadan önce daha fazla algoritmanın daha fazla iterasyon yapmasına neden olur.

Her kuşun kişisel en iyi pozisyonları, kuşun t=0 anındaki başlangıç pozisyonu olarak alınır.

KSE'nin son olarak ele alınacak yönü durdurma koşuludur. Bir durdurma koşulu seçerken dikkat edilmesi gerekenler şu şekilde sıralanabilir:

- Durdurma koşulu KSE'nin erken yakınsamasına neden olmamalıdır.
- Durdurma koşulu, uygunluk fonksiyonunun defalarca hesaplanmasını önlemelidir. Bu durum, algoritmanın hesaplama karmaşıklığının artmasına neden olur.

Önerilen durdurma koşulları şunlardır:

1. En büyük iterasyon sayısı ya da uygunluk fonksiyonunun değerlendirilmesi sayısına verilen eşik aşıldığında durdurma
2. Kabul edilebilir bir çözüm bulunduğunda durdurma
3. Belirli sayıda iterasyon boyunca çözümde iyileşme olmadığında durdurma
4. Normalize edilmiş sürü çapı sıfıra yaklaştığında durdurma

5. Amaç fonksiyonu eğimi sıfıra yaklaştığında durdurma

5.3.3 Global En İyi KSE ile Yerel En İyi KSE'nin karşılaştırılması

Bu iki yaklaşım arasındaki temel farklılıklar şunlardır:

1. Global En İyi KSE kuşlar arasındaki daha fazla etkileşimden dolayı Yerel En İyi KSE'ye göre daha hızlı biçimde yakınsar. Bu daha hızlı yakınsama, yerel en iyi KSE'ye göre arama uzayında daha az farklı noktalara ulaşmasına neden olur.
2. Yerel En İyi KSE'nin arama uzayında daha fazla noktayı arayabilmesinin bir sonucu olarak yerel en iyi değere yakalanma olasılığı daha azdır. Yerel En İyi KSE'de genel olarak, halka topolojiye sahip komşuluk yapıları algoritmanın performansını geliştirmektedir.

5.4. Temel KSE Parametreleri

KSE, problem boyutu, kuşların sayısı, hızlanma katsayıları, eylemsizlik ağırlığı, komşu sayısı ve bilişsel ve sosyal bileşenlerin katkılarını belirleyen rassal değerler gibi birçok kontrol parametresinden etkilenmektedir. Bu bölümde bu parametrelerin etkilerine ilişkin açıklamalara yer verilecektir [56].

Sürünün büyüklüğü: Sürünün büyüklüğü n_s ile ifade edilen sürüde yer alan kuşların sayısıdır. Sürüde daha fazla kuşun yer alması demek sürünün başlangıçta daha fazla nokta araştırması demektir. Diğer bir ifadeyle düzgün dağılmış bir başlangıç şeması elde edilebilir. Fazla sayıda kuşun olması her iterasyonda daha fazla sayıda arama noktasına ulaşılabilmesini sağlar. Bunun yanında fazla sayıdaki kuş her iterasyonda hesaplama zorluğunu artırır ve aramayı paralel rassal aramaya indirger. Daha fazla sayıda kuşun olması daha az sayıdaki parçacığa göre iyi çözüme daha az iterasyonda ulaşılmasını sağlar. Yapılan ampirik çalışmalar sürü büyüklüğünün 10 ile 30 olmasının optimal çözüme ulaşmada yeterli olacağını gösterse de bu değerler probleme bağlı değerlerdir. Düzgün arama yüzeyleri düzgün olmayan aram

yüzeylerine göre daha az sayıda kuş gerektirecektir. Buna karar vermede çapraz sağlama ve deney tasarımı gibi metotlardan yararlanmak daha doğru olacaktır.

Komşu büyüklüğü: Komşu büyüklüğü sürü içindeki sosyal etkileşimin ölçüsüdür. Komşular küçükse bu daha az etkileşimin meydana geleceği anlamına gelir. Daha az komşu yakınsamanın daha az olmasına sebep olurken en iyi çözüme daha güvenilir bir biçimde yakınsama sağlar. Daha az komşunun yerel en küçük değere yakalanma olasılığı daha azdır. Bu nedenle çözüme daha az sayıda komşu ile başlayıp itersyondaki artışa bağlı olarak komşu sayısını artırmak avantajlı bir yaklaşım olacaktır. Bu yaklaşım başlangıçta daha fazla noktanın aranmasını ve yakınsamanın daha hızlı yapılmasını sağlayacaktır.

İterasyon sayısı: İyi sonuca ulaşmak için gerekli olan iterasyon sayısı probleme bağlı olan bir durumdur. Az sayıdaki iterasyon algoritmanın erken yakınsamasına neden olabilir. Çok fazla sayıdaki iterasyon da hesaplama zorluğu yaratır. Fakat bahsedilen bu durumlar yalnızca durdurma koşulu iterasyon sayısına bağlı olduğunda geçerlidir.

Hızlanma katsayıları: Rassal r_1 ve r_2 vektörleriyle birlikte c_1 ve c_2 hızlanma katsayıları parçacıların bilişsel ve sosyal bileşenlerinin stokastik etkilerini kontrol eder. c_1 ve c_2 katsayıları ayrıca *güven parametreleri* olarak da adlandırılırlar. c_1, kuşun kendine ne kadar güvendiğini, c_2 ise kuşun komşularına ne kadar güvendiğini ifade eder. $c_1 = c_2 = 0$ durumunda kuşlar mevcut hızlarıyla arama uzayının sınırlarına gelinceye kadar hareketlerine devam ederler. Eğer $c_1>0$ ve $c_2 = 0$ ise tüm sürü tek bir $\hat{y}$ noktasına çekilir. Bu durumda tüm sürü tek bir stokastik tepe tırmanıcıya dönüşür.

Kuşlar güçlerini dayanışmacı yapılarından alırlar ve c_1 ve c_2 değerleri iyi bir şekilde dengelendiğinde ($c_1 \approx c_2$) daha etkin hale gelirler. $c_1 = c_2$ durumunda kuşlar $\hat{y}$ ve y_i'nin ortalamasına doğru çekilirler. $c_1 >> c_2$ durumunda ise kuşlar aşırı bir sapmayla sonuçlanacak biçimde kendi en iyi değerlerine doğru çekilirler. Diğer yandan $c_2 >> c_1$ durumunda ise kuşlar erkenden en iyi değere doğru hareket etmesine sebep olacak şekilde global en iyi değere doğru çekilirler.İki tepeli (unimodal) düzgün arama

uzayına sahip problemlerde daha büyük c_1 değeri, çok tepeli (multi modal) düzgün olmayan arama uzayına sahip problemlerde daha büyük c_2 değeri avantajlıdır.

c_1 ve c_2'nin küçük değerler alması kuşların iyi çözüm bölgelerine geri çekilmeden iyi çözüm bölgelerinden çok uzaklara gitmesine neden olur. c_1 ve c_2'nin büyük değerler alması ise fazla hızlanmalarına ve beklenmedik yönlere veya önceki iyi çözüm bölgelerine doğru hareketlenmelerine neden olur.

Hız sınırlandırma: Bir eniyileme algoritmasının etkinlik ve doğruluğuna karar veren tarafı *araştırma (exploration)* ve *yoğunlaşma (exploitation)* yetenekleridir. Araştırma, iyi bir en iyi değer elde edebilmek için arama uzayının farklı bölgelerini tarayabilme yeteneğidir. Yoğunlaşma ise, aday bir çözümü daha rafine bir hale getirebilmek için umut veren bir nokta üzerine yoğunlaşmaktır. İyi bir eniyileme algoritması bu çelişen amaçları dengeleyebilmelidir. KSE'de bu amaçların adresi hız güncelleme eşitliğidir.

KSE'nin ilk yıllarında özellikle kişisel en iyi ve komşu en iyi değerlerden uzakta olan kuşların hızlarının çok yüksek değerlere çıktığı görülmüştür. Bu durum kuşların arama uzayının sınırlarının dışına çıkmasına neden olmuştur. Kuşların global araştırmalarını kontrol etmek amacıyla hızların belirli bir aralıkla sınırlandırılması önerilmiştir. $V_{max,j}$ boyutundaki j kuşunun ulaşabileceği maksimum hızı göstermek üzere kuş hızları şu şekilde güncellenmektedir:

$$v_{ij}(t+1) = \begin{cases} v_{ij}(t+1) & if \;\; v_{ij}(t+1) < V_{\max,j} \\ V_{\max,j} & if \;\; v_{ij}(t+1) \geq V_{\max,j} \end{cases} \tag{5.8}$$

v'_{ij}, hız eşitliği kullanılarak hesaplanan değerdir.

V_{max}, j'nin büyük değer alması global arama yapmayı sağlar, küçük değerler ise yoğunlaşma sağlar. Eğer $V_{max,j}$ değeri çok küçükse sürü iyi çözüm bölgelerinde arama gerçekleştiremez, ayrıca en iyi değere ulaşmak için gereken sürenin de

uzamasına neden olur. Sürünün bir daha çıkamamak üzere yerel en küçük tuzağına takılmasına neden olur. $V_{max,\ j}$'nin çok büyük değerler alması da sürünün iyi çözüm bölgelerini kaçırması ve iyi çözüm vermeyen bölgelerde dolaşmasına neden olur.

Hız sınırlamanın yukarıda belirtilen avantajları yanında dezavantajları da mevcuttur. Bunlardan ilki, hız sınırlama sadece adım büyüklüğünü değiştirmez, ayrıca kuşların hareket ettiği yönün de değişmesine neden olur. Hız sınırlamayla ilgili diğer bir problem tüm hızların en büyük değere eşit olduğunda ortaya çıkar. Eğer bunu önlemeye yönelik hiçbir tedbir alınmazsa kuşlar arama uzayının sınırlarında arama yaparlar. Bu sorun eylemsizlik ağırlığı (inertia weight) veya maksimum hızın zaman içinde değişkenlik göstermesi yoluyla giderilebilir. Algoritmaya yüksek hız değerleriyle başlanarak zaman içinde hız değerinin düşürülmesi yoluna gidilebilir.

Eylemsizlik ağırlığı (Inertia weight - w): Eylemsizlik ağırlığı Shi ve Eberhart tarafından arama ve yoğunlaşmasını kontrol edilmesine yardımcı olmak ve hız sınırlandırılmasını elimine etmek için önerilmiştir. Fakat bu bileşen birinci amacını gerçekleştirirken ikinci amacında başarılı olamamıştır. w değeri kuşun momentumunu önceki hızının katkısını dengeleyerek kontrol eder. Diğer bir ifadeyle önceki yöne ilişkin hafızanın yeni hızı nasıl etkileyeceğini belirler. w hız güncelleme eşitliğinde önceki hız değerine çarpan olarak yazılır. $w>1$ durumunda kuşların hızları zaman boyunca maksimum hıza doğru yükselir ve sürü ıraksar. Kuşlar geriye doğru daha iyi bölgelere yön değiştiremezler. Eğer $w<0$ ise parçacılar hızları sıfır oluncaya kadar yavaşlarlar. Büyük w değeri artırılmış ıraksamayla birlikte araştırmayı sağlarken, küçük w değeri yerel yoğunlaşmayı sağlar. Çok küçük w değerleri sürünün araştırma kabiliyetini önler ve bilişsel ve sosyal bileşenlerin pozisyon güncellemesinde daha etkin olmalarını sağlar.

5.6. Sürekli KSE ve Kesikli KSE

KSE algoritmasıyla ilgili olarak buraya kadar anlatılan kısımda algoritma gerçek sayılarla sınırlandırılmıştı. Fakat birçok eniyileme problemi kesikli değerlere ve niteliksel değişkenlere ihtiyaç duymaktadır. Bu ihtiyacı karşılamak için Kennedy ve

Eberhart (1997) kesikli KSE'yi geliştirmişlerdir. Kesikli KSE, sürekli KSE'den iki noktada ayrılmaktadır: Birincisi, kuşlar ikili sayı (0 – 1) değerlerinden oluşmaktadır. İkincisi ise, kuşun 0 ya da 1 değerlerinden hangisini alacağını belirlemek amacıyla hız bir olasılık değişimine dönüştürülmelidir.

Daha önce de bahsedildiği üzere sürekli KSE'de hız güncellemesi aşağıdaki eşitlik kullanılarak gerçekleştirilmektedir:

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_{ij}(t) - x_{ij}(t)] \tag{5.9}$$

Bu eşitliğe göre her kuş her itarasyonda yeni bir hıza sahip olmaktadır. Kuşların ikili sayı değeri alacağı göz önüne alınırsa her x_{ii}'nin 1 değeri alma olasılığını belirlemek amacıyla aşağıdaki sigmoid fonksiyonu kullanılır:

$$s(v_{ij}(t)) = \frac{1}{1 + \exp(-v_{ij}(t)} \tag{5.10}$$

Böylelikle hız değerini [0,1] aralığında belirledikten sonra x_{ij}'nin 0 ya da 1 değerlerinden hangisini alacağını belirlemek gerekmektedir. Bunun için de aşağıdaki eşitlikten yararlanılır:

$$x_{ij}(t) = \begin{cases} 1, & if \quad U(0,1) < s(v_{ij}(t)) \\ 0, & d.d. \end{cases} \tag{5.11}$$

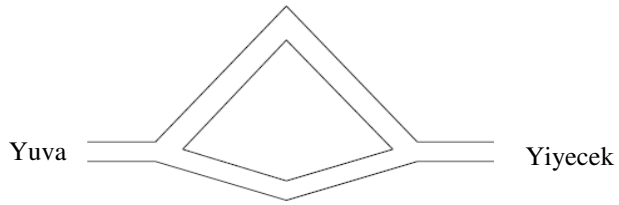
Buna göre düzgün dağılıma göre (0,1) aralığında üretilen rassal sayı sigmoid fonksiyonuyla dönüştürülen hız değerinden küçükse x_{ij} 1 değerini, değilse 0 değerini alır.

Kesikli KSE'nin sürekli KSE'den farklı olan yanı yalnızca yukarıda değinilen kısımdadır. Kesikli KSE için de yerel en iyi KSE ve global en iyi KSE için yapılan açıklamalar geçerlidir [57].

6. KARINCA KOLONİSİ ENİYİLEMESİ

Karınca algoritmaları, gerçek karıncaların davranışlarından etkilenilmiş ajanlardan meydana gelen çok ajanlı sistemlerdir. Karınca algoritmaları, gezgin satıcı probleminden haberleşme şebekelerinin yönlendirilmesine kadar birçok problem türünde kullanılmış sürü zekası sistemlerinin başarılı bir örneğidir.

Karınca Kolonisi Optimizasyonu (Ant Colony Optimization – KKE) algoritması, Marco Dorigo tarafından Goss ve arkadaşlarının gerçek karıncaları kullanarak yaptıkları bir deneyden elde edilen sonuçlara dayanarak geliştirilmiştir. Arjantin karıncalarından oluşan bir laboratuar kolonisine bulundukları yuva ile yiyecek arasına farklı uzunluktaki iki koldan oluşan bir köprü yerleştirilmiştir. (Şekil 6.1) Köprünün kolları karıncaların her iki yönde kollardan birini seçebileceği şekilde yerleştirilmiştir [58].



Şekil 6.1. Karınca deneyinde kullanılan köprü [59]

Deneyin sonucunda birkaç dakikalık bir geçiş aşamasından sonra karıncaların tümü en kısa köprü kolunu kullanmaya başlamışlardır. Bunun yanında, koloninin en kısa kolu seçme olasılığının köprünün kolları arasındaki uzunluk farkına bağlı olarak arttığı gözlenmiştir. Bu en kısa yol seçme durumunun ortaya çıkması pozitif geri besleme ve farklı yol uzunlukları türünden ifade edilebilir. Ayrıca bu durumun ortaya çıkmasında dolaylı haberleşme formlarının da etkisi bulunmaktadır.

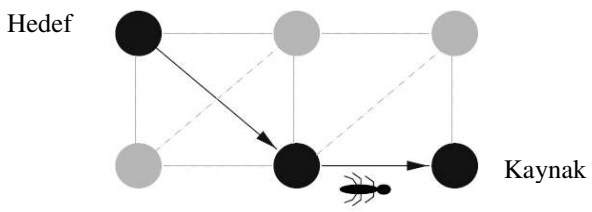
Aslında Arjantin karıncaları yuvalarından yiyeceğe giderken ve tersi yönde hareket ederken geçtikleri yerlere *feromen (pheromen)* adı verilen kimyasal bir madde bırakmaktadırlar. Karıncalar köprünün kısa ve uzun kollarının kesiştiği karar verme

noktasına geldiklerinde hangi taraftan gideceklerine geçiş yollarına bırakılan feromen maddesini koklayarak olasılıklı bir seçim ile karar vermektedir. Bu durum seçilen bir yolun diğer karıncalar tarafından seçilme olasılığını artıran bir otomatik katalizör görevi görmektedir. Deneyin başlangıcında köprünün kollarının hiçbirinde feromen bulunmamaktadır. Bu nedenle karıncalar hangi taraftan gideceklerinin kararını eşit olasılıkla vermektedirler. Köprü kollarının farklı uzunluklarından dolayı daha kısa yolu seçen karıncalar yiyeceğe en önce varan karıncalar olacaktır. Bu karıncalar yuvaya geri dönerken kesişme noktasına geldiklerinde önceden bıraktıkları feromen izine bağlı olarak daha büyük olasılıkla daha kısa olan yolu seçeceklerdir. Seçilen yola dönüşte yeniden feromen bırakılacak ve bu yol diğer karıncalar için daha fazla tercih edilir hale gelecektir. İşlem ilerlerken kısa olan yolda daha fazla feromen birikecek ve bu yol karıncalar tarafından daha fazla kullanılır hale gelecektir [59].

6.1. Temel Karınca Kolonisi Optimizasyonu Algoritması

Her bir yapay karıncanın amacı doğal karıncalarda olduğu gibi çözülen problemi ifade eden bir grafta yer alan iki düğüm arasındaki en kısa yolu bulmaktır.

G = (V,E) şeklinde ifade edilen, *n* tane düğüme sahip bağlı bir graf düşünelim. Bu grafta *s* kaynak düğümü ve *d* hedef düğümlerini birbirine bağlayan en kısa yol uzaklığın düğüm sayısıyla ifade edilmesi durumunda karınca kolonisi optimizasyonu algoritması kullanılarak bulunabilir.



Şekil 6.2. Karıncaların çözüm üretme şekli [59]

Grafta yer alan her (*i*,*j*) arkı τ_{ij} ile gösterilen feromen miktarı ile ilişkilendirilir. Bu feromen miktarları karıncalar tarafından okunur ve yazılır. Feromen miktarının yoğunluğu bu arkı kullanmakla elde edilecek yararın oransal ifadesidir. Her bir karınca problemin çözümünü bulmak için adım adım kurucu bir karar politikası izler. Her bir düğümde, düğümden veya onu izleyen arktan elde edilen yerel bilgiye bağlı olarak hareket edilecek düğüm stokastik bir biçimde seçilir.

i düğümünde yer alan *k* karıncası bulunduğu düğümden sonra hangi $j \in N_i$ düğümüne yönleneceğine karar verirken τ_{ij} feromen miktarını kullanır. N_i, karıncanın bulunduğu *i* düğümüne bir adım uzaklıkta komşu olan düğümlerin kümesini göstermektedir. Unutulmamalıdır ki arama prosesinin başlangıcında her arkta τ_0 kadar bir başlangıç feromeni bulunmaktadır.

$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ik}}{\sum_{j \in N_i} \tau_{ij}} & if \quad j \in N_i \\ 0 & if \quad j \notin N_i \end{cases} \tag{6.1}$$

Çözüm süreci devam ederken karıncalar feromen miktarlarını üzerlerinden geçtikleri arklar üzerine kaydederler. Bu arklar üzerindeki feromen miktarları temel KKE algoritmasında $\Delta\tau$ kadar sabit olarak artırılır. *t* anında *i* düğümünden *j* düğümüne hareket eden bir karınca *(i,j)* arkındaki feromen miktarını Eş. 6.2'deki gibi artırır:

$$\tau_{ij}(t) \leftarrow \tau_{ij}(t) + \Delta\tau \tag{6.2}$$

Bu kural sayesinde *(i,j)* arkını kullanan karınca diğer karıncaların da gelecekte bu arkı kullanma olasılıklarını artırmaktadır. Hızlı bir alt-optimal yakınsamadan kaçmak amacıyla bir araştırma mekanizması eklenmiştir. Buna göre, gerçek feromenlerde olduğu gibi bu yapay feromenlerde de buharlaşma olmaktadır. Bu durumda tüm algoritma boyunca araştırmayı desteklemek amacıyla feromen yoğunluğu kendiliğinden azalmaktadır. Her iterasyonda feromen miktarlarının azaltılması üstel

ve basit bir biçimde yapılmaktadır. $\tau \leftarrow (1-\rho)\tau$, $\rho \in (0,1]$ Yapılan deneysel çalışmalar KKE algoritmasının etkin bir algoritma olduğunu göstermiştir. Ayrıca yapılan çalışmalar grafta yer alan düğüm sayısı arttıkça algoritmanın durağanlığı azaltmakta olduğunu ve parametrelerin seçilen düzeyleri kritik öneme sahip olduğunu göstermiştir.

Temel KKE basitliğinden dolayı bazı sınırlamalara sahiptir. İleriki kısımlarda bu kısıtlamaları gidermek amacıyla algoritmada yapılan değişikliklere değinilecektir. Örneğin, karıncalar tarafından bırakılan feromen miktarları elde edilen çözümün kalitesine oranlanarak karıncaların yönlendirilmesinde daha etkili olunabilir. Ayrıca birçok problemde düğümlerde probleme ilişkin sezgisel bilgi (heuristic information) yer aldığından bu bilginin de karıncalar tarafından kullanılması sağlanabilir.

Bu kısımda anlatılan temel KKE algoritması kısıtsız olarak en kısa yol problemine uygulanabilir. Eğer Hamiltonian yola sahip bir grafta (elde edilen yolun düğümlere yalnızca bir kere uğraması) uygulamak istenirse karıncalara sınırlı bir hafıza şekli kazandırılması gerekmektedir [59].

Bir sonraki kısımda temel KKE'nin zenginleştirilmiş hali olan KKE sezgiseline değinilecektir.

6.2. Karınca Kolonisi Optimizasyonu Sezgiseli

KKE algoritmasında kullanılan karıncalar olasılıklı olarak çözüm üreten ve her iterasyonda kısmi çözümlere çözüm bileşenleri ekleyen ajanlardır. Bu ajanlar (i) çözülen mevcut problemde eğer varsa sezgisel bilgiyi kullanarak, (ii) karıncanın arama tecrübesini algoritmanın çalışma anında dinamik bir şekilde yansıtan feromenler üreterek çözümü gerçekleştirirler.

Yapıcı sezgisellerin bir türü olarak KKE birkaç nedenden dolayı ilgi çekici özelliklere sahiptir. Birincisi, KKE içinde yer alan stokastik bileşen karıncaların çok

fazla sayıda çözüm üretebilmelerine ve böylece aç gözlü (greedy) sezgisellere göre daha fazla sayıda çözümü inceleyebilir. Aynı zamanda sezgisel bilginin kullanımıyla da karıncaları en fazla umut veren çözümlere doğru yönlendirir. İkincisi, karıncalar tarafından elde edilen arama tecrübesi algoritmanın ileriki iterasyonlarında çözüm üretmede kullanılabilir. Ayrıca karınca kolonisinin kullanılması algoritmaya güçlü olma özelliği katmaktadır.

Daha önce de belirtildiği gibi karıncalar *G(V,E)* grafı üzerinde hareket ederek çözüm üreten stokastik üretim ajanlarıdır. Karıncalar *G* grafı üzerinde rasgele hareket etmezler. Problemin kısıtlarını sağlayacak biçimde bir çözüm üretme politikası izlerler. Genellikle karıncalar geçerli çözümler üretirler fakat gerekiyorsa geçersiz çözümler de üretebilirler. *G* grafının bileşenleri $c_i \in V$ ve $l_i \in E$ karıncaların çözüm elde etmesinde yönlendirici rol oynayan iki tür bilgiyi barındırır. Bunlardan ilki *feromen miktarı* τ (düğümlerle ilişkili ise τ_i, arklarla ilgili ise τ_{ij}) değeridir. Bu değer tüm arama prosesine ilişkin uzun dönemli hafızayı temsil eder. İkincisi ise algoritmanın çalışma süresi boyunca karıncalardan farklı kaynaklardan elde edilen *sezigsel bilgi* η (η_i ve η_{ij}) dir. Birçok problemde η maliyet veya maliyet tahmini gibi mevcut durumu belirtir. Bu değerler karıncaların graf üzerinde nasıl hareket edeceklerini belirleyen olasılıklı kararı vermelerini sağlar.

Her *k* karıncası aşağıdaki özelliklere sahiptir:

- Karınca *G(V,E)* grafında geçerli çözümü bulabilmek için yoğunlaşma şeklinde bir arama yapar.
- Karınca algoritma boyunca izlediği yollara ilişkin bilgilerin tutulduğu M^k hafızasına sahiptir. Bu hafıza geçerli çözümler üretmek, bulunan çözümü değerlendirmek ve feromen biriktirmek için gelinen yolun tersi yönde gitmek için kullanılır.
- Karıncaya bir başlangıç durumu ve bir veya birden fazla sonlandırma koşulu atanabilir. Genellikle başlangıç durumu bir düğüm olarak ifade edilir, fakat boş bir dizi de olabilir.

- Genellikle geçerli çözümlere doğru olan hareketler desteklenir. Bu destekleme düzgün tanımlanmış η sezgisel değeriyle veya karıncanın hafızasındaki bilgilerle olur.
- Karınca olasılıklı geçiş kuralını kullanarak hareketini gerçekleştirir. Olasılıklı geçiş kuralı şu üç bileşenin fonksiyonudur: (i) yerel olarak bulunan feromen izleri ve sezgisel değer, (ii) karıncanın geçmişine ait bilgileri tuttuğu özel hafızası ve (iii) problemin kısıtları
- *k* karıncasının arama süreci en az bir durdurma koşulu sağlandığında sonlanır.
- Çözüme bir düğüm eklendiğinde karınca feromen miktarını düğümü veya bağlı arkı kullanarak günceller. Buna *çevrimiçi adım adım feromen güncelleme (online step by step pheromen update)* adı verilir.
- Bir çözüm üretildikten sonra karınca geldiği yolun tersi yönde hareket ederek kullanılan düğüm ve arklardaki feromenleri günceller. Buna *çevrimiçi gecikmiş feromen güncelleme (online delayed pheromen update)* adı verilir.

Karıncalar birbirlerinden bağımsız ve eş zamanlı olarak hareket ederler. İyi çözümler feromen bilgilerini okuyup yazarak oluşturdukları dolaylı iletişim sonucu elde edilen kolektif etkileşimin sonucu olarak ortaya çıkarlar. Bu, karıncaların kendilerini değil problemin sunuş şekline uyarladıkları bir dağıtık öğrenme (disturbed learning) şeklidir [60].

6.3. Algoritma

Karınca kolonisi tesadüfi ve eş zamansız bir biçimde *G* grafı üzerinde yollar yaparak ilerler. Karıncalar ilerlerken sezgisel bilgi ve feromen izlerine bağlı olarak olasılıklı bir geçiş kuralına göre seçim yaparlar. Bu ilerleme sırasında karıncalar eniyileme problemine çözüm kurarlar. Her bir karınca, çözüm ürettikten sonra elde edilen çözümü değerlendirerek düğümler veya arklar üzerindeki feromen miktarlarını güncellerler. Bu güncellenen feromen bilgisi gelecek karıncaların yönlendirilmesinde etkili olacaktır.

Karıncaları aktivitelerinin yanında KKE algoritması iki adet prosedüre daha sahiptir: *feromen buharlaştırma* ve *arka plan faaliyetleri(daemon actions)*. Arka plan faaliyetleri isteğe bağlı olarak kullanılmaktadır. *Feromen buharlaştırma,* düğüm veya arklardaki feromen miktarının zaman içinde azaltılması demektir. Feromen buharlaştırma, algoritmanın erken yakınsamasını engellemek amacıyla gerçekleştirilir. Bu durum yararlı bir *unutma* sağlar. Diğer bir ifadeyle arama uzayında yeni noktaların araştırılmasını destekler. Arka plan faaliyetler ise tek tek karıncalar tarafından gerçekleştirilemeyen merkezi faaliyetlerin gerçekleştirilmesinde kullanılır. Buna örnek olarak yerel optimizasyon modelinin aktive edilmesi veya aramayı yanlılaştırmak için arklar üzerinde daha fazla feromen biriktirme veya biriktirmeme kararının verilebilmesi için global bilginin toplanması verilebilir. Örneğin en iyi çözümü bulan karıncanın geçtiği arklar üzerinde daha fazla feromen biriktirilmesi kararı da arka plan faaliyetler tarafından verilir. Aşağıda KKE'ye ait algoritma verilmektedir:

procedure KKE genel amaçlı sezgiseli
Aktiviteleri zamanla
Karınca faaliyetlerini yönet ( )
Feromen buharlaştır ( )
Arka plan faaliyetleri (isteğe bağlı)
end Aktiviteleri zamanla
end KKE genel amaçlı sezgiseli

Şekil 6.3'te yer alan *KKE genel amaçlı sezgiseli* prosedürü aktivitelerin zamanlanmasını yönetir. Aktiviteleri zamanlaması daha önce de değinilen üç KKE bileşenin zamanlanmasını içerir: (i) karıncaların aktivitelerinin yönetilmesi, (ii) feromen buharlaştırma, (iii) arka plan faaliyetleri. *Aktiviteleri zamanla*, bu üç aktivitenin nasıl zamanlanacağını ve senkronize edileceğiyle ilgilenmez. Diğer bir ifadeyle, bu aktivitelerin bağımsız ya da paralel şekilde gerçekleştirilip gerçekleştirilmeyeceğini belirlemez. Tasarımcı bu aktivitelerin ne şekilde etkileşeceğini belirlemede özgürdür [60].

6.4. Karınca Kolonisi Eniyilemesi Algoritması'nın Varyasyonları

KKE algoritması, ortaya atıldığından bu yana çeşitli uyarlamalarla farklılıklara uğrayarak çeşitli algoritmalar türetilmiştir. Bu algoritmalara ilişkin ayrıntılara aşağıda yer verilmiştir.

6.4.1. Karınca sistemi

Dorigo ve arkadaşları tarafından gezgin satıcı problemini çözmek üzere KKE'ye dayalı ilk algoritmadır. Karınca Sistemi'nde her karınca tarafından bırakılan tüm feromenlerde buharlaşma gerçekleşir. Bunun yanında feromenler ilgilenilen çözümün kalitesine bağlı olarak bırakılır [61].

Karınca Sistemi'nde çözümler şu şekilde oluşturulur: Her adımda *j* pozisyonundaki her *i* karıncası bir sonraki *j+1* pozisyonuna Eş. 6.3'teki geçiş kuralına bağlı olarak geçer:

$$p_{jk} = \left[\frac{(\tau_{jk})^{\alpha} \cdot (\eta_{jk})^{\beta}}{\sum (\tau_{jk})^{\alpha} \cdot (\eta_{jk})^{\beta}} \right] \tag{6.3}$$

Bu eşitlikte η_{jk}, geçişin sezgisel istenilirliğini α ve β ise feromen miktarı ve sezgisel bilginin göreceli önemini ağırlıklandıran parameterelerdir. Feromen buharlaştırması Eş. 6.4 kullanılara tüm feromenlerin azaltılmasıyla gerçekleştirilir:

$$\tau_{jk} = \left[(1-\rho) \frac{\tau_{jk}}{\sum \tau_{jk}} \right] \tag{6.4}$$

$\rho \in (0,1]$ buharlaşma oranıdır. Bu adımdan sonra her *i* karıncası geçtiği rota üzerindeki feromen biriktirir.

$$\tau_{jk} = \tau_{jk} + \Delta\tau_{jk} \tag{6.5}$$

$$\Delta\tau_{jk} = \frac{1}{F(x)} \tag{6.6}$$

F(x), i karıncasının oluşturduğu çözümün uygunluk fonksiyonu değeridir. Uygunluk fonksiyonu ise oluşturulan çözümün kalitesidir. Karınca Sisremi'nin algoritması aşağıda verilmiştir.

for her koloni *do*
for her karınca *do*
rota üret
rotayı değerlendir
tüm feromenleri belli oranda buharlaştır
feromen eklemeyi gerçekleştir
end
end

6.4.2. Karınca kolonisi sistemi

Dorigo ve Maniezzo tarafından gezgin satıcı problemini çözmek için geliştirilmiştir. Bu yapıda feromen güncellemenin yanında her bir karınca rotasını tamamlamadan önce global en iyi rotada ekstra bir güncelleme yapar. Bu, ρ_2 buharlaşma oranıyla gerçekleştirilir. Feromen eklemede Eş. 6.5 kullanılır. Bir sonraki eşitlik ise σ sabiti kullanılarak modifiye edilir [62].

$$\Delta\tau_{jk} = \frac{1}{\sigma F(x)} \tag{6.7}$$

Karınca Kolonisi Sistemi'nde rota seçimi q_0 karar indeksi kullanılarak iki şekilde gerçekleştirilir. Her iterasyonda rassal bir q değeri seçilir ve bu değer eğer

algoritmanın başlangıcında belirlenen q_0 değerinden büyük ise karınca bir önceki eşitlikle belirlenen yolu seçer. Diğer durumda ise aşağıdaki eşitlikle belirlenen rotayı seçer.

$$k = \arg\max_l \tau_{jl} \tag{6.8}$$

Karınca Kolonisi Sistemi'nin algoritması aşağıda verilmiştir.

for her koloni *do*
for her karınca *do*
rota üret
rotayı değerlendir
tüm feromenleri belli oranda buharlaştır (ρ oranında)
feromen eklemeyi gerçekleştir
end
en iyi global rotadan feromen buharlaştır. (ρ_2 oranında)
global en iyi rotada feromen eklemeyi gerçekleştir
end

6.4.3. Max-min karınca sistemi

Stützle ve Hoos tarafından geliştirilmiştir. Max-Min Karınca Sistemi, feromen biriktirme ve buharlaşma aşamalarında farklılık gösterir. Feromen buharlaştırma işlemi tüm feromenlerde gerçekleştirilir, fakat feromen ilavesi yalnızca global en iyi veya yerel en iyi rotada gerçekleştirilir. (Eş. 6.5 ve Eş. 6.7'deki gibi) Max-Min Karınca Sistemi'nin algoritmasının adımları aşağıda verilmiştir [63].

for her koloni *do*
for her karınca *do*
rota üret
rotayı değerlendir

end
en iyi global veya yerel en iyi rotayı belirle
tüm feromenlerden buharlaşma gerçekleştir
global en iyi rotada feromen biriktir
end

6.4.4. Sıra bazlı karınca sistemi

Bullnheimer tarafından Karınca Sistemi'nin bir varyasyonu olarak feromenlerin güncellenmesinde sıra fikrini kullanır.
Buna göre, *i.* karınca feromen bıraktıktan ve feromen buharlaştırmaları gerçekleştirildikten sonra kolonideki tüm *F(x)* uygunluk fonksiyonları değerlendirilerek uygunluk değerleri çözüm kalitesine göre sıralanırlar. Genel olarak feromen ekleme için iyi sıralanan çözümlerin %25'i kullanılır. Eş. 6.7'deki σ değeri σ_2 değeri ile değiştirilir. En iyi global rotada Eş. 6.7 kullanılarak feromen güncelleme yapılır. Böylelikle sadece global en iyi rotayı desteklemek yerine elit rotalar arasında araştırma yapılır. Sıra Bazlı Karınca Sistemi'nin algoritması aşağıda verilmiştir [64].

for her koloni *do*
for her karınca *do*
rota üret
rotayı değerlendir
end
en iyi global veya yerel en iyi rotayı belirle
tüm feromenlerden buharlaşma gerçekleştir
elit rotalara Y kadar feromen ilave et
global en iyi rotaya X kadar feromen ilave et
end

6.4.5. En iyi – en kötü karınca sistemi

Cordon ve arkadaşları tarafından Karınca Sistemi'nin bir varyasyonu olarak ortaya atılmıştır. En İyi – En Kötü Karınca Sistemi, iki tür feromen buharlaştırma yaklaşımı uygular. Birinci olarak her koloninin sonunda ρ oranıyla ve ikinci olarak da ρ_2 oranıyla kolonideki en kötü rotada gerçekleştirilir. Feromen güncelleme ise yalnızca en iyi rotada gerçekleştirilir. En İyi – En Kötü Karınca Sistemi'nin adımları aşağıda verilmiştir [65].

for her koloni *do*
for her karınca *do*
rota üret
rotayı değerlendir
end
en iyi global veya yerel en iyi rotayı belirle
tüm feromenlerden buharlaşma gerçekleştir (ρ oranında)
mevcut kolonideki en kötü rotayı belirle
en iyi global rotaya feromen ilave et
mevcut kolonideki en kötü rotadan global en iyi rotada yer almayan arklardan feromen buharlaştırması gerçekleştir (ρ_2 oranında)
end

7. HABERLEŞME ŞEBEKELERİNİN TASARIMI İÇİN GELİŞTİRİLEN ALGORİTMALAR

Çalışmanın bu bölümünde haberleşeme şebekelerinin tasarımı için bir önceki bölümde özellikleri anlatılan sezgisellere dayalı olarak geliştirilen algoritmaların ayrıntıları verilecektir. Geliştirilen algoritmaların çözüm kalitesini artırmak amacıyla çeşitli karma yaklaşımlar ve algoritma parametrelerine dinamiklik kazandırma yaklaşımı uygulanmıştır.

7.1. Problemin Tanımı ve Varsayımları

Bu çalışmada güvenilirlik kısıtı altında minimum maliyetli şebekelerin tasarımı problemi üzerine çalışılmıştır. Bu probleme ilişkin matematiksel ifade aşağıdaki gibidir:

$$F(x) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} c_{ij} x_{ij}$$

$$s.t. \quad R(x) \geq R_0$$

$$Yol\ sayısı \geq 2 \quad \forall\ (i, j)\ düğüm\ çifti\ için$$

$$x_i = 0\ ya\ da\ 1 \qquad \forall i \in L$$

Bu matematiksel modelde;

N: şebekedeki toplam düğüm sayısı,

c_{ij}: i düğümü ile j düğümü arasındaki hattın maliyeti,

x: i,j hatlarından oluşan aday şebeke

$$x_{ij} = \begin{cases} 1, & \text{(i,j) düğüm çifti arasında hat varsa} \\ 0, & \text{d.d} \end{cases}$$

R(x): x aday şebekesinin güvenilirliği

R_0: Şebeke için istenen güvenilirlik düzeyinin alt sınırı anlamına gelmektedir.

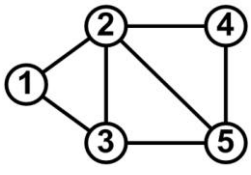
Güvenilirlik kısıtına ilave olarak elde edilecek şebekenin 2-bağlılık kısıtını sağlaması da gerekmektedir. 2-bağlı şebekelerde her düğüm çifti arasında en az iki farklı yol bulunmaktadır.

Bu probleme ilişkin diğer varsayımlar ise şunlardır:

- Şebekedeki düğümlerin konumları bellidir.
- Düğümler tam güvenlidir.
- Bütün c_{ij} ve p (hatların maliyeteri ve çalışma olasılıkları) bilinmektedir.
- Hatlar ya çalışmaktadır ya da arızalıdır.
- Hatların arızalanma olasılıkları birbirinden bağımsızdır.
- Arızalı hattın tamirine izin verilmemektedir.

7.2. Aday Çözümlerin Gösterimi

Haberleşme şebekelerinin güvenilirlik kısıtı altında minimum maliyetli tasarımı problemine ilişkin verilen matematiksel modelde de ifade edildiği üzere bir haberleşme şebekesi ikili (0 – 1) sayı gösterimiyle ifade edilmektedir. N, şebekedeki düğüm sayısı olmak üzere bir çözümün uzunluğu $N(N-1)$ kadar olacaktır. Bu değer ayrıca tam bağlı bir şebekedeki mümkün hatların sayısıdır. Eğer şebekedeki bir (i,j) hattı bağlı ise $x_{ij} = 1$, hat bağlı değilse $x_{ij} = 0$ değerini alacaktır. Şekil 7.1 (a) 5 düğümlü bir şebekedeki aday çözümü ve şekil 7.1 (b) aday şebekenin 0 – 1 gösterimini vermektedir.



x_{12}	x_{13}	x_{14}	x_{15}	x_{23}	x_{24}	x_{25}	x_{34}	x_{35}	x_{45}
1	1	0	0	1	1	1	0	1	1

Şekil 7.1. Bir aday şebeke ve bu şebekenin 0 – 1 gösterimi
a) 5 düğümlü bir şebekede aday çözüm b) Aday şebekenin 0 -1 gösterimi

7.3. Değişken Komşu Arama İle Haberleşme Şebekelerinin Tasarımı

4. bölümde de değinildiği üzere değişken komşu arama metodunun 4 adet varyasyonu bulunmaktadır. Haberleşme şebekelerinin topolojik tasarımı probleminde bu varyasyonlardan *değişken komşu iniş (DKİ)* kullanılmıştır.

7.3.1. Başlangıç çözümünün elde edilmesi

DKİ algoritması için öncelikle bir başlangıç çözümünün oluşturulması gerekmektedir. Başlangıç çözümünün oluşturulmasında stokstik derinlemesine arama algoritması (stochastic depth – first search algorithm) kullanılmıştır. Düğümlerin kümesi $N_r = \{1,2,...,N\}$ ve oluşturulan hatların kümesi ise $L_r = \{\ \}$ olmak üzere başlangıç yayılan ağacın oluşturulması aşağıda verilmiştir.

Başlangıç: $N_r = \{1,2,....,N\}$ ve $L_r = \{\ \}$
Adım 1. N_r'den rassal olarak 2 adet düğüm seç ve elde edilen hattı L_r'ye kaydet
$N_r = \{\ \}$ oluncaya kadar aşağıdaki adımları tekrarla
Adım 2. N_r'den rassal olarak bir düğüm seç.
Adım 3. Seçilen düğümün Lr'de yer alan düğümlerden birine rassal olarak ekle.
Adım 4. N_r ve L_r'yi güncelle.

Şebeke oluşturulurken, N_r'den seçilen düğüm, önceki iterasyonlarda seçilmiş olan düğümlere eklenirken rassal olarak seçilen bir düğüme eklenmektedir.. Böylelikle elde edilen şebeke bir yayılan ağaç olmaktadır. Bununla beraber, elde edilen şebeke doğal olarak 2-bağlılık ve güvenilirlik kısıtlarını sağlayamamaktadır. Elde edilen bu şebekeye 2-bağlılık kısıtını sağlamak üzere bir tamir algoritması uygulanmaktadır. Aşağıda algoritmanın adımlarını vermeden önce kullanılan notasyonlara ait açıklamalar şu şekildedir:

N_k: k derecesine sahip düğümlerin kümesi
N_{min}: derecesi 1 olan düğümler hariç minimum dereceli düğümlerin kümesi
n_k: N_k kümesindeki düğümlerin sayısı

m_{1j}: N_1 kümesindeki düğüm etiketleri

$m_{min\ j}$: N_{min} kümesindeki düğüm etiketleri $j = 1,2,\ldots,N_{min}$

Adım 1. N_k ve n_k'yı oluştur. (k = 1,2…,şebekedeki maksimum düğüm derecesi

Adım 2. N_1 ve n_1 haricinde tüm N_k ve n_k'ları (k = 2,…,maksimum düğüm derecesi) artan şekilde sırala; N_{min} ve n_{min}'e karar ver

a) $n_1 = 1$ ise bu düğüm ve N_{min}'deki düğümlerden minimum maliyetli olanı bağla

b) $n_1 = 2$ ise

i. N_1 kümesinde yer alan iki düğümün bağlanma maliyetini hesapla ($c_{m11,m12}$)

ii. $j = 1,2,\ldots,n_{min}$ için bütün $c_{m12,m_{min\ j}}$ değerlerini hesapla

iii. $c_{m11,m12} < [min\ (c_{m11,m_{min\ j}}) + min(c_{m11,m_{min\ j}})]$ ise n_1 kümesindeki iki düğümü bağla, değilse n_1'deki düğümleri N_{min} kümesindeki düğümlerle bağla. [$min(c_{m11,m_{minj}}), min(c_{m12,\ m_{min\ j}})$]

c) $n_1 > 2$ ise

i. N_1 kümesinden rassal olarak iki düğüm seç

ii. b adımını bu iki düğüme $n_1 = 0$ oluncaya kadar uygula

Kaliteli çözümler elde edebilmek için başlangıç çözümünün güvenilirlik kriterini de sağlaması gerekmektedir. Bu nedenle 2-bağlılık tamir algoritmasından sonra elde edilen şebekenin güvenilirlik düzeyinin bilinmesi gerekmektedir. Eğer güvenilirlik kısıtı sağlanmıyorsa şebekeye hat ilave etmeye devam edilecektir. Güvenilirliğin tahmini daha önce de değinildiği gibi oldukça zaman alıcı olduğundan başlangıç çözümünün güvenilirlik üst sınırının ($R_{ÜS}$) belirlenmesi yoluna gidilmiştir. Eğer şebekenin güvenilirlik üst sınırı ($R_{ÜS}$) değeri istenen güvenilirlik değerinin altında kalıyorsa güvenilirlik üst sınırı istenen güvenilirlik değerine ulaşıncaya kadar hat eklenmeye devam edilmiştir. Hat ekleme, minimum maliyetli hat seçilerek gerçekleştirilmiştir.

7.3.2. Komşuluk yapılarının belirlenmesi

Başlangıç çözümünün elde edilmesinden sonraki aşama değişken komşu iniş algoritmasında kullanılacak olan komşuluk yapılarının belirlenmesidir. Değişken

komşu iniş, bir yerel aramada komşuluk yapılarının sistematik olarak değiştirilmesine dayanır. Bu çalışmada ele alınan haberleşme şebekelerinin topolojik tasarımı probleminin değişken komşu iniş algoritması ile çözümünde 3 adet komşuluk yapısı kullanılmıştır ($N = 3$). Bu komşuluk yapılarına ilişkin ayrıntılar şöyledir:

N_1*:* Bu komşuluk yapısında ekle/çıkar hareketi kullanılmıştır. Çözüm elde edildikten sonra şebekedeki bir hat şebekede bulunmuyorsa eklenmekte, şebekede bulunuyorsa çıkarılmaktadır. Diğer bir deyişle, her aşamada şebekedeki bir adet hat değiştirilmektedir. Bu komşuluk yapısı için başlangıç olarak kullanılan bir şebekeden elde edilebilecek farklı çözüm sayısı hat sayısı kadardır.

N_2*:* Bu komşuluk yapısında ekle ve çıkar hareketi birlikte kullanılmıştır. Şebekede var olan bir hat var olmayan bir hatla yer değiştirmektedir. Diğer bir ifadeyle çözüm elde etmenin her aşamasında iki adet hat değiştirilmektedir. Bu komşuluk yapısında elde edilebilecek farklı çözüm sayısı şebekede var olan hatların sayısı ile var olmayan hatların sayısının çarpımına eşittir.

N_3*:* Bu komşuluk yapısında ise ekle ve çıkar hareketiyle birlikte ekle/çıkar hareketi gerçekleştirilmektedir. Ekle/çıkar hareketi şebekeye yeni giren veya şebekeden yeni çıkan hatlar dışındaki hatlara uygulanmaktadır. Bu komşuluk yapısında da N_2 komşuluk yapısında olduğu gibi şebekede var olan hatların sayısı ile var olmayan hatların sayısının çarpımı kadar farklı çözüm elde edilebilmektedir.

7.3.3. Şebekenin bağlılık kontrolü

Komşuluk yapılarıyla elde edilen şebekelerin geçerli olabilmesi için mutlaka bağlı şebeke olmaları gerekmektedir. Şebekenin bağlı olması en azından bir yayılan ağaç olması anlamına gelmektedir. Şebekelerin bağlılık kontrolü için Hopcraft ve Ullman (1976) tarafından geliştirilen *küme birleştirme algoritması* kullanılmıştır. Aşağıda bu algoritmanın adımları yer almaktadır:

Adım 1. Şebekedeki her düğümü ayrı kümelere ata
Adım 2. Şebekeden bir hat seç. Bu hattın birleştirdiği düğümler aynı kümede ise başka bir hat seç, değilse Adım 3'e git.
Adım 3. Bu düğümlerin kümelerini birleştir. Şebekedeki tüm hatlar seçilmişse adım 4'e git, değilse adım 2'ye git.
Adım 4. Şebekedeki düğümlerin hepsi bir kümede toplanmışsa şebeke bağlı, değilse şebeke bağlı değil. DUR

7.3.4. Değişken komşu iniş algoritması

Komşuluk yapılarıyla elde edilen şebekelerin bağlılık kontrolleri gerçekleştirildikten sonra aday çözümler arasından minimum maliyetli şebekenin seçilerek güvenilirliğinin tahmin edilmesi gerekmektedir. Belirlenen güvenilirlik düzeyini sağlayan minimum maliyetli bu şebeke bir sonraki iterasyonun başlangıç çözümü olacaktır. Seçilen minimum maliyetli şebekenin güvenilirliğini tahmin etmeden önce Bölüm 2.5.3'te anlatıldığı biçimde Jan (1993) tarafından geliştirilen güvenilirlik üst sınırı hesaplaması yapılmaktadır. Eğer bu şebekenin güvenilirlik üst sınırı $R_{ÜS}$ R_0'a eşit ve R_0'dan büyükse şebekeye güvenilirlik tahmini için Bölüm 2.5.3.'te anlatılan, Yeh ve arkadaşları (1994) tarafından geliştirilmiş Monte Carlo benzetimi kullanılarak gerçekleştirilmektedir. Böylelikle, güvenilirlik üst sınırına göre güvenilirlik kısıtını sağlamayan şebekelerin güvenilirlik tahmini yapılarak algoritmanın çalışma zamanının uzaması önlenmiş olmaktadır.

Güvenilirlik kısıtı altında minimum maliyetli şebekenin tasarımı probleminde kullanılan değişken komşu iniş algoritmasının adımları şu şekildedir:

Başlangıç
N_k komşuluk yapıları kümesini belirle *(k = 1,2,3)*
2-bağlılık kısıtını sağlayan ve $R_{ÜS} \geq R_0$ olan *s* başlangıç çözümünü üret.
$k = 1$.
$k > 3$ oluncaya kadar adım 1-7'yi tekrarla.
Adım 1. N_k komşuluk yapısını kullanarak aday şebekeler üret.

Adım 2. Elde edilen şebekelerin bağlılığını kontrol et. Bağlı olmayan şebekeleri aday şebeke kümesinden çıkar.
Adım 3. Bağlı şebekelerin $R_{ÜS}$ değerlerini hesapla. $R_{ÜS} < R_0$ ise aday şebeke kümesinden çıkar.
Adım 4. Aday şebeke kümesinde yer alan şebekelerin maliyetlerini ($C(x)$) hesapla.
Adım 5. Aday şebekeler içinden $R(s) \geq R_0$ koşulunu sağlayan minimum maliyetli şebekeyi (s') belirle.
Adım 6. $G(s) = s'$ olarak ata.
Adım 7. $C(s') < C(s)$ ise $k = 1$, değilse $k=k+1$
DUR.

7.3.5. Tabu listeli değişken komşu iniş algoritması

Güvenilirlik kısıtı altında minimum maliyetli şebekenin tasarımı probleminin çözümünde Değişken Komşu İniş algoritması temel haliyle uygulandıktan sonra algoritmaya tabu listesi eklenerek çözüm kalitesini iyileştirmek yoluna gidilmiştir. Yapılan bu değişiklikle algoritmaya tabu listeli değişken komşu iniş algoritması (TL_DKİ) ismi verilmiştir.

Tabu listesinin amacı, algoritmanın tabu listesi uzunluğu kadarki son iterasyolarında yapılmış hareketleri yasaklayarak önceki çözümlere dönmeyi engellemektir. TL_DKİ algoritmasında tabu listeleri giren ve çıkan hatlar için ayrı ayrı tutulmuştur. Tabu listelerinin uzunluğu tasarımıyla ilgilenilen şebekenin düğüm sayısıyla sınırlı tutulmuştur.

Geliştirilen tabu listeli değişken komşu iniş algoritmasının adımları aşağıdaki gibidir:

Başlangıç
N_k komşuluk yapıları kümesini belirle *(k = 1,2,3)*
2-bağlılık kısıtını sağlayan ve $R_{ÜS} \geq R_0$ olan s başlangıç çözümünü üret.
k = 1 olarak ayarla.
k > 3 oluncaya kadar aşağıdaki Adım 1-7'yi tekrarla.

Adım 1. N_k komşuluk yapısını kullanarak aday şebekeler üret.
Adım 2. Elde edilen şebekelerin bağlılığını kontrol et. Bağlı olmayan şebekeleri aday şebeke kümesinden çıkar.
Adım 3. Bağlı şebekelerin $R_{ÜS}$ değerlerini hesapla. $R_{ÜS} < R_0$ ise aday şebeke kümesinden çıkar.
Adım 4. Tabu olan hareketleri aday şebeke kümesinden çıkar.
Adım 5. Aday şebeke kümesinde yer alan şebekelerin maliyetlerini ($C(x)$) hesapla.
Adım 6. Aday şebekeler içinden $R(s) \geq R_0$ koşulunu sağlayan minimum maliyetli şebekeyi belirle.
Adım 7. $G(s) = s'$ olarak ata.
Adım 8. $C(s') < C(s)$ ise $k = 1$, değilse $k = k+1$
Adım 9. Tabu listesini güncelleştir. DUR.

7.4. Kuş Sürüsü Eniyilemesi ile Haberleşme Şebekelerinin Tasarımı

Kuş Sürüsü Eniyilemesi (KSE), sürü zekası (swarm intelligence) olarak adlandırılan sınıfta yer alan bir algoritmadır. Bölüm 5.6'da da belirtildiği gibi KSE'nin sürekli ve kesikli olmak üzere iki türü bulunmaktadır. Bu tezde dikkate alınan şebekelerin topolojik tasarımı probleminde kesikli KSE kullanılmıştır. Ayrıca, global en iyi KSE yaklaşımı kullanılmıştır.

7.4.1. Hız güncelleme ve kuşların pozisyonlarının belirlenmesi

j boyutundaki bir i kuşunun $t+1$ anındaki hızı $v_{ij}(t)$ aşağıdaki eşitlik kullanılarak hesaplanmaktadır.

$$v_{ij}(t+1) = w \cdot v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}[\hat{y}_j(t) - x_{ij}(t)] \qquad (7.1)$$

Bu eşitlikte c_1 bilişsel bileşen, c_2 sosyal bileşen, r_1 ve r_2 [0,1] aralığında rassal sayılardır. $y_{ij}(t)$ kuşun kişisel en iyi çözümünü, $\hat{y}_j(t)$ ise global en iyi çözümü ifade etmektedir. $v_{ij}(t)$ değeri kuşun t anındaki hızını, w ise eylemsizlik ağırlığı (inertia

weight)dır. Kuşların arama uzayının sınırlarının dışına çıkmalarını önlemek amacıyla simetrik olacak şekilde hız değerlerine alt ve üst sınırlar getirilmiştir [$-V_{max}$, V_{max}]. Hız güncelleme sırasında kuşun hızı V_{max} değerini aşarsa kuşun hızı V_{max} olarak, $-V_{max}$ değerinin altına düşerse $-V_{max}$ olarak alınmaktadır.

Kuşların konumlarının belirlenmesi ise hız değerlerinden yararlanarak yapılmaktadır. (i,j) hattının varlığının belirlenmesinde kullanılan olasılık sınırı belirlemede bir sigmoid fonksiyonundan yararlanılarak elde edilmiştir:

$$s(v_{ij}(t)) = \frac{1}{1+\exp(-v_{ij}(t)} \tag{7.2}$$

Bu eşitlikte $s(v_{ij}(t))$, $x_{ij}(t)$'nin 1 değeri alma olasılığını göstermektedir. $x_{ij}(t)$'nin 0 veya 1 değerini almasına ise Eş. 7.3 ile karar verilmektedir.

$$x_{ij}(t) = \begin{cases} 1, & if \quad U(0,1) < s(v_{ij}(t)) \\ 0, & d.d. \end{cases} \tag{7.3}$$

7.4.2. Başlangıç hızlarının belirlenmesi ve başlangıç sürüsünün oluşturulması

Çözüm uzayında arama gerçekleştirecek olan kuşların başlangıç hızları aşağıdaki eşitlik kullanılarak elde edilmektedir. Bu eşitlikte r [0,1] aralığında bir rassal sayıdır.

$$v_{ik}(0) = -V_{\max} + (V_{\max} - (-V_{\max})) \cdot r \tag{7.4}$$

Başlangıç hızlarıyla birlikte başlangıç sürüsünün de oluşturulması gerekmektedir. Başlangıç çözümünün oluşturulmasında düğümlerin kümesi $N_r = \{1,2,...,N\}$ ve oluşturulan hatların kümesi ise L_r = { }'dir. Şekil 7.6'da başlangıç sürüsünün oluşturulmasına ilişkin adımlar verilmektedir.

Aşağıdaki adımları toplam kuş sayısı (sürü büyüklüğü) kadar tekrarla

Başlangıç: $N_r = \{1,2,....,N\}$ ve $L_r = \{\ \}$
Adım 1. N_r'den rassal olarak 2 adet düğüm seç ve elde edilen hattı L_r'ye kaydet
$N_r = \{\ \}$ oluncaya kadar aşağıdaki adımları tekrarla
Adım 2. N_r'den rassal olarak bir düğüm seç.
Adım 3. Seçilen düğümün Lr'de yer alan düğümlerden birine rassal olarak ekle.
Adım 4. N_r ve L_r'yi güncelle.

Şebeke oluşturulurken N_r'den seçilen düğüm, önceki iterasyonlarda seçilmiş olan ve *Lr*'de tutulan düğümlerden rassal olarak seçilen bir düğüme eklenmektedir. Böylelikle elde edilen şebeke bir yayılan ağaç olmaktadır. Yayılan ağaç elde edildikten sonra şebekeye 2-bağlılık tamir algoritması uygulanmaktadır. Bu prosedür şekil 7.3'te belirtilen adımlar izlenerek yerine getirilmektedir.
Bu tezde, KSE algoritmasında çeşitli durumlar incelenerek yeni algoritmalar önerilmiştir. Önerilen 10 farklı algoritma çözüm kalitesi ve çözüm zamanı açısından değerlendirilmiştir. Aşağıda önerilen algoritmalar verilmektedir.

1. Temel KSE Algoritması
2. Geliştirilen KSE_D Algoritması
3. Geliştirilen KSE_TB_1 Algoritması
4. Geliştirilen KSE_D_TB_1 Algoritması
5. Geliştirilen KSE_TB_{DBS}_1 Algoritması
6. Geliştirilen KSE_D_TB_{DBS}_1 Algoritması
7. Geliştirilen KSE_TB_2 Algoritması
8. Geliştirilen KSE_D_TB_2 Algoritması
9. Geliştirilen KSE_TB_{DBS}_2 Algoritması
10. Geliştirilen KSE_D_TB_{DBS}_2 Algoritması

7.4.3. Temel KSE algoritması

Temel KSE algoritmasının (KSE) adımları şekil 7.7'de verilmiştir. *x* aday şebekeyi, *C(x)* aday şebekenin maliyetini, $R_{ÜS}$,şebekelerin güvenilirlik üst sınırı değerini, *R(x)*

şebekenin güvenilirlik değerini, p_b kuşların kişisel en iyi değerlerini, g_b global en iyi değeri ifade etmektedir.

Başlangıç
Başlangıç sürüsünü oluştur.
Global en iyi çözümde ard arda 50 defa değişim olmayıncaya kadar tekrarla
Adım 1. Aday şebekelerin bağlılıklarını kontrol et. Bağlı olmayan şebekeleri sürüden çıkar.
Adım 2. Aday şebekelerin 2-bağlılıklarını kontrol et.
Adım 2.1. Aday şebeke 2-bağlı değilse 2-bağlılık tamir algoritmasını uygula
Adım 3. Aday şebekelerin güvenilirlik üst sınırlarını ($R_{ÜS}$) belirle.
Adım 4. Aday şebeke kümesinde yer alan şebekelerin maliyetlerini (C(x)) hesapla, minimum maliyetli şebekeyi belirle.
Adım 5. Kuşların p_b değerlerini güncelle.
Adım 6. Eğer algoritma 1. iterasyonda ise adım 7'ye değilse Adım 8'e git.
Adım 7. Minimum maliyetli şebekenin güvenilirliğini (R(x)) Monte Carlo benzetimi ile hesapla.
Adım 7.1. $R(x) > R_0$ ise $g_b = x$ ve tüm kuşlar için $p_b = x$ olarak ata.
Adım 8. $R(x) > R_0$ ise $g_b = x$ olarak ata.
Adım 9. Kuşların hız değerlerini güncelle.
Adım 10. Sigmoid fonksiyonunu kullanarak kuşların xij değerlerini belirle. Adım1'e git.

7.4.4. Geliştirilen KSE_D algoritması

İncelenecek olan bu ikinci durumda eylemsizlik ağırlığı (w) değeri dinamik olarak değişmektedir. Diğer bir ifadeyle algoritmanın her iterasyonunda farklı bir w değeri kullanılacaktır. KSE'da w değerine dinamiklik kazandırmada kullanılan çeşitli yaklaşımlar bulunmaktadır. Bu yaklaşımlar aşağıda sıralanmıştır [72,73]:

Rassal ayarlama; her iterasyonda farklı bir w değerinin rassal olarak seçilmesidir.

Doğrusal azalma; başlangıçtaki büyük bir *w* değerinin algoritmanın her iterasyonunda doğrusal olarak azaltılmasıdır.

Doğrusal olmayan azalma; başlangıçtaki büyük bir *w* değerinin algoritmanın her iterasyonunda doğrusal olmayan şekilde azaltılmasıdır.
Bulanık uyumlu eylemsizlik (fuzzy adaptive inertia); w değerinin bulanık küme kuralları kullanılarak dinamik biçimde değiştirlimesidir.

Doğrusal artırma; başlangıçtaki küçük bir *w* değerinin algoritmanın her iterasyonunda doğrusal olarak artırılmasıdır.

Bu çalışmada ise *w* değeri için rassal ayarlama kullanılmıştır. Her iterasyonda *w* değeri U (0,1) aralığında rassal bir sayı olmak üzere aşağıdaki Eş. 7.5 kullanılarak belirlenmiştir.

$$w = (1 + U(0,1)) \tag{7.5}$$

7.4.5. Geliştirilen KSE_TB_1 algoritması

KSE, bu kısımda incelenecek türünde stokastik arama yöntemlerinden biri olan tavlama benzetimi algoritması ile karma olarak kullanılmıştır. Geliştirilen bu algoritmada tavlama benzetimi algoritması her iterasyonda sürüdeki kuşlara %5 olasılıkla uygulanmıştır. Diğer bir ifadeyle, kuşların %5'lik bir kısmı eğer iyileşme meydana gelmişse tavlama benzetimi algoritmasıyla elde edilen sonuçla yer değiştirmektedir.

Tavlama Benzetimi, Kirkpatrick ve arkadaşları (1983) tarafından geliştirilmiş ve bir çok eniyileme problemine başarıyla uygulanmış bir algoritmadır. Tavlama Benzetimi, katıların tavlama işleminden esinlenerek geliştirilmiş stokastik bir arama tekniğidir ve bir olasılık fonksiyonu kullanarak yerel en iyi değerden kurtulmaya olanak sağlar.

Tavlama Benzetimi, *i* başlangıç çözümüyle başlayarak başlangıç çözümünün komşularından türetilmiş *j* çözümünü üretir. Amaç fonksiyonundaki değişme $\Delta = f(j) - f(i)$ ifadesiyle hesaplanır. Eğer $\Delta \leq 0$ ise o zaman yeni çözüm kabul edilir. $\Delta > 0$ olduğunda ise çözüm belli bir olasılıkla ,genellikle $e^{-\Delta /T}$ olasılığıyla, kabul edilir. Buradaki *T* değeri tavlama olayındaki sıcaklığa benzer bir kontrol parametresidir. Genellikle *T* değeri algoritma boyunca monoton biçimde düşürülür ve buna *soğuma planı* adı verilir. Bahsedilen işlemler önceden tanımlanan durdurma koşulu sağlanana kadar tekrarlanır. Tavlama benzetimi algoritmasının genel adımları aşağıda verilmiştir.

Adım 1. Bir başlangıç durumu seç: $i \in S$
Bir başlangıç sıcaklığı seç: T > 0
Sıcaklık değişim sayacını sıfırlar: t = 0
Adım 2. Durdurma koşulu sağlanmışsa DUR, değilse tekrar sayacını sıfırla: n = 0 ve devam et.
Adım 3. i'nin bir komşusu olan j durumunu rassal olarak üret.
$\Delta = f(j) - f(i)$
Eğer $\Delta < 0$ ise i = j, değilse ve U(0, 1) < exp(-Δ/T) ise i = j.
Adım 4. n = n + 1. Eğer n < M ise adım 3'e git, değilse t = t + 1,
T= T(t) ve adım 2'ye git.

Bu çalışmada kullanılan tavlama benzetimi algoritmasında ise aday şebekeler şu şekilde elde edilmiştir: Şebekeden rassal olarak 4 adet düğüm seçilir (n_1, n_2, n_3, n_4). Bu düğümlerin oluşturduğu iki adet hat (n_1, n_2) ve (n_3, n_4) dikkate alınır. Eğer şebekede her iki hat da yer almıyorsa hatlardan biri rassal olarak seçilerek şebekeye eklenir ve yeni bir çözüm elde edilir. Eğer hatların her ikisi de şebekede yer alıyorsa yine hatlardan biri rassal olarak seçilir ve şebekeden çıkarılır. Eğer hatlardan yalnız biri şebekede yer alıyorsa bu durumda şebekede bulunan hat çıkarılır, şebekede bulunmayan hat eklenerek yeni çözüm elde edilir. Eğer bu hareketler geçersiz bir çözüm üretirse bu şebeke çıkarılır ve yeni bir çözüm üretilir. Aşağıda geliştirilen KSE_TB_1 Algoritması'nın adımları yer almaktadır.

Başlangıç

Başlangıç sürüsünü oluştur.

Global en iyi çözümde ard arda 50 defa değişim olmayıncaya kadar tekrarla

Adım 1. Aday şebekelerin bağlılıklarını kontrol et. Bağlı olmayan şebekeleri sürüden çıkar.

Adım 2. Aday şebekelerin 2-bağlılıklarını kontrol et.

Adım 2.1. Aday şebeke 2-bağlı değilse 2-bağlılık tamir algoritmasını uygula

Adım 3. Aday şebekelerin güvenilirlik üst sınırlarını (RÜS) belirle.

Adım 4. $R_{ÜS} < R_0$ olan aday şebekeleri sürüden çıkar.

Adım 5. Aday şebeke kümesinde yer alan şebekelerin maliyetlerini (C(x)) hesapla, şebekeleri maliyetlerine göre artan şekilde sırala.

Adım 6. Her bir aday şebeke için $u \sim U(0,1)$ üret. Eğer $u \leq 0.05$ ise tavlama benzetimi algoritması kullan.

//Tavlama Benzetimi Algoritması//

Adım 6.1. Başlangıç çözümü S = gb

Başlangıç sıcaklığı T(0) = 475

Durdurma koşulu sağlanmışsa DUR, değilse devam et.

Adım 6.2. S'nin komşularından aday çözümler üret

Adım 6.3. Elde edilen aday şebekelerin 2-bağlı değilse 2-bağlılık tamir algoritması uygula.

Adım 6.4. Aday şebekelerin bağlılıklarını kontrol et, bağlı olmayan şebekeleri çıkar

Adım 6.5. Aday şebekelerin $R_{ÜS}$ değerlerini kontrol et, $R_{ÜS} < R_0$ olan şebekeleri çıkar.

Adım 6.6. Aday şebekelerin maliyetlerini hesapla.

Adım 6.7. $R_{ÜS} \geq R_0$ koşulunu sağlayan minimum maliyetli şebekenin güvenilirliğini Monte Carlo benzetimi ile hesapla.

(a) $R(x) \geq R_0$ ise *S'* çözümü olarak ata.

Adım 6.8. $f(S') > f(S)$ ise $\Delta = [(f(S') - s(S))/f(S)]*100$ değerini elde et.

(a) $\Delta < 0$ ise $S = S'$ olarak ata.

(b) $\Delta > 0$ ve $e^{-\Delta/T} > U(0,1)$ ise S = S'

Adım 6.9. $T(t) = \dfrac{T(t-1)}{1+\gamma\sqrt{T(t-1)}}$ değerini hesapla.

Adım 6.10. T(t) $\leq$ 0.15 veya en iyi çözümde ard arda 15 defa iyileşme meydana gelmemişse DUR.

// Tavlama Benzetimi algoritmasının sonu //

Adım 7. Kuşların p_b değerlerini güncelle.

Adım 8. Eğer algoritma 1. iterasyonda ise adım 9'a değilse Adım 10'a git.

Adım 9. Minimum maliyetli şebekenin güvenilirliğini (*R(x)*) Monte Carlo benzetimi ile hesapla.

Adım 9.1. $R(x) \geq R_0$ ise $g_b = x$ ve tüm kuşlar için $p_b = x$ olarak ata.

Adım 10. $R(x) \geq R_0$ ise $g_b = x$ olarak ata.

Adım 11. g_b'yi güncelle

Adım 12. Kuşların hız değerlerini güncelle.

Adım 13. Sigmoid fonksiyonunu kullanarak kuşların x_{ij} değerlerini belirle. Adım1'e git.

Sıcaklık güncelleme ifadesi $T(t) = \dfrac{T(t-1)}{1+\gamma\sqrt{T(t-1)}}$ 'de γ değeri 0.05 olarak alınmıştır.

7.4.6 Geliştirilen KSE_D_TB_1 algoritması

Önerilen bu algoritmada, Bölüm 7.2.3'te karma şekilde uygulanan tavlama benzetimi algoritmasının yanında bölüm 7.2.2'deki şekilde KSE'nun her iterasyonunda *w* değerinin rassal biçimde değiştirildiği durum söz konusudur.

7.4.7. Geliştirilen KSE_TB_{DBS}_1 algoritması

Bu algoritmada, Bölüm 7.2.3'te karma şekilde uygulanan tavlama benzetimi algoritmasının yanında Bölüm 7.2.2'deki şekilde tavlama algoritmasının kullanıldığı her seferde başlangıç sıcaklığının %10 oranında düşürüldüğü durum birlikte dikkate alınmıştır.

7.4.8. Geliştirilen KSE_D_TB_{DBS}_1 algoritması

Bu algoritmada, Bölüm 7.2.3'teki şekilde tavlama algoritmasının kullanıldığı her seferde başlangıç sıcaklığının %10 oranında düşürüldüğü ve Bölüm 7.2.2'de bahsedilen *w* değerinin her iterasyonda rassal olarak değiştirildiği durum birlikte dikkate alınmıştır.

7.4.9. Geliştirilen KSE_TB_2 algoritması

KSE'da elde edilen global en iyi değerde bir iyileşme meydana geldiğinde bu global en iyi değer tavlama benzetimi algoritmasının başlangıç çözümü olmakta ve tavlama benzetimi algoritmasıyla yeni bir çözüm elde edilmektedir. Elde edilen çözüm global en iyi değerden daha iyi bir çözümse yeni global en iyi değer bu çözüm olmaktadır.

7.4.10. Geliştirilen KSE_D_TB_2 algoritması

Bu algoritmada, Bölüm 7.2.2'de anlatılan *w* değerinin dinamik olarak kullanımı ile Bölüm 7.2.7'de anlatılan global en iyi değerde iyileşme meydana geldiğinde tavlama benzetimi algoritmasının uygulanması durumları bir arada kullanılmıştır.

7.4.11. Geliştirilen KSE_TB_{DBS}_2 algoritması

Bu algoritmada, kullanılan global en iyi değerde iyileşme sağlandığında kullanılan tavlama benzetiminin başlangıç sıcaklığı, tavlama benzetimi algoritmasının başlatıldığı her seferde bir önceki değerinin %10'u kadar azaltılmaktadır.

7.4.12. Geliştirilen KSE_D_TB_{DBS}_2 algoritması

Bu algoritmada, KSE'de elde edilen global en iyi değere tavlama benzetimi algoritmasının uygulandığı her seferde başlangıç sıcaklığı bir öncekine göre %10 oranında düşürülmektedir. Ayrıca algoritmanın *w* değeri de her iterasyonda rassal olarak belirlenmektedir.

7.5. Karınca Kolonisi Eniyilemesi İle Haberleşme Şebekelerinin Tasarımı

Karınca Kolonisi Eniyilemesi (KKE) algoritması da sürü zekası (swarm intelligence) sınıfına giren bir algoritmadır. Çalışmanın bu bölümünde öncelikle KKE algoritmasının şebeke tasarımı probleminde nasıl kullanıldığına ilişkin bilgiler verilecektir.

7.5.1. Başlangıç feromenlerinin elde edilmesi

(*i,j*) hattının başlangıç feromeninin (τ_{ij}^0) elde edilmesinde bir başlangıç çözümü kullanılmıştır. τ_{ij}^0 değerleri, G_x^s başlangıç çözümünü ifade etmek üzere (1/f(G_x^s)) ifadesiyle elde edilmektedir.

7.5.2. Çözümlerin oluşturulması

Tüm hatlara başlangıç feromen değerleri atandıktan sonra başlangıç çözümü olarak elde edilen şebeke aynı zamanda global en iyi çözüm de olmaktadır. Karıncalar çözüm oluşturmaya başlamadan önce her düğüme bir adet karınca yerleştirilmektedir. Bu, her problemde kullanılan karınca sayısının düğüm sayısına eşit olduğu anlamına da gelmektedir. Karıncalar bulundukları düğümlerden hareket ederken feromen bilgisi τ_{ij} ve sezgisel bilgi η_{ij} kullanılmaktadır. η_{ij} sezgisel bilgi değeri (*i,j*) hattının bağlanma maliyetinin tersi ($\eta_{ij} = 1/c_{ij}$) alınarak hesaplanmaktadır. Feromen bilgisi ise hatları önceden kullanan ve iyi çözümler elde eden karıncaların bıraktıkları feromenler yardımıyla elde edilmektedir. *i* düğümünde yer alan karıncanın bir sonraki *j* düğümünü seçme olasılığı aşağıdaki geçiş kuralı kullanılarak gerçekleştirilmektedir:

$$j = \begin{cases} \arg\max_{J} (\tau_{ij})^{\alpha} \cdot (\eta_{ij})^{\beta}, & q \le q_0 \\ J, & q > q_0 \end{cases} \tag{7.6}$$

J ise aşağıdaki geçiş kuralı kullanılarak seçilmektedir:

$$p_{ij} = \begin{cases} \dfrac{(\tau_{ij})^{\alpha} \cdot (\eta_{ij})^{\beta}}{\sum_{j \in N / N_r} (\tau_{ij})^{\alpha} \cdot (\eta_{ij})^{\beta}}, & j \in N / N_r \\ 0, & d.d. \end{cases} \tag{7.7}$$

Bu ifadede yer alan α ve β değerleri karıncalar tarafından kullanılacak olan feromen bilgisi ve sezgisel bilgi arasındaki göreceli etkiyi belirleyen değerlerdir. q ise 0 ile 1 arasında üretilen rassal bir değerdir. q_0 değeri, araştırma ve yoğunlaşma stratejileri arasındaki göreceli önemi belirlemekte kullanılan bir parametredir ve algoritmanın başlangıcında belirlenir.

Böylelikle hareket edilecek olan *j* düğümüne karar verildikten sonra gelinen yeni düğüm başlangıç düğümü olmakta ve yukarıda anlatılan karar verme süreçleri tekrarlanmaktadır. Bu işlemler, şebekede hareket edilecek düğümler kalmayıncaya kadar devam etmektedir.

7.5.3. Feromen güncelleme

Bu çalışmada kullanılan KKE algoritmasında feromen güncelleme işlemi *çevrim içi feromen güncelleme* (*online pheromen update*) ve *çevrim dışı feromen güncelleme* (*offline pheromen update*) yöntemlerinin her ikisi de kullanılarak gerçekleştirilmiştir. Çevrim içi feromen güncelleme, her bir şebeke oluşturulduktan sonra yapılan güncellemedir ve amacı topolojide yer alan hatların feromen yoğunluklarını azaltarak bir sonraki topolojilerde yapılacak hat seçimlerinde araştırmayı (exploration) desteklemektir. Çevrim içi feromen güncelleme aşağıdaki gibi yapılır:

$$\tau_{ij}^{yeni} = (1-\rho)\tau_{ij}^{eski} + \rho\tau_{ij}^{0}$$

Bu eşitlikte $\rho \in [0,1]$ feromen buharlaştırma oranıdır. Bu çalışmada 0,1 olarak alınmıştır.

Bir kolonideki tüm şebekeler oluşturulduktan sonra aşağıda yer alan çevrimdışı feromen güncelleştirme kuralı uygulanır. Fakat daha önce tüm hatlara ait feromen değerlerinde buharlaştırma yapılır.

$$\tau_{ij} = (1-\rho)\tau_{ij} \tag{7.8}$$

Hatlardaki feromen miktarları iterasyonun en iyi değeri ve global en iyi değer göz önüne alınarak artırılır.

$$\tau_{ij}^{yeni} = \begin{cases} \tau_{ij}^{eski} + \left(\dfrac{1}{f(G_x^b)}\right) & , (i,j) \in L_x^b \\ \tau_{ij}^{eski} + \left(\dfrac{f(G_x^b)}{f(G_x^i)}\right)\left(\dfrac{1}{f(G_x^i)}\right), & (i,j) \in L_x^i \,/\, L_x^b \end{cases} \tag{7.9}$$

Global en iyi çözümde (G_x^b) yer alan her bir hatta bulunan feromenlerdeki artış miktarı f(G_x^b) değerinin tersine eşittir. Yerel en iyi çözümde (G_x^i) yer alan hatlardaki feromen artışları ise bu değerin global en iyi değere olan yakınlığına göre karar verilmektedir. Eğer yerel en iyi değer global en iyi değere eşitse bu durumda feromen artışına yapılan katkı global en iyi değerinki kadar olmaktadır.

7.5.4. Durdurma koşulu

KKE algoritması uygulanırken durdurma koşulu olarak toplam iterasyon sayısı belirlenmiştir ve bu değer her problem türünde tam bağlı şebekenin toplam hat sayısının iki katı olarak alınmıştır.

KKE algoritması ile şebeke tasarımında da 10 farklı algoritma önerilmiştir. Bunlar:

1. Geliştirilen TB_KKE Algoritması
2. Geliştirilen TB_KKE$_D$ Algoritması
3. Geliştirilen KKE_TB Algoritması

4. Geliştirilen KKE_D_TB Algoritması
5. Geliştirilen KKE_TB_{DBS} Algoritması
6. Geliştirilen KKE_D_TB_{DBS} Algoritması
7. Geliştirilen TB_KKE_TB Algoritması
8. Geliştirilen TB_KKE_D_TB Algoritması
9. Geliştirilen TB_KKE_TB_{DBS} Algoritması
10. Geliştirilen TB_KKE_D_TB_{DBS} Algoritması

7.5.5. Geliştirilen TB_KKE algoritması

İncelenecek olan bu durumda algoritma tavlama benzetimi algoritması ile başlamakta ve güvenilirlik kısıtını sağlayan geçerli çözüm KKE algoritmasının başlangıç çözümü olmaktadır. Kullanılan tavlama benzetimi algoritmasının işleyişi Bölüm 7.2'de anlatılan KSE algoritmasında kullanıldığı biçimiyle kullanılmaktadır. Tavlama benzetimi için başlangıç çözümü elde edilişi aşağıda verilmiştir. Başlangıç çözümünün oluşturulmasında düğümlerin kümesi $N_r = \{1,2,...,N\}$ ve oluşturulan hatların kümesi ise $L_r = \{\ \}$'dir.

Başlangıç: $N_r = \{1,2,....,N\}$ ve $L_r = \{\ \}$
Adım 1. N_r'den rassal olarak 2 adet düğüm seç ve elde edilen hattı L_r'ye kaydet
$N_r = \{\ \}$ oluncaya kadar aşağıdaki adımları tekrarla
Adım 2. N_r'den rassal olarak bir düğüm seç.
Adım 3. Seçilen düğümü Lr'de yer alan düğümlerden en son eklenen düğümle birleştir.
Adım 4. N_r ve L_r'yi güncelle.

Bu şekilde elde edilen şebeke bir yayılan ağaç olmaktadır. Bu nedenle şebekeye 2-bağlılık tamir algoritması uygulanmakta ve bu şebeke için $R_{ÜS} > R_0$ koşulu sağlanıncaya kadar hat eklenmeye devam edilmektedir. Geliştirilen TB_KKE algoritmasının adımları aşağıdaki gibidir.

Adım 1. S başlangıç çözümünü üret.

S çözümüne 2-bağlılık tamir algoritması uygula

S'nin $R_{ÜS}$ değerini kontrol et.

$R_{ÜS} < R_0$ ise $R_{ÜS} \geq R_0$ oluncaya kadar minimum maliyetli hat eklemeye devam et.

Başlangıç sıcaklığı T(0) = 475

Durdurma koşulu sağlanmışsa DUR, değilse devam et.

Adım 2. *S*'nin komşularından aday çözümler üret

Adım 3. Elde edilen aday şebekelerin 2-bağlı değilse 2-bağlılık tamir algoritması uygula.

Adım 4. Aday şebekelerin bağlılıklarını kontrol et, bağlı olmayan şebekeleri çıkar.

Adım 5. Aday şebekelerin $R_{ÜS}$ değerlerini kontrol et, $R_{ÜS} \geq R_0$ oluncaya kadar hat ekle.

Adım 6. Aday şebekelerin maliyetlerini hesapla.

Adım 7. Minimum maliyetli şebekenin güvenilirliğini Monte Carlo benzetimi ile hesapla.

Adım 7.1. $R(x) \geq R_0$ ise *S* çözümü olarak ata.

Adım 8. $f(S') > f(S)$ ise $\Delta = [(f(S') - f(S))/f(S)]*100$ değerini elde et.

Adım 8.1. $\Delta < 0$ ise $S = S'$ olarak ata.

Adım 8.2. $\Delta > 0$ ve $e^{-\Delta/T} > U(0,1)$ ise S = S'

Adım 9. $T(t) = \dfrac{T(t-1)}{1+\gamma\sqrt{T(t-1)}}$

Adım 10. T(t) $\leq$ 0.15 veya en iyi çözümde ard arda 15 defa iyileşme meydana gelmemişse DUR

//Karınca Kolonisi Eniylemesi//

Adım 11. τ_{ij}^{0} değerlerini elde et.

q_0 değerini ayarla.

Durdurma koşulu sağlanıncaya kadar aşağıdaki adımları tekrarla

Adım 12. Aday çözümler üret.

Adım 13. Aday şebekelerin 2-bağlılıklarını kontrol et.

Adım 13.1. Aday şebeke 2-bağlı değilse 2-bağlılık tamir algoritmasını uygula

Adım 14. Aday şebekelerin güvenilirlik üst sınırlarını ($R_{ÜS}$) belirle.

Adım 15. $R_{ÜS} < R_0$ olan aday şebekeleri sürüden çıkar.

Adım 16. Aday şebeke kümesinde yer alan şebekelerden güvenilirlik kısıtını sağlayan minimum maliyetli şebekeyi seç.

Adım 17. Elde edilen şebekeyi G_x^l olarak ata. Eğer şebekenin maliyeti G_x^b'in maliyetinden düşükse bu şebekeyi G_x^l olarak ata.

Adım 18. Hatların feromen miktarlarını güncelle

7.5.6. Geliştirilen TB_KKE$_D$ algoritması

Bu algoritmada, tavlama benzetimi algoritması ile elde edilen topoloji KKE için başlangıç çözümü olarak kullanılmaktadır ve KKE'nin her iterasyonunda düşük bir değerle başlanan q_0 değeri artırılmaktadır. Böylelikle q_0 değerine dinamiklik kazandırılmaktadır. Her iterasyonda yapılan q_0 değerinin güncellenmesi eşitlik 7.1'deki gibi gerçekleştirilmektedir.

$$q_0^{yeni} = q_0 \frac{\text{iterasyon no}}{\text{toplam iterasyon sayısı}} \tag{7.10}$$

7.5.7. Geliştirilen KKE_TB algoritması

Bu algoritmada, karınca kolonisi eniyilemesi algoritmasının global en iyi değerinde iyileşme meydana geldiğinde elde edilen bu çözüme tavlama benzetimi algoritması uygulanmaktadır.

Başlangıç çözümü Bölüm 7.3.1'de tavlama benzetimi algoritmasında olduğu gibi elde edilmektedir.

7.5.8. Geliştirilen KKE$_D$_TB algoritması

Bu algoritmada, Bölüm 7.3.3'te olduğu gibi global en iyi değerde iyileşme meydana geldiğinde bu şebekeye tavlama benzetimi algoritması uygulanmaktadır. Ayrıca

Bölüm 7.3.2'de olduğu gibi KKE'de küçük bir q_0 değeri ile başlanmakta ve bu değer her iterasyonda azaltılmaktadır.

7.5.9. Geliştirilen KKE_TB$_{DBS}$ algoritması

Bu algoritmada, Bölüm 7.3.3'te olduğu gibi global en iyi değerde iyileşme meydana geldiğinde bu şebekeye tavlama benzetimi algoritması uygulanmaktadır. Bunun yanında tavlama benzetimi algoritmasının başlatıldığı her seferde T(0) başlangıç sıcaklığı %10 oranında azaltılmaktadır.

7.5.10. Geliştirilen KKE$_D$_TB algoritması

Bu algoritmada, dinamik q_0 değeri kullanılmakta ve global en iyi değerde iyileşme meydana geldiğinde tavlama benzetimi algoritması uygulanmaktadır. Ayrıca tavlama benzetiminin başlangıç sıcaklığı tavlama benzetiminin her başlatılışında %10 oranında azaltılmaktadır.

7.5.11. Geliştirilen TB_KKE_TB algoritması

Bu algoritmada, öncelikle tavlama benzetimi algoritması uygulanarak problemin kısıtlarını sağlayan bir çözüm elde edilmektedir. Daha sonra bu çözüm KKE algoritması için başlangıç çözümü olmakta ve KKE algoritması uygulanırken global en iyi değerde bir iyileşme meydana geldiğinde yeniden tavlama benzetimi algoritması uygulanmaktadır. TB_KKE_TB algoritmasının adımları aşağıdaki gibidir.

Adım 1. S başlangıç çözümünü üret.
S çözümüne 2-bağlılık tamir algoritması uygula
S'nin $R_{ÜS}$ değerini kontrol et.
$R_{ÜS} < R_0$ ise $R_{ÜS} \geq R_0$ oluncaya kadar minimum maliyetli hat eklemeye devam et.
Başlangıç sıcaklığı T(0) = 475
Durdurma koşulu sağlanmışsa DUR, değilse devam et.

Adım 2. *S*'nin komşularından aday çözümler üret

Adım 3. Elde edilen aday şebekelerin 2-bağlı değilse 2-bağlılık tamir algoritması uygula.

Adım 4. Aday şebekelerin bağlılıklarını kontrol et, bağlı olmayan şebekeleri çıkar.

Adım 5. Aday şebekelerin $R_{ÜS}$ değerlerini kontrol et, $R_{ÜS} > R_0$ oluncaya kadar hat ekle.

Adım 6. Aday şebekelerin maliyetlerini hesapla.

Adım 7. Minimum maliyetli şebekenin güvenilirliğini Monte Carlo benzetimi ile hesapla.

Adım 7.1. $R(x) \geq R_0$ ise *S'* çözümü olarak ata.

Adım 8. $f(S') > f(S)$ ise $\Delta = [(f(S') - s(S))/f(S)]*100$ değerini elde et.

Adım 8.1. $\Delta < 0$ ise $S = S'$ olarak ata.

Adım 8.2. $\Delta > 0$ ve $e^{-\Delta/T} > U(0,1)$ ise S = S'

Adım 9. $T(t) = \frac{T(t-1)}{1+\gamma\sqrt{T(t-1)}}$

Adım 10. T(t) $\leq$ 0.15 veya en iyi çözümde ard arda 15 defa iyileşme meydana gelmemişse DUR

//Karınca Kolonisi Eniyilemesi//

Adım 11. τ_{ij}^0 değerlerini elde et.

Durdurma koşulu sağlanıncaya kadar aşağıdaki adımları tüm koloni için tekrarla

Adım 12. Aday çözümler üret.

Adım 13. Aday şebekelerin 2-bağlılıklarını kontrol et.

Adım 13.1. Aday şebeke 2-bağlı değilse 2-bağlılık tamir algoritmasını uygula

Adım 14. Aday şebekelerin güvenilirlik üst sınırlarını ($R_{ÜS}$) belirle.

Adım 15. $R_{ÜS} < R_0$ olan aday şebekeleri sürüden çıkar.

Adım 16. Aday şebeke kümesinde yer alan şebekelerden güvenilirlik kısıtını sağlayan minimum maliyetli şebekeyi seç.

Adım 17. Elde edilen şebekeyi G_x^l olarak ata. Eğer şebekenin maliyeti G_x^b'in maliyetinden düşükse adım 10'a git, değilse adım 10'a git.

Adım 18. Elde edilen şebekeyi G_x^b olarak ata, hatların feromen miktarlarını güncelle ve adım 12'ye git.

Adım 19. Hatların feromen miktarlarını güncelle ve adım 4'e git.

Adım 20. Tavlama Benzetimi Algoritması'nı uygula**.**

//Tavlama Benzetimi Algoritması//

Başlangıç sıcaklığı T(0) = 475

Durdurma koşulu sağlanmışsa DUR, değilse devam et.

Adım 21. *S*'nin komşularından aday çözümler üret

Adım 22. Elde edilen aday şebekelerin 2-bağlı değilse 2-bağlılık tamir algoritması uygula.

Adım 23. Aday şebekelerin bağlılıklarını kontrol et, bağlı olmayan şebekeleri çıkar.

Adım 24. Aday şebekelerin $R_{ÜS}$ değerlerini kontrol et, $R_{ÜS} \prec R_0$ olan şebekeleri çıkar.

Adım 25. Aday şebekelerin maliyetlerini hesapla.

Adım 26. Minimum maliyetli şebekenin güvenilirliğini Monte Carlo benzetimi ile hesapla.

Adım 26.1. $R(x) \succeq R_0$ ise *S'* çözümü olarak ata.

Adım 27. $f(S') > f(S)$ ise $\Delta = [(f(S') - s(S))/f(S)]*100$ değerini elde et.

Adım 27.1. $\Delta < 0$ ise $S = S'$ olarak ata.

Adım 27.2. $\Delta > 0$ ve $e^{-\Delta/T} > U(0,1)$ ise $S = S'$

Adım 28. $T(t) = \frac{T(t-1)}{1+\gamma\sqrt{T(t-1)}}$

Adım 29. G_x^b şebekesini güncelle.

Adım 30. T(t) $\leq$ 0.15 veya en iyi çözümde ard arda 15 defa iyileşme meydana gelmemişse DUR

7.5.12. Geliştirilen TB_KKE$_D$_TB Algoritması

Bu algoritmada, KKE dinamik q_0 değerine sahiptir ve global en iyi değerde iyileşme meydana geldiğinde tavlama benzetimi algoritması uygulanmaktadır.

7.5.13. Geliştirilen TB_KKE_TB_{DBS} algoritması

Bu algoritmada, Bölüm 7.5.11'de adımları verilen algoritmada global en iyi değerde iyileşme elde edildiğinde başlatılan tavlama benzetimi algoritmasında her başlangıç yapıldığında T(0) sıcaklığı %10 oranında azaltılmaktadır.

7.5.14. Geliştirilen TB_KKE_D_TB_{DBS} algoritması

Bu algoritmada, KKE dinamik q_0 değerine sahiptir ve global en iyi değerde iyileşme meydana geldiğinde tavlama benzetimi algoritması uygulanmaktadır. Tavlama benzetimi algoritmasının her uygulanışında başlangıç yapıldığında T(0) sıcaklığı %10 oranında azaltılmaktadır.

8. DENEY TASARIMI VE SONUÇLARIN DEĞERLENDİRİLMESİ

Bu çalışmada güvenilirlik kısıtı altında minimum maliyetli şebekenin topolojik tasarımı probleminin çözümü için DKİ, KSE ve KKE algoritmaları kullanılmıştır. Bu algoritmalar kullanılırken algoritmalara ilişkin parametre setlerinin belirlenmesi gerekmektedir. DKİ algoritmasının yapısı nedeniyle belirlenmesi gereken parametreler bulunmadığından yalnızca KSE ve KKE algoritmalarının parametreleri için deney tasarımı gerçekleştirilmiştir.

8.1. Etkinlik Ölçüleri

Bu bölümde, tezde kullanılan algoritmaların etkinliklerini değerlendirmek amacıyla kullanılacak olan etkinlik ölçüleri ve bu ölçülere ilişkin açıklamalara yer verilecektir.

Çözümlerin En İyi Değerden Sapma Oranı: Bu ölçüt en iyi değerleri bilinen test problemlerinde kullanılan sezgisel algoritmaların elde ettikleri değerlerin ne ölçüde en iyi değerden saptığını belirleyen bir ölçüttür. En iyi değerden sapma oranı Eş. 8.1'deki gibi hesaplanmaktadır.

$$\mathrm{ES} = \frac{\mathrm{SÇ} - \mathrm{END}}{\mathrm{END}} x100 \tag{8.1}$$

SÇ: Sezgisel algoritma ile elde edilen çözüm
END: Problemin en iyi çözümü
ES: En iyi değerden sapma oranı (%)

Değişim Katsayısı: Değişim katsayısı problemler için gerçekleştirilen tekrarlarla elde edilen veri setinde verilerin ortalaması olarak beklenmeyen değişkenliği ölçer. Eş. 8.2'de değişim katsayısının hesaplanışı yer almaktadır.

$$\mathrm{DK} = \frac{\mathrm{SS}}{\mathrm{ORT}} \tag{8.2}$$

DK: Değişim katsayısı

SS: Standart sapma

ORT: Ortalama

Çözüm Zamanı: Geliştirilen algoritmaların problemi çözmek için harcadığı zamandır. Bu etkinlik ölçüsünün birimi saniyedir.

8.2. Test Problemleri

Bu çalışmada algoritmaları test etmek amacıyla kullanılan problem setinin özellikleri Çizelge 8.1'deki gibidir:

Çizelge 8.1. Test problemleri

Düğüm Sayısı	Hat Sayısı	Şebeke Türü	En İyi Çözüm
6	15	Tam bağlı şebeke	Biliniyor
7	21	Tam bağlı şebeke	Biliniyor
8	28	Tam bağlı şebeke	Biliniyor
9	36	Tam bağlı şebeke	Biliniyor
10	45	Tam bağlı şebeke	Biliniyor
11	55	Tam bağlı şebeke	Biliniyor
14	21	Tam bağlı olmayan şebeke	Biliniyor
16	24	Tam bağlı olmayan şebeke	Biliniyor
20	30	Tam bağlı olmayan şebeke	Biliniyor
15	105	Tam bağlı şebeke	Bilinmiyor
20	190	Tam bağlı şebeke	Bilinmiyor
25	300	Tam bağlı şebeke	Bilinmiyor
30	435	Tam bağlı şebeke	Bilinmiyor
40	780	Tam bağlı şebeke	Bilinmiyor

Şebeke türü olarak verilen tam bağlı şebeke tüm düğüm çiftlerinin birbiriyle direkt olarak bağlanabilen şebekeler, tam bağlı olmayan şebekeler ise tüm düğüm çiftlerinin birbiriyle direkt bağlanamadığı şebekelerdir. Tam bağlı şebekelerden 11 düğümlü şebeke ile tam bağlı olmayan şebekelerin tümü haricinde diğer şebekeler için hat güvenilirliği p ve istenen güvenilirlik düzeyi p_0 için 3 farklı kombinasyon dikkate

alınmış ve her şebeke türüne ait 5 farklı maliyet matrisi kullanılmıştır. Dolayısıyla bir şebeke boyutu için toplam 15 adet problem elde edilmektedir. Problemlerde yer alan p ve p_0 kombinasyonları Çizelge 8.2'deki gibidir:

Çizelge 8.2. Hat ve şebeke güvenilirliği kombinasyonları

p	p_0
0.9	0.9
0.9	0.95
0.95	0.95

Bu çalışmada test problemlerinin çözümünde Intel Centrino 1.73 GHz işlemciye ve 512 MB DDR2 belleğe sahip bir bilgisayarda PASCAL programlama dili kullanılarak gerçekleştirilmiştir. Geliştirilen algoritmaları yukarıda belirtilen etkinlik ölçütleri göz önünde bulundurarak değerlendirmek amacıyla 154 adet test problemi kullanılmıştır. [68,69]

8.3. Deney Tasarımı

Daha önce de belirtildiği üzere KSE ve KKE algoritmalarına ait parametrelerin düzeylerinin belirlenmesinde deney tasarımı kullanılmıştır. Deney tasarımı çalışmasında 3^k faktöriyel tasarım dikkate alınmıştır. Bu tasarımda gerçekleştirilen her bir parametre için 3 adet düzey belirlenmiştir ve faktör düzeylerinin her kombinasyonu için 5 adet en iyi değerden sapma değeri elde edilmiştir. k faktör sayısını göstermek üzere toplam $5x3^k$ tane gözlem yapılmıştır.

8.3.1 KSE algoritması için uygun parametre kümesinin belirlenmesi

KSE algoritmasında deney tasarımı gerçekleştirilecek olan parametreler ve bu parametrelerin düzeyleri Çizelge 8.3'te verilmiştir:

Çizelge 8.3. KSE algoritmasında dikkate alınan parametreler ve düzeyleri

Parametre	**Düzey**		
	1	**2**	**3**
Düğüm Sayısı (DS)	7	8	10
Eylemsizlik ağırlığı (w)	0,8	1	1,2
Bilişsel bileşen (c_1)	2	6	10
Sosyal bileşen (c_2)	2	6	10
Kuşların maksimum hızı (V_{max})	1	2	4
Sürü büyüklüğü (SB)	2 x hat sayısı	3 x hat sayısı	4 x hat sayısı

Faktöriyel tasarımda düğüm sayısı parametresi için 7, 8 ve 10 düğümlü tam bağlı şebekeler kullanılmıştır. Eylemsizlik ağırlığı (w) için ise 0.8, 1 ve 1,2 değerleri, hız eşitliğinin bilişsel bileşen (c1) ve sosyal bileşen (c2) parametreleri için 2, 6 ve 10 değerleri incelenmiştir. Kuşların hızlarının çıkabileceği maksimum değeri belirten V_{max} parametresi için ise 1, 2 ve 4 değerleri incelenmiştir. Son olarak da sürünün büyüklüğü için ise incelenecek değerler probleme bağlı olarak hat sayısının 2, 3 ve 4 katı için inceleme yapılmıştır.

Her bir faktör kombinasyonu için 5'er deneme yapılarak toplam 3645 ($=5x3^6$) deneme gerçekleştirilmiştir. Bu deneme sonucunda ANOVA (Analysis of Variance) Çizelgesi oluşturularak hangi parametrelerin deney üzerinde etkili olduğuna karar verilmiştir. Çizelge 8.4'te ANOVA Çizelgesi verilmiştir. Daha sonra Duncan Çoklu Açıklık Testi yapılarak parametre düzeyleri arasında anlamlı bir farklılık olup olmadığına karar verilmiştir. Çizelge 8.5'te bu teste ait sonuçlar yer almaktadır. Bunun sonucunda da uygun parametre değerleri belirlenmiştir.

ANOVA Çizelgesi incelendiğinde ana etkilerden yalnızca c_1'in anlamlı etkiye sahip olmadığı, diğer parametrelerin anlamlı etkiye sahip oldukları görülmektedir. Ana etkilerin hangi düzeylerde anlamlı farklılıklara sahip olduğu ise Çizelge 8.5'te incelenmiştir.

Çizelge 8.4. KSE parametreleri için ANOVA çizelgesi

Kaynak	Serbestlik Derecesi	Toplam Kare	Ortalama Kare	F Değeri	P Değeri
Ana Etki					
DS	2	116348,6	58174,3	1016*	0,000
w	2	11579,1	5789,6	101,11*	0,000
c_1	2	6,6	3,3	0,06	0,944
c_2	2	994,3	497,2	8,68*	0,000
V_{max}	2	13334,0	6667,0	116,44*	0,000
SB	2	1573,6	786,8	13,74*	0,000
2 - ortak					
DS * w	4	21036,6	5259,2	91,85*	0,000
DS *c_1	4	7,3	1,8	0,03	0,998
DS *c_2	4	2058,1	514,5	8,99*	0,000
DS *V_{max}	4	25475,6	6368,9	111,23*	0,000
DS * SB	4	2743,1	685,8	11,98*	0,000
w *c_1	4	481,2	120,3	2,10	0,078
w *c_2	4	341,0	85,3	1,49	0,203
w *V_{max}	4	62328,0	15582,0	272,14*	0,000
w * SB	4	512,9	128,2	2,24	0,062
c_1*c_2	4	234,9	58,7	1,03	0,392
c_1*V_{max}	4	183,1	45,8	0,80	0,525
c_1* SB	4	94,7	23,7	0,41	0,799
c_2*V_{max}	4	141,1	35,3	0,62	0,651
c_2* SB	4	209,0	52,3	0,91	0,456
V_{max}* SB	4	249,7	62,4	1,09	0,359
Error	3572	204526,4	57,3		
Total	3644	464459,3			

* $\alpha = 0.05$ geçerlilik düzeyinde anlamlı

Çizelge 8.5. KSE algoritması parametreleri için duncan çoklu açıklık testi sonuçları

Faktörler	Düzeyler	N	Grup Ortalaması	Düzey Kombinasyonu	Anlamlı Fark
Düğüm Sayısı	7	243	0,0070	7-8	Yok
	8	243	0,1403	7-10	Var
	10	243	12,0581	8-10	Var
w	0.8	243	6,5813	0.8-1	Var
	1	243	2,9831	0.8-1.2	Var
	1.2	243	2,6409	1-1.2	Yok
c_1	2	243	4,1127	2-6	Yok
	6	243	4,0815	2-10	Yok
	10	243	4,0112	6-10	Yok
c_2	2	243	3,6637	2-6	Var
	6	243	3,7358	2-10	Var
	10	243	4,8059	6-10	Yok
V_{max}	1	243	5,2553	1-2	Var
	2	243	1,3701	1-4	Yok
	4	243	5,5800	2-4	Var
SB	2x hat sayısı	243	4,9731	2-3	Yok
	3x hat sayısı	243	3,8001	2-4	Var
	4x hat sayısı	243	3,4322	3-4	Var

Çizelge 8.5 incelendiğinde *w* değerinin alt ve orta değerleri arasında, c_1 değerinin hiçbir düzeyi arasında, c_2 değerinin orta ve üst düzeyleri arasında ve V_{max} değerinin alt ve üst değerleri arasında anlamlı bir farklılık bulunmamaktadır. Parametreler için seçilen uygun değerler aşağıda verilmiştir:

w: 1,2

c_1: 6

c_2: 2

V_{max}: 2

SB: 4xhat sayısı

8.3.2. KKE algoritması için uygun parametre kümesinin belirlenmesi

KKE algoritmasında deney tasarımı gerçekleştirilecek olan parametreler ve bu parametrelerin düzeyleri Çizelge 8.6'da verilmiştir:

Çizelge 8.6. KKE algoritmasında dikkate alınan parametreler ve düzeyleri

Parametre	**Düzey**		
	1	**2**	**3**
Düğüm Sayısı	7	8	10
α	1	2	3
β	1	2	3
q_0	0.1	0.4	0.7
Buharlaşma Oranı (ρ)	0.3	0.2	0.1

Faktöriyel tasarımda düğüm sayısı parametresi için 7, 8 ve 10 düğümlü tam bağlı şebekeler kullanılmıştır. α ve β değerleri için 1, 2 ve 3 değerleri, q_0 değeri için ise 0,1, 0,4 ve 0,7 değerleri incelenmiştir. Son olarak da feromenlerin her iterasyonda buharlaşma miktarlarını belirleyen ρ parametresi için ise 0,1, 0,2 ve 0,3 değerleri incelenmiştir.

Her bir faktör kombinasyonu için 5'er deneme yapılarak toplam 1215 ($=5x3^5$) deneme gerçekleştirilmiştir. Bu deneme sonucunda ANOVA (Analysis of Variance) Çizelgesi oluşturularak hangi parametrelerin deney üzerinde etkili olduğuna karar verilmiştir. Çizelge 8.7'de ANOVA Çizelgesi verilmiştir. Daha sonra Duncan Çoklu Açıklık Testi yapılarak parametre düzeyleri arasında anlamlı bir farklılık olup olmadığına karar verilmiştir. Çizelge 8.8'de bu teste ait sonuçlar yer almaktadır. Bunun sonucunda da uygun parametre değerleri belirlenmiştir.

ANOVA Çizelgesi α %5 anlamlılık düzeyinde elde edilmiştir. *F* değerleri göz önüne alınarak incelendiğinde düğüm sayısı, α, β ve q_0 parametrelerinin algoritma performansı üzerinde etkili olduğu görülmüştür. ρ değeri ise algoritma üzerinde anlamlı bir etkiye sahip değildir. İkili ortak etkiler incelendiğinde ise düğüm sayısı

ve β ile düğüm sayısı ve q_0 parametrelerinin birlikte anlamlı bir etkiye sahip olduğu görülmektedir. Üçlü, dörtlü ve beşli ortak etkilerden anlamlı bir etki ortaya çıkmamaktadır.

Çizelge 8.7. KKE parametreleri için ANOVA çizelgesi

Kaynak	**Serbestlik Derecesi**	**Toplam Kare**	**Ortalama Kare**	**F Değeri**	**P Değeri**
Ana Etki					
DS	2	873,422	436,711	211,18*	0,000
α	2	32,911	16,456	7,96*	0,000
β	2	22,033	11,017	5,33*	0,000
q_0	2	27,728	13,864	6,70*	0,005
ρ	2	12,130	6,065	2,93*	0,001
2 - ortak					
DS*α	4	20,328	5,082	2,46	0,054
DS*β	4	63,043	15,761	7,62*	0,044
DS *q_0	4	23,180	5,795	2,80*	0,000
DS * ρ	4	11,158	2,789	1,35*	0,025
α*β	4	1,746	0,436	0,21	0,250
α*q_0	4	13,849	3,462	1,67	0,932
α* ρ	4	1,530	0,382	0,18	0,154
β*q_0	4	3,348	0,837	0,40	0,946
β* ρ	4	5,727	1,432	0,69	0,805
q0*ρ	4	4,695	1,174	0,57	0,597
Hata	1164	2407,049	2,068		0,686
Toplam	1214	3523,880			

* α = 0.05 geçerlilik düzeyinde anlamlı

Çizelge 8.8. KKE algoritması parametreleri için duncan çoklu açıklık testi sonuçları

Faktörler	Düzeyler	N	Grup Ortalaması	Düzey Kombinasyonu	Anlamlı Fark
Düğüm Sayısı	7	81	0,000	7-8	Var
	8	81	1,062	7-10	Var
	10	81	2,077	8-10	Var
α	1	81	0,982	1-2	Yok
	2	81	0,885	1-3	Var
	3	81	1,272	2-3	Var
β	1	81	1,115	1-2	Yok
	2	81	1,166	1-3	Var
	3	81	0,858	2-3	Var
q_0	0.1	81	1,256	0.1-0.4	Var
	0.4	81	0,977	0.1-0.7	Var
	0.7	81	0,907	0.4-0.7	Yok
Buharlaşma Oranı (ρ)	0.1	81	0,966	0.1-0.2	Yok
	0.2	81	0,986	0.1-0.3	Yok
	0.3	81	1,187	0.2-0.3	Yok

Çizelge 8.8 incelendiğinde α değerinin alt ve orta değerleri arasında, c_1 değerinin hiçbir düzeyi arasında, c_2 değerinin alt ve orta düzeyleri arasında, ve q_0 değerinin orta ve üst değerleri arasında, son olarak da ρ değerinin hiçbir düzeyi arasında anlamlı bir farklılık bulunmamaktadır. Parametreler için seçilen uygun değerler aşağıda verilmiştir:

α: 2

β: 3

q_0: 0,4

ρ: 0,1

8.4. Sonuçların Değerlendirilmesi

Çalışmanın bu bölümünde deney tasarımı sonuçlarına göre belirlenen en uygun parametre değerleri kullanılarak elde edilen algoritma sonuçlarının etkinlikleri değerlendirilecektir. Algoritmaların etkinlikleri değerlendirilirken öncelikle her bir

çözüm algoritmasının varyasyonları kendi aralarında incelenecek, daha sonra her bir algoritmanın en iyi varyasyonları birbirleriyle karşılaştırılacaktır. Geliştirilen algoritmaların etkinliklerinin değerlendirilmesinde kullanılacak olan ölçütler en iyi değerden sapma oranı, değişim katsayısı ve çözüm zamanı olacaktır.

Geliştirilen algoritmaların etkinliklerinin değerlendirilmesinde kullanılacak olan problemler Bölüm 8.2'de belirtildiği gibi boyutları 6 – 11 düğüm arasında değişen en iyi çözüm değerleri bilinen tam bağlı şebekler ile 14, 16 ve 20 düğümlü en iyi çözüm değerleri bilinen tam bağlı olmayan şebeklerdir. Ayrıca en iyi çözümleri bilinmeyen 15, 20, 25, 30 ve 40 düğümlü tam bağlı şebekeler üzerinde de değerlendirme yapılacaktır.

Geliştirilen algoritmalarla küçük boyutlu (6 – 11 düğüm) tam bağlı şebekeler ile tam bağlı olmayan şebekelerin çözümleri elde edilirken her bir problem türü için 10'ar adet deneme yapılmıştır. Büyük boyutlu problemlerde ise 15, 20 ve 25 düğümlü şebekeler için 10'ar adet, 30 ve 40 düğümlü şebekeler için ise 5'er adet deneme yapılmıştır. 6 – 10 düğümlü şebekelerde 5 farklı maliyet matrisi kullanılmış ve güvenilirlik olasılıklarının 3 farklı türü dikkate alınmıştır. Büyük boyutlu problemler için (15, 20, 25, 30 ve 40 düğüm) yine 5 farklı maliyet matrisi ile birlikte 3 farklı güvenilirlik olasılığı dikkate alınmıştır. Böylelikle toplam 154 adet test problemi elde edilmiştir.

Geliştirilen algoritmaların etkinliklerini değerlendirirken küçük boyutlu ve tam bağlı şebekeler, tam bağlı olmayan şebekeler ve büyük boyutlu şebekeler şeklinde bir sınıflandırmaya gidilecektir.

Sonuçlar elde edilirken yapılan denemelerde küçük boyutlu ve tam bağlı şebekeler ile tam bağlı olmayan şebekeler geliştirilen tüm algoritmalarda kullanılmış, fakat büyük boyutlu problemler için daha iyi çözüm kalitesine sahip olan KSE_TB_2 algoritması ile TB_KKE_TB algoritması değerlendirmeye alınmıştır. Büyük boyutlu problemlerde en iyi sonuçlar bilinmediğinden sadece değişim katsayısı ve çözüm zamanı açısından karşılaştırma yapılmıştır. KSE_TB_2 algoritması ile 15, 20, 25, 30

ve 40 düğümlü problemlere ait çözümler elde edilebilirken, TB_KKE_TB algoritması ile 15, 20, 25 ve 30 düğümlü problemler için çözüm elde edilebilmiştir.

8.4.1. Geliştirilen algoritmaların en iyi değerden sapma oranına göre değerlendirilmesi

Daha önce de belirtildiği gibi küçük boyutlu tam bağlı şebeke problemleri ile tam bağlı olmayan şebeke problemlerinin en iyi sonuçları bilinmektedir. Geliştirilen algoritmaların bu problemler üzerindeki etkinliklerini görebilmek amacıyla 10'ar deneme yapılmıştır. Bu kısımda geliştirilen algoritmaların en iyi değerleri bilinen problemler için en iyi değerden sapma oranına göre performansları değerlendirilecektir. Bu değerlendirme sırasıyla DKİ, KSE ve KKE algoritmaları için yapılacaktır. Daha sonra her bir algoritmanın en iyi varyasyonu birbiriyle karşılaştırılacaktır.

Çizelge 8.9'da geliştirilen DKİ algoritmasının 6 – 11 düğümlü problemler için en iyi değerden sapma oranlarına ilişkin veriler yer almaktadır.

Çizelge 8.9. Geliştirilen DKİ algoritmalarının 6 – 11 düğümlü şebekeler için en iyi değerden sapma oranları

Geliştirilen DKİ Algoritmaları	Şebeke Boyutu						Genel Ortalama
	G(6,15)	G(7,21)	G(8,28)	G(936)	G(10,45)	G(11,45)	
DKİ	%4,596	%2,244	%1,830	%3,820	%2,682	%0,610	%1,866
TL_DKİ	%5,616	%1,997	%0,126	%4,418	%3,684	%0,000	%2,036

6 – 11 düğümlü problemler için en iyi değerden sapma oranına göre elde edilen değerler incelendiğinde DKİ algoritmasının çoğunlukla TL_DKİ'ye göre daha iyi bir performans gösterdiği görülmektedir. DKİ algoritmasının en iyi değerden sapma oranı ortalama olarak %1,866 iken TL_DKİ algoritması için bu değer %2,036'dır.

Çizelge 8.10'da geliştirilen DKİ algoritmasının tam bağlı olmayan şebeke problemleri için en iyi değerden sapma oranlarına ilişkin veriler yer almaktadır.

Çizelge 8.10. Geliştirilen DKİ algoritmalarının tam bağlı olmayan şebekeler için en iyi değerden sapma oranları

Geliştirilen DKİ Algoritmaları	Şebeke Boyutu			Genel Ortalama
	G(14,21)	G(16,24)	G(20,30)	
DKİ	%0,038	%0,000	%0,973	%0,337
TL_DKİ	%0,000	%0,000	%0,772	%0,257

Geliştirilen DKİ algoritmalarının en iyi değerden sapma oranları tam bağlı olmayan şebekeler için incelendiğinde TL_DKİ algoritmasının daha iyi performans gösterdiği anlaşılmaktadır.

Geliştirilen KSE algoritmalarının 6 – 11 düğümlü tam bağlı şebeke problemleri için en iyi değerden sapma oranlarına ilişkin veriler Çizelge 8.11'de yer almaktadır.

Çizelge 8.11. Geliştirilen KSE algoritmalarının 6 – 11 düğümlü tam bağlı şebeke problemleri için en iyi değerden sapma oranları

Geliştirilen KSE Algoritmaları	Şebeke Boyutu						Genel Ortalama
	G(6,15)	G(7,21)	G(8,28)	G(936)	G(10,45)	G(11,45)	
Temel KSE	%0,012	%0,622	%0,163	%0,382	%0,724	%0,000	%0,317
KSE_D	%0,000	%0,664	%0,126	%0,387	%0,537	%0,203	%0,319
KSE_TB_1	%0,000	%0,242	%0,056	%0,355	%0,754	%0,610	%0,336
KSE_D_TB_1	%0,000	%0,624	%0,080	%0,266	%0,957	%0,000	%0,321
KSE_TB_{DBS}_1	%0,110	%0,602	%0,044	%5,985	%0,428	%0,163	%1,222
KSE_D_TB_{DBS}_1	%0,000	%0,486	%0,075	%0,454	%0,955	%0,163	%0,356
KSE_TB_2	%0,126	%0,085	%0,065	%0,191	%0,166	%0,000	%0,106
KSE_D_TB_2	%0,000	%0,355	%0,057	%0,290	%0,138	%0,081	%0,154
KSE_TB_{DBS}_2	%0,019	%0,488	%0,056	%0,120	%0,168	%0,000	%0,142
KSE_D_TB_{DBS}_2	%0,068	%0,244	%0,053	%0,107	%0,123	%0,163	%0,126

Çizelge 8.11 incelendiğnde en iyi değerden sapma oranına göre en iyi performansı KSE_TB_2 olarak adlandırılan algoritma vermektedir. Bu algoritma için en iyi değerden sapmanın ortalama değeri %0,106'dır. En kötü performansı ise

$KSE_D\_TB_{DBS}\_1$ algoritması göstermektedir. Bu algoritma için en iyi değerden sapmanın ortalama değeri %0,356'dır.

Geliştirilen KSE algoritmalarının tam bağlı olmayan şebeke problemlerinde en iyi değerden sapma oranına göre gösterdikleri performans değerleri Çizelge 8.12'de verilmektedir. Buna göre, KSE_TB_1 ve KSE_D_TB_1 dışındaki algoritmalarda en iyi değerden sapma oranları %0 olmuştur. Bu algoritmaların en iyi değerden sapma oranları ise sırasıyla %0,905 ve %0,527 olarak gerçekleşmiştir.

Çizelge 8.12. Geliştirilen KSE algoritmalarının tam bağlı olmayan şebeke problemleri için oıpimumdan sapma oranları

Geliştirilen KSE Algoritmaları	Şebeke Boyutu			Genel Ortalama
	G(14,21)	G(16,24)	G(20,30)	
Temel KSE	% 0,012	% 0,622	% 0,163	% 0,000
KSE_D	% 0,000	% 0,664	% 0,126	% 0,000
KSE_TB_1	% 0,000	% 0,242	% 0,056	% 0,905
KSE_D_TB_1	% 0,000	% 0,624	% 0,080	% 0,527
$KSE\_TB_{DBS}\_1$	% 0,110	% 0,602	% 0,044	% 0,000
$KSE_D\_TB_{DBS}\_1$	% 0,000	% 0,486	% 0,075	% 0,000
KSE_TB_2	% 0,126	% 0,085	% 0,065	% 0,000
KSE_D_TB_2	% 0,000	% 0,355	% 0,057	% 0,000
$KSE\_TB_{DBS}\_2$	% 0,019	% 0,488	% 0,056	% 0,000
$KSE_D\_TB_{DBS}\_2$	% 0,068	% 0,244	% 0,053	% 0,000

Geliştirilen KKE algoritmalarının 6 – 11 düğümlü tam bağlı şebeke problemlerinde en iyi değerden sapma oranına göre karşılaştırılmasına ilişkin veriler Çizelge 8.13'te yer almaktadır. Değerler incelendiğinde en düşük ortalamaya sahip olan algoritmanın %0,30 değeri ile TB_KKE_TB algoritması olduğu görülmektedir. En yüksek ortalamaya sahip olan algoritma ise %1,302 değeri ile KKE_D_TB algoritmasıdır.

Geliştirilen KKE algoritmalarının tam bağlı olmayan şebeke problemlerinde en iyi değerden sapma oranına göre karşılaştırılmasına ilişkin veriler Çizelge 8.14'te yer verilmektedir.

Çizelge 8.13. Geliştirilen KKE algoritmalarının 6 – 11 düğümlü tam bağlı şebeke problemlerinde en iyi değerden sapma oranları

Geliştirilen KKE Algoritmaları	Şebeke Boyutu						Genel Ortalama
	G(6,15)	G(7,21)	G(8,28)	G(936)	G(10,45)	G(11,45)	
TB_KKE	% 0,220	% 0,259	% 0,530	% 0,810	% 0,424	% 0,000	% 0,374
TB_KKE$_D$	% 0,685	% 1,168	% 0,798	% 1,528	% 1,054	% 0,000	% 0,872
KKE_TB	% 0,364	% 0,536	% 0,721	% 1,295	% 1,258	% 0,000	% 0,696
KKE$_D$_TB	% 1,273	% 1,371	% 1,070	% 2,181	% 1,916	% 0,000	% 1,302
KKE_TB$_{DBS}$	% 0,224	% 0,333	% 0,495	% 1,718	% 1,303	% 0,000	% 0,679
KKE$_D$_TB$_{DBS}$	% 0,916	% 1,029	% 0,991	% 1,957	% 1,738	% 0,000	% 1,105
TB_KKE_TB	% 0,193	% 0,012	% 0,284	% 0,770	% 0,554	% 0,000	% 0,302
TB_KKE$_D$_TB	% 0,667	% 0,742	% 0,964	% 2,068	% 0,962	% 0,203	% 0,934
TB_KKE_TB$_{DBS}$	% 0,000	% 0,179	% 0,323	% 1,028	% 0,793	% 0,000	% 0,387
TB_KKE$_D$_TB$_{DBS}$	% 0,776	% 0,979	% 1,075	% 1,277	% 1,129	% 0,244	% 0,914

Çizelge 8.14. Geliştirilen KKE algoritmalarının tam bağlı olmayan şebeke problemlerinde en iyi değerden sapma oranları

Geliştirilen KKE Algoritmaları	Şebeke Boyutu			Genel Ortalama
	G(14,21)	G(16,24)	G(20,30)	
TB_KKE	% 0,000	% 0,059	% 2,181	% 0,747
TB_KKE$_D$	% 0,047	% 0,068	% 2,114	% 0,743
KKE_TB	% 0,179	% 0,264	% 2,953	% 1,132
KKE$_D$_TB	% 0,169	% 0,421	% 2,534	% 1,041
KKE_TB$_{DBS}$	% 0,179	% 0,401	% 2,953	% 1,178
KKE$_D$_TB$_{DBS}$	% 0,169	% 0,294	% 1,762	% 0,742
TB_KKE_TB	% 0,000	% 0,000	% 0,000	% 0,000
TB_KKE$_D$_TB	% 0,000	% 0,068	% 2,601	% 0,890
TB_KKE_TB$_{DBS}$	% 0,169	% 0,049	% 1,477	% 0,565
TB_KKE$_D$_TB$_{DBS}$	% 0,442	% 0,068	% 2,601	% 1,037

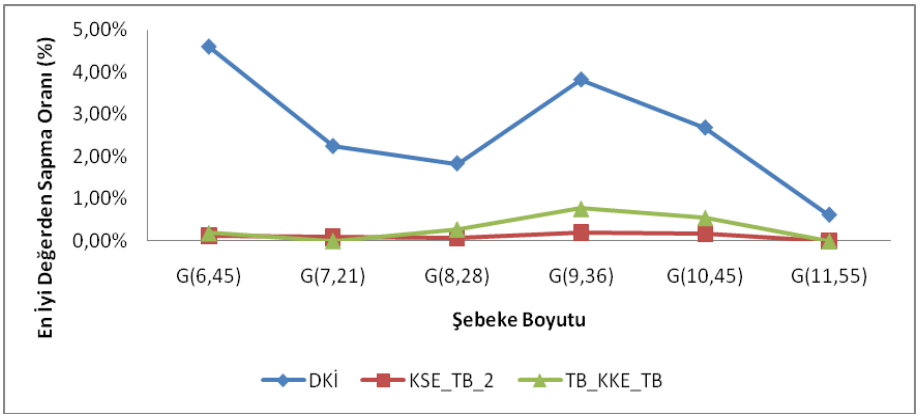
Çizelge 8.14'teki değerler incelendiğinde en iyi performansı %0 değeri ile TB_KKE_TB algoritmasının gösterdiği görülmektedir. En kötü performansı ise %1,178 değeri ile KKE_TB$_{DBS}$ algoritması göstermektedir.

Her bir algoritmanın ayrı ayrı en iyi değerden sapma oranları incelendikten sonra bu algoritmalardan elde edilen en iyi değerlerin de karşılaştırılması faydalı olacaktır. Bu aşamada 6 – 11 düğümlü tam bağlı şebekeler için karşılaştırılacak olan algoritmalar

DKİ, KSE_TB_2 ve TB_KKE_TB algoritmalarıdır. Çizelge 8.15'te bu üç algoritmanın 6 – 11 düğümlü tam bağlı şebeke problemlerinde en iyi değerden sapma oranlarına ilişkin değerleri yer almaktadır. Şekil 8.1'de de bu çizelgenin grafiksel gösterimi yer almaktadır.

Çizelge 8.15. Geliştirilen en iyi algoritmaların 6 – 11 düğümlü tam bağlı şebeke problemlerinde en iyi değerden sapma oranları

Geliştirilen Algoritma	Şebeke Boyutu						Genel Ortalama
	G(6,45)	G(7,21)	G(8,28)	G(9,36)	G(10,45)	G(11,55)	
DKİ	% 4,60	% 2,24	% 1,83	% 3,82	% 2,68	% 0,61	% 2,63
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



Şekil 8.1. Geliştirilen en iyi algoritmaların 6 – 11 düğümlü tam bağlı şebeke problemlerinde en iyi değerden sapma oranlarının karşılaştırılması

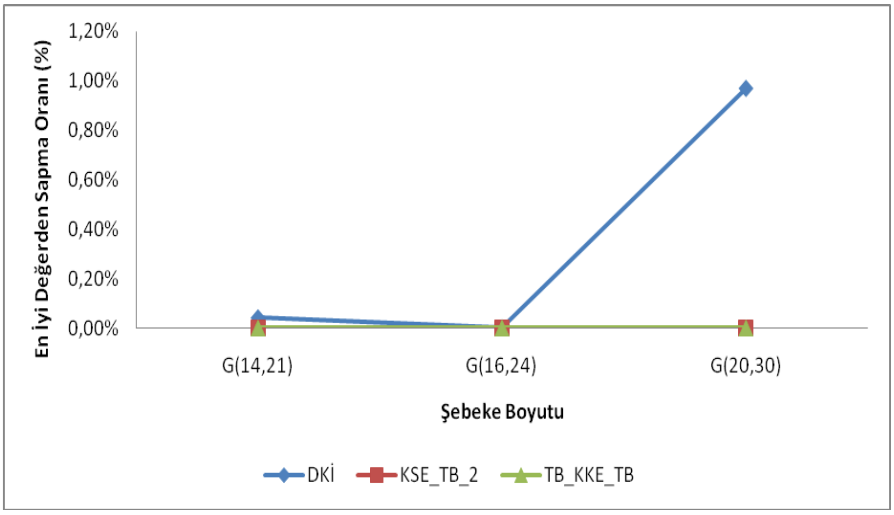
Veriler incelendiğinde en az oranda en iyi değerden sapmanın gerçekleştiği algoritma KSE_TB_2 algoritması, en fazla oranda en iyi değerden sapmanın gerçekleştiği algoritma ise DKİ algoritmasıdır.

Yukarıda yapılan değerlendirme tam bağlı olmayan şebeke problemleri için de gerçekleştirilecektir. Çizelge 8.16'da geliştirilen algoritmaların tam bağlı olmayan

şebeke problemlerinde en iyi değerden sapma oranlarına ilişkin değerleri yer almaktadır. Şekil 8.2'de de bu çizelgenin grafiksel gösterimi yer almaktadır.

Çizelge 8.16. Geliştirilen en iyi algoritmaların tam bağlı olmayan şebeke problemlerinde en iyi değerden sapma oranları

Geliştirilen Algoritma	Şebeke Boyutu			Genel Ortalama
	G(14,21)	G(16,24)	G(20,30)	
DKİ	% 0,040	% 0,000	% 0,97	% 0,337
KSE_TB_2	% 0,000	% 0,000	% 0,000	% 0,000
TB_KKE_TB	% 0,000	% 0,000	% 0,000	% 0,000



Şekil 8.2. Geliştirilen en iyi algoritmaların tam bağlı olmayan şebeke problemlerinde en iyi değerden sapma oranlarının karşılaştırılması

Elde edilen değerler incelendiğinde KSE_TB_2 ve SA_KKE_SA algoritmaları tam bağlı olmayan şebeke problemlerinin tümünde en iyi değerden hiç sapma göstermemiştir. Yalnızca DKİ algoritması 20 düğümlü şebekede %0,97 oranında bir sapma göstermiştir.

Son olarak, KSE ve KKE ile geliştirilen algoritmalar arasında en iyi değerden sapma oranına göre anlamlı bir fark olup olmadığı incelenmiştir. ANOVA testi kullanılarak yapılan analizde KSE ile geliştirilen on farklı algoritmanın en iyi değerden sapma

oranına göre kendi aralarında anlamlı bir farklılığa sahip olduğu görülmüştür. Bunun yanında KKE ile geliştirilen on farklı algoritmanın en iyi değerden sapma oranına göre kendi aralarında anlamlı bir farklılığa sahip olmadığı görülmüştür. Çizelge 8.17'de geliştirilen KSE algoritmaları ile gerçekleştirilen en iyi değerden sapma oranına göre gerçekleştirilen ANOVA testine ilişkin sonuçlar, Çizelge 8.18'de Tukey testi sonuçları ve Çizelge 8.19'da geliştirilen KKE algoritmaları ile en iyi değerden sapma oranına göre gerçekleştirilen ANOVA testi sonuçları ve Çizelge 8.20'de Tukey testi sonuçları yer almaktadır.

Çizelge 8.17. Geliştirilen KSE algoritmalarının en iyi değerden sapma oranına göre elde edilen ANOVA testi sonuçları

	Kareler Toplamı	Serbestlik Derecesi	Ortalama Kare	F Değeri	P Değeri
Gruplar arası	0,009	9	0,001	3,946	0,000
Grup içi	0,199	780	0,000		
Toplam	0,208	789			

ANOVA testi %95 güvenilirlik düzeyinde gerçekleştirilmiştir ve P değeri 0,05'ten küçük olduğundan geliştirilen KSE algoritmaları arasında en iyi değerden sapma oranına göre anlamlı bir fark vardır.

Çizelge 8.18. Geliştirilen KSE algoritmalarının en iyi değerden sapma oranına göre Tukey Testi ile karşılaştırılmasına ait sonuçlar

Grup (I)	Grup (J)	Ortalama Fark (I-J)	Std. Hata	Anlamlılık	%95 Güven Aralığı	
		Alt Sınır	Üst Sınır	Alt Sınır	Üst Sınır	Alt Sınır
KSE	KSE_D	0,0006	0,0025	1	-0,0074	0,0087
	KSE_TB_1	0,0006	0,0025	1	-0,0074	0,0087
	KSE_D_TB_1	0	0,0025	1	-0,0081	0,0081
	KSE_TB_{DBS}_1	-0,0098(*)	0,0025	0,005	-0,0178	-0,0017
	KSE_D_TB_{DBS}_1	0,0003	0,0025	1	-0,0078	0,0083
	KSE_TB_2	0,0023	0,0025	0,997	-0,0058	0,0103
	KSE_D_TB_2	0,0019	0,0025	0,999	-0,0062	0,01
	KSE_TB_{DBS}_2	0,0019	0,0025	0,999	-0,0062	0,01
	KSE_D_TB_{DBS}_2	0,0025	0,0025	0,992	-0,0055	0,0106
KSE_TB_1	KSE_TB_1	-0,0006	0,0025	1	-0,0087	0,0074
	KSE_D_TB_1	0	0,0025	1	-0,0081	0,0081
	KSE_TB_{DBS}_1	-0,0006	0,0025	1	-0,0087	0,0074
	KSE_D_TB_{DBS}_1	-0,0104(*)	0,0025	0,002	-0,0184	-0,0023

Çizelge 8.18. (Devam) Geliştirilen KSE algoritmalarının en iyi değerden sapma oranına göre Tukey Testi ile karşılaştırılmasına ait sonuçlar

Grup (I)	Grup (J)	Ortalama Fark (I-J)	Std. Hata	Anlamlılık	%95 Güven Aralığı	
		Alt Sınır	Üst Sınır	Alt Sınır	Üst Sınır	Alt Sınır
	KSE_TB_2	-0,0004	0,0025	1	-0,0084	0,0077
	KSE_D_TB_2	0,0017	0,0025	1	-0,0064	0,0097
	KSE_TB_{DBS}_2	0,0013	0,0025	1	-0,0068	0,0093
	KSE_D_TB_{DBS}_2	0,0013	0,0025	1	-0,0068	0,0093
KSE_D_TB_1	KSE_D_TB_1	0,0019	0,0025	0,999	-0,0062	0,01
	KSE_TB_{DBS}_1	-0,0006	0,0025	1	-0,0087	0,0074
	KSE_D_TB_{DBS}_1	0	0,0025	1	-0,0081	0,0081
	KSE_TB_2	-0,0006	0,0025	1	-0,0087	0,0074
	KSE_D_TB_2	-0,0104(*)	0,0025	0,002	-0,0184	-0,0023
	KSE_TB_{DBS}_2	-0,0004	0,0025	1	-0,0084	0,0077
	KSE_D_TB_{DBS}_2	0,0017	0,0025	1	-0,0064	0,0097
KSE_D_TB_1	KSE_TB_{DBS}_1	0,0013	0,0025	1	-0,0068	0,0093
	KSE_D_TB_{DBS}_1	0,0013	0,0025	1	-0,0068	0,0093
	KSE_TB_2	0,0019	0,0025	0,999	-0,0062	0,01
	KSE_D_TB_2	0	0,0025	1	-0,0081	0,0081
	KSE_TB_{DBS}_2	0,0006	0,0025	1	-0,0074	0,0087
	KSE_D_TB_{DBS}_2	0,0006	0,0025	1	-0,0074	0,0087
KSE_TB_{DBS}_1	KSE_D_TB_{DBS}_1	-0,0098(*)	0,0025	0,005	-0,0178	-0,0017
	KSE_TB_2	0,0003	0,0025	1	-0,0078	0,0083
	KSE_D_TB_2	0,0023	0,0025	0,997	-0,0058	0,0103
	KSE_TB_{DBS}_2	0,0019	0,0025	0,999	-0,0062	0,01
	KSE_D_TB_{DBS}_2	0,0019	0,0025	0,999	-0,0062	0,01
KSE_D_TB_{DBS}_1	KSE_TB_2	0,0025	0,0025	0,992	-0,0055	0,0106
	KSE_D_TB_2	-0,0098(*)	0,0025	0,005	0,0017	0,0178
	KSE_TB_{DBS}_2	0,0104(*)	0,0025	0,002	0,0023	0,0184
	KSE_D_TB_{DBS}_2	0,0104(*)	0,0025	0,002	0,0023	0,0184
KSE_TB_2	KSE_D_TB_2	0,0098(*)	0,0025	0,005	0,0017	0,0178
	KSE_TB_{DBS}_2	0,0100(*)	0,0025	0,004	0,0019	0,0181
	KSE_D_TB_{DBS}_2	0,0120(*)	0,0025	0	0,004	0,0201
KSE_D_TB_2	KSE_TB_{DBS}_2	0,0117(*)	0,0025	0	0,0036	0,0197
	KSE_D_TB_{DBS}_2	0,0117(*)	0,0025	0	0,0036	0,0197
KSE_TB_{DBS}_2	KSE_D_TB_{DBS}_2	0,0123(*)	0,0025	0	0,0042	0,0203

* Ortalama farklar 0,05 düzeyinde anlamlıdır..

Çizelge 8.19. Geliştirilen KKE algoritmalarının en iyi değerden sapma oranına göre elde edilen ANOVA testi sonuçları

	Kareler Toplamı	Serbestlik Derecesi	Ortalama Kare	F Değeri	P Değeri
Gruplar arası	0,011	9	0,001	6,064	0,000
Grup içi	0,157	780	0,000		
Toplam	0,168	789			

ANOVA testi %95 güvenilirlik düzeyinde gerçekleştirilmiştir ve P değeri 0,05'ten küçük olduğundan geliştirilen KKE algoritmaları arasında en iyi değerden sapma oranına göre anlamlı bir fark vardır.

Çizelge 8.20. Geliştirilen KKE algoritmalarının en iyi değerden sapma oranına göre Tukey Testi ile karşılaştırılmasına ait sonuçlar

Grup (I)	Grup (J)	Ortalama Fark (I-J)	Std. Hata	Anlamlılık	%95 Güven Aralığı	
		Alt Sınır	Üst Sınır	Alt Sınır	Üst Sınır	Alt Sınır
TB_KKE	TB_KKE_D	-0,0056	0,0023	0,289	-0,0127	0,0016
	KKE_TB	-0,0035	0,0023	0,863	-0,0107	0,0036
	KKE_D_TB	-0,0109(*)	0,0023	0	-0,0181	-0,0037
	KKE_TB_{DBS}	-0,0034	0,0023	0,887	-0,0106	0,0038
	$KKE_D\_TB_{DBS}$	-0,0082(*)	0,0023	0,011	-0,0154	-0,0011
	TB_KKE_TB	0,0015	0,0023	1	-0,0057	0,0087
	$TB\_KKE_D\_TB$	-0,0063	0,0023	0,138	-0,0135	0,0008
	$TB\_KKE\_TB_{DBS}$	0,0001	0,0023	1	-0,007	0,0073
	$TB\_KKE_D\_TB_{DBS}$	-0,0056	0,0023	0,289	-0,0127	0,0016
TB_KKE_D	KKE_TB	0,0056	0,0023	0,289	-0,0016	0,0127
	KKE_D_TB	0,0020	0,0023	0,997	-0,0051	0,0092
	KKE_TB_{DBS}	-0,0053	0,0023	0,356	-0,0125	0,0019
	$KKE_D\_TB_{DBS}$	0,0022	0,0023	0,995	-0,005	0,0093
	TB_KKE_TB	-0,0027	0,0023	0,976	-0,0098	0,0045
	$TB\_KKE_D\_TB$	0,0071	0,0023	0,056	-0,0001	0,0143
	$TB\_KKE\_TB_{DBS}$	-0,0008	0,0023	1	-0,0079	0,0064
	$TB\_KKE_D\_TB_{DBS}$	0,0057	0,0023	0,259	-0,0015	0,0129
KKE_TB	KKE_D_TB	0,0000	0,0023	1	-0,0072	0,0072
	KKE_TB_{DBS}	0,0035	0,0023	0,863	-0,0036	0,0107
	$KKE_D\_TB_{DBS}$	-0,0020	0,0023	0,997	-0,0092	0,0051
	TB_KKE_TB	-0,0073(*)	0,0023	0,04	-0,0145	-0,0002
	$TB\_KKE_D\_TB$	0,0001	0,0023	1	-0,007	0,0073
	$TB\_KKE\_TB_{DBS}$	-0,0047	0,0023	0,547	-0,0119	0,0025
	$TB\_KKE_D\_TB_{DBS}$	0,0051	0,0023	0,43	-0,0021	0,0122
KKE_D_TB	KKE_TB_{DBS}	-0,0028	0,0023	0,967	-0,01	0,0044
	$KKE_D\_TB_{DBS}$	0,0037	0,0023	0,836	-0,0035	0,0108
	TB_KKE_TB	-0,0020	0,0023	0,997	-0,0092	0,0051
	$TB\_KKE_D\_TB$	0,0109(*)	0,0023	0	0,0037	0,0181
	$TB\_KKE\_TB_{DBS}$	0,0053	0,0023	0,356	-0,0019	0,0125
	$TB\_KKE_D\_TB_{DBS}$	0,0073(*)	0,0023	0,04	0,0002	0,0145
KKE_TB_{DBS}	$KKE_D\_TB_{DBS}$	0,0075(*)	0,0023	0,033	0,0003	0,0146
	TB_KKE_TB	0,0027	0,0023	0,976	-0,0045	0,0098

Çizelge 8.20. (Devam) Geliştirilen KKE algoritmalarının en iyi değerden sapma oranına göre Tukey Testi ile karşılaştırılmasına ait sonuçlar

Grup (I)	Grup (J)	Ortalama Fark (I-J)	Std. Hata	Anlamlılık	%95 Güven Aralığı	
		Alt Sınır	Üst Sınır	Alt Sınır	Üst Sınır	Alt Sınır
	$TB\_KKE_D\_TB$	0,0124(*)	0,0023	0	0,0052	0,0196
	$TB\_KKE\_TB_{DBS}$	0,0046	0,0023	0,587	-0,0026	0,0117
	$TB\_KKE_D\_TB_{DBS}$	0,0110(*)	0,0023	0	0,0038	0,0182
$KKE_D\_TB_{DBS}$	TB_KKE_TB	0,0053	0,0023	0,356	-0,0019	0,0125
	$TB\_KKE_D\_TB$	0,0034	0,0023	0,887	-0,0038	0,0106
	$TB\_KKE\_TB_{DBS}$	-0,0022	0,0023	0,995	-0,0093	0,005
	$TB\_KKE_D\_TB_{DBS}$	-0,0001	0,0023	1	-0,0073	0,007
TB_KKE_TB	$TB\_KKE_D\_TB$	-0,0075(*)	0,0023	0,033	-0,0146	-0,0003
	$TB\_KKE\_TB_{DBS}$	-0,0048	0,0023	0,507	-0,012	0,0024
	$TB\_KKE_D\_TB_{DBS}$	0,0049	0,0023	0,468	-0,0022	0,0121
$TB\_KKE_D\_TB$	$TB\_KKE\_TB_{DBS}$	-0,0029	0,0023	0,956	-0,0101	0,0043
	$TB\_KKE_D\_TB_{DBS}$	0,0035	0,0023	0,863	-0,0036	0,0107
$TB\_KKE\_TB_{DBS}$	$TB\_KKE_D\_TB_{DBS}$	-0,0022	0,0023	0,995	-0,0093	0,005

* Ortalama farklar 0,05 düzeyinde anlamlıdır..

8.4.2. Geliştirilen algoritmaların değişim katsayısına göre değerlendirilmesi

Bu kısımda, geliştirilen algoritmaların performansları değişim katsayısı değerleri göz önüne alınarak değerlendirilecektir. Değişim katsayılarının hesaplanması Eş. 8.2'de belirtilen şekilde hesaplanacaktır. Değişim katsayılarına ilişkin değerlendirmeler sırasıyla DKİ, KSE ve KKE algoritmaları için yapılacaktır. Daha sonra her bir algoritmanın en iyi varyasyonu birbiriyle karşılaştırılacaktır.

Çizelge 8.20'de DKİ algoritmasına ait 6 – 11 düğümlü tam bağlı şebeke problemleri için değişim katsayısı değerleri yer almaktadır.

Çizelge 8.21. Geliştirilen DKİ algoritmalarının 6 – 11 düğümlü tam bağlı şebeke problemleri için değişim katsayısı değerleri

Geliştirilen DKİ Algoritması	Şebeke Boyutu						Genel Ortalama
	G(6,15)	G(7,21)	G(8,28)	G(9,36)	G(10,45)	G(11,55)	
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	0,024

Çizelge 8.21'deki değerler incelendiğinde TL_DKİ algoritmasının 6 – 11 düğümlü tam bağlı şebeke problemlerinde değişim katsayısı bakımından DKİ algoritmasına göre daha iyi performans gösterdiği anlaşılmaktadır.

Çizelge 8.22'de geliştirilen DKİ algoritmalarının tam bağlı olmayan şebeke problemlerinde değişim katsayısı değerleri yer almaktadır.

Çizelge 8.22. Geliştirilen DKİ algoritmalarının tam bağlı olmayan şebeke problemlerinde değişim katsayısı değerleri

Geliştirilen DKİ Algoritması	Şebeke Boyutu			Genel Ortalama
	G(14,21)	G(16,24)	G(20,30)	
DKİ	0,001	0	0,013	0,005
TL_DKİ	0	0	0,014	0,005

Çizelge 8.22'deki değerler incelendiğinde TL_DKİ algoritmasının tam bağlı olmayan şebekelerde de değişim katsayısı değeri bakımından daha iyi performans gösterdiği anlaşılmaktadır.

Çizelge 8.23'te geliştirilen KSE algoritmalarının 6 – 11 düğümlü tam bağlı şebeke problemleri için elde edilen değişim katsayısı değerleri yer almaktadır.

Çizelge 8.23. Geliştirilen KSE algoritmalarının 6 – 11 düğümlü tam bağlı şebeke problemleri için elde edilen değişim katsayısı değerleri

Geliştirlen KSE Algoritması	Şebeke Boyutu						Genel Ortalama
	G(6,15)	G(7,21)	G(8,28)	G(9,36)	G(10,45)	G(11,55)	
Temel KSE	0,000	0,007	0,003	0,008	0,011	0,000	0,005
KSE_D	0,000	0,008	0,002	0,008	0,008	0,004	0,005
KSE_TB_1	0,000	0,004	0,001	0,008	0,010	0,008	0,005
KSE_D_TB_1	0,000	0,004	0,001	0,005	0,010	0,000	0,003
KSE_TB_{DBS}_1	0,002	0,005	0,001	0,050	0,004	0,003	0,011
KSE_D_TB_{DBS}_1	0,000	0,003	0,001	0,008	0,011	0,005	0,005
KSE_TB_2	0,002	0,002	0,000	0,002	0,002	0,000	0,002
KSE_D_TB_2	0,000	0,004	0,001	0,002	0,002	0,003	0,002
KSE_TB_{DBS}_2	0,001	0,004	0,001	0,002	0,002	0,000	0,002
KSE_D_TB_{DBS}_2	0,002	0,004	0,001	0,002	0,002	0,003	0,002

Çizelge 8.23'teki değerler incelendiğinde değişim katsayısı bakımından en iyi performansı KSE_TB_2 algoritmasının gösterdiği anlaşılmaktadır. En kötü performansı ise KSE_D algoritması göstermektedir.

Çizelge 8.24'te geliştirilen KSE algoritmalarının tam bağlı olmayan şebeke problemleri için elde edilen değişim katsayısı değerleri yer almaktadır.

Çizelge 8.24'teki değerler incelendiğinde tam bağlı olmayan şebeke problemlerinde KSE_TB_1 ve KSE_D_TB_1 algoritmaları haricindeki diğer tüm algoritmalar ortalama değişim katsayısı değerlerinin 0 olduğu görülmektedir.

Çizelge 8.24. Geliştirilen KSE tam bağlı olmayan şebeke problemleri için elde edilen değişim katsayısı değerleri

Geliştirilen KSE Algoritması	Şebeke Boyutu			Genel Ortalama
	G(14,21)	G(16,24)	G(20,30)	
Temel KSE	0,000	0,000	0,000	0,000
KSE_D	0,000	0,000	0,000	0,000
KSE_TB_1	0,000	0,000	0,082	0,028
KSE_D_TB_1	0,000	0,000	0,032	0,011
KSE_TB_{DBS}_1	0,000	0,000	0,000	0,000
KSE_D_TB_{DBS}_1	0,000	0,000	0,000	0,000
KSE_TB_2	0,000	0,000	0,000	0,000
KSE_D_TB_2	0,000	0,000	0,000	0,000
KSE_TB_{DBS}_2	0,000	0,000	0,000	0,000
KSE_D_TB_{DBS}_2	0,000	0,000	0,000	0,000

Geliştirilen KKE algoritmasının 6 – 11 düğümlü tam bağlı şebeke problemleri için elde edilen değişim katsayısı değerleri Çizelge 8.25'te ve tam bağlı olmayan şebeke problemleri için elde edilen değişim katsayısı değerleri Çizelge 8.26'da yer almaktadır.

Çizelge 8.25'teki değerler incelendiğinde değişim katsayısı genel ortalaması bakımından en iyi performansı TB_KKE_TB algoritması göstermiştir. En kötü performansı ise KKE_D_TB_{DBS} algoritması göstermiştir.

Çizelge 8.26'daki değerler incelendiğinde genel ortalama bakımından en iyi performansı TB_KKE_TB algoritması göstermiştir. En kötü performansa sahip algoritma ise $TB\_KKE_D\_TB_{DBS}$ algoritmasıdır.

Çizelge 8.25. Geliştirilen KKE algoritmalarının 6 – 11 düğümlü tam bağlı şebeke problemleri için elde edilen değişim katsayısı değerleri

Geliştirilen KKE Algoritması	Şebeke Boyutu						Genel Ortalama
	G(6,15)	G(7,21)	G(8,28)	G(9,36)	G(10,45)	G(11,55)	
TB_KKE	0,002	0,003	0,007	0,008	0,004	0,000	0,004
TB_KKE_D	0,007	0,010	0,010	0,013	0,011	0,000	0,008
KKE_TB	0,005	0,006	0,007	0,013	0,015	0,000	0,008
KKE_D_TB	0,012	0,012	0,011	0,021	0,020	0,000	0,013
KKE_TB_{DBS}	0,002	0,004	0,006	0,019	0,015	0,000	0,008
$KKE_D\_TB_{DBS}$	0,010	0,007	0,011	0,029	0,020	0,000	0,013
TB_KKE_TB	0,002	0,000	0,004	0,007	0,005	0,000	0,003
$TB\_KKE_D\_TB$	0,007	0,007	0,009	0,019	0,008	0,003	0,009
$TB\_KKE\_TB_{DBS}$	0,000	0,001	0,004	0,010	0,008	0,000	0,004
$TB\_KKE_D\_TB_{DBS}$	0,008	0,011	0,012	0,010	0,011	0,003	0,009

Çizelge 8.26. Geliştirilen KKE algoritmalarının tam bağlı olmayan şebeke problemleri için değişim katsayısı değerleri

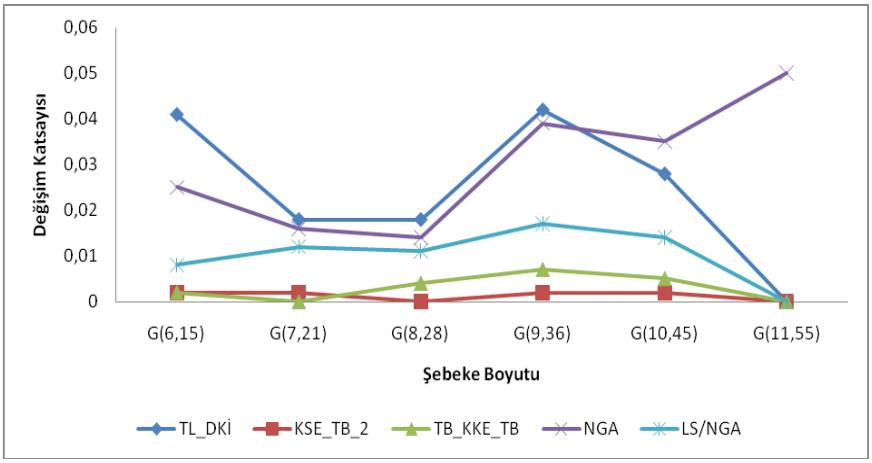
Geliştirilen KKE Algoritması	Şebeke Boyutu			Genel Ortalama
	G(14,21)	G(16,24)	G(20,30)	
TB_KKE	0,002	0,003	0,007	0,008
TB_KKE_D	0,007	0,010	0,010	0,013
KKE_TB	0,005	0,006	0,007	0,013
KKE_D_TB	0,012	0,012	0,011	0,021
KKE_TB_{DBS}	0,002	0,004	0,006	0,019
$KKE_D\_TB_{DBS}$	0,010	0,007	0,011	0,029
TB_KKE_TB	0,002	0,000	0,004	0,007
$TB\_KKE_D\_TB$	0,007	0,007	0,009	0,019
$TB\_KKE\_TB_{DBS}$	0,000	0,001	0,004	0,010
$TB\_KKE_D\_TB_{DBS}$	0,008	0,011	0,012	0,010

Her bir algoritma değişim katsayısı değerleri bakımından değerlendirildikten sonra bu algoritmaların en iyi performans gösteren varyasyonları karşılaştırılacaktır. İncelenecek olan algoritmalar TL_DKİ, KSE_TB_2 ve TB_KKE_TB algoritmaları olacaktır. Bunun yanı sıra Dengiz ve arkadaşları [16] bu çalışmada ele alınan problemin çözümünde iki farklı genetik algoritma kullanmışlardır. Bu çalışmada

geliştirilen algoritmalarla birlikte Dengiz ve arkadaşları tarafından genetik algoritma kullanılarak geliştirilen NGA ve LS/NGA algoritmalarının da değişim katsayısı değerleri de incelenmiştir. Çizelge 8.27'de 6 – 11 düğümlü tam bağlı şebeke problemleri için ortalama değişim katsayısı değerleri yer almakta ve Şekil 8.3'te bu karşılaştırmaya ait grafik yer almaktadır. Çizelge 8.28'de ise bu algoritmaların tam bağlı olmayan şebeke problemleri için ortalama değişim katsayısı değerleri yer almakta ve Şekil 8.4'te bu karşılaştırmaya ait grafik yer almaktadır.

Çizelge 8.27. Geliştirilen en iyi algoritmaların 6 – 11 tam bağlı şebeke problemleri için değişim katsayısı değerleri

Geliştirlen Algoritmalar	Şebeke Boyutu						Genel Ortalama
	G(6,15)	G(7,21)	G(8,28)	G(9,36)	G(10,45)	G(11,55)	
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
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



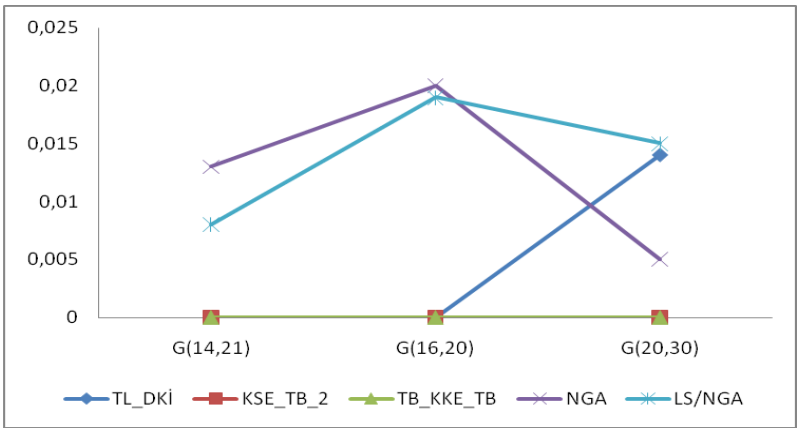
Şekil 8.3. Geliştirilen En İyi Algoritmaların 6 – 11 düğümlü tam bağlı şebeke problemleri için değişim katsayısı bakımından karşılaştırılması

Algoritmaların 6 – 11 düğümlü şebekeler için değişim katsayılarının ortalama değerleri incelendiğinde en düşük ortalamaya KSE_TB_2 algoritmasının sahip olduğu görülmektedir. Daha sonra sırasıyla TB_KKE$_D$_TB, LS/NGA, NGA ve TL_DKİ algoritmaları gelmektedir.

Çizelge 8.28 ve Şekil 8.4 incelendiğinde tam bağlı olmayan şebeke problemlerinde KSE_TB_2 ve TB_KKE_TB algoritmalarının değişim katsayısı değerleri 0 iken yalnızca TL_DKİ algoritması G(20,30) problemi için 0,014'lük bir değişim katsayısı değerine sahiptir.

Çizelge 8.28. Geliştirilen en iyi algoritmaların tam bağlı olmayan şebeke problemleri için değişim katsayısı değerleri

Geliştirilen Algoritmalar	Şebeke Boyutu			Genel Ortalama
	G(14,21)	G(16,20)	G(20,30)	
TL_DKİ	0,000	0,000	0,014	0,005
KSE_TB_2	0,000	0,000	0,000	0,000
TB_KKE_TB	0,000	0,000	0,000	0,000
NGA	0,013	0,020	0,005	0,013
LS/NGA	0,008	0,019	0,015	0,014

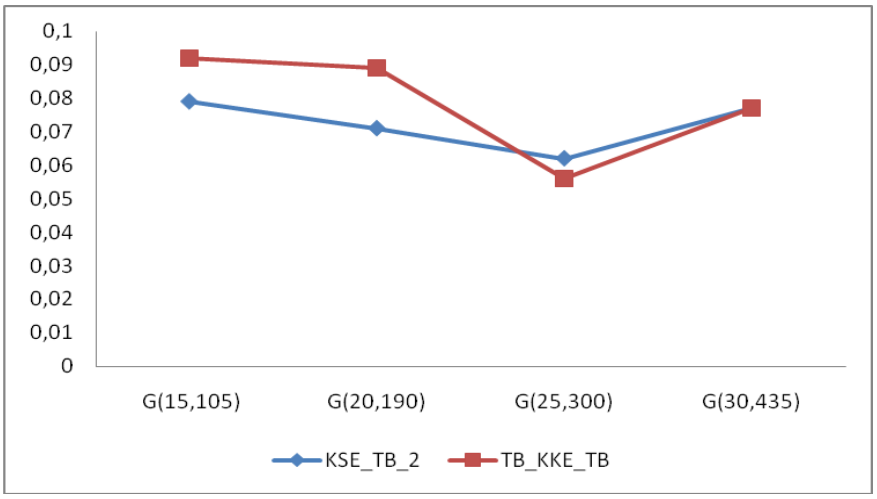


Şekil 8.4. Geliştirilen En İyi Algoritmaların tam bağlı olmayan şebeke problemleri için değişim katsayısı bakımından karşılaştırılması

Çizelge 8.29 ve Şekil 8.5'te KSE_TB_2 ve TB_KKE_TB algoritmalarının büyük boyutlu test problemlerinde elde edilen değişim katsayısı değerleri verilmiştir. Elde edilen değerler incelendiğinde 15 ve 20 düğümlü problemlerde KSE_TB_2 algoritması ortalama değişim katsayısı bakımından KKE_TB_KKE algoritmasına göre daha düşük ortalamaya sahipken, 25 ve 30 düğümlü problemlerde TB_KKE_TB algoritması ortalama değişim katsayısı bakımından KSE_TB_2 algoritmasına göre daha düşük ortalamaya sahiptir.

Çizelge 8.29. Büyük boyutlu test problemlerinde KSE_TB_2 ve TB_KKE_TB algoritmalarıyla elde edilen ortalama değişim katsayısı değerleri

Geliştirilen Algoritmalar	Şebeke Boyutu				
	G(15,105)	G(20,190)	G(25,300)	G(30,435)	G(40,780)
KSE_TB_2	0,079	0,071	0,062	0,077	0,085
TB_KKE_TB	0,092	0,089	0,056	0,077	-



Şekil 8.5. Büyük boyutlu test problemlerinde KSE_TB_2 ve TB_KKE_TB algoritmalarıyla elde edilen ortalama değişim katsayısı değerlerinin karşılaştırılması

Son olarak, KSE ve KKE ile geliştirilen algoritmalar arasında değişim katsayısına göre anlamlı bir fark olup olmadığı incelenmiştir. ANOVA testi kullanılarak yapılan analizde KSE ve KKE ile geliştirilen algoritmaların değişim katsayısına göre kendi aralarında anlamlı bir farklılığa sahip olduğu görülmüştür. Çizelge 8.30'da geliştirilen KSE algoritmaları ile gerçekleştirilen optimumdan sapma oranına göre gerçekleştirilen ANOVA testine ilişkin sonuçlar, Çizelge 8.31'de Tukey testi sonuçları ve Çizelge 8.32'de geliştirilen KSE algoritmaları ile optimumdan sapma oranına göre gerçekleştirilen ANOVA testi sonuçları ve Çizelge 8.33'te Tukey testi sonuçları yer almaktadır.

ANOVA testi %95 güvenilirlik düzeyinde gerçekleştirilmiştir ve P değeri 0,05'ten küçük olduğundan geliştirilen KSE algoritmaları arasında değişim katsayısına göre anlamlı bir fark vardır.

Çizelge 8.30. Geliştirilen KSE algoritmalarının değişim katsayısına göre elde edilen ANOVA testi sonuçları

	Kareler Toplamı	Serbestlik Derecesi	Ortalama Kare	F Değeri	P Değeri
Gruplar arası	0,007	9	0,001	3,244	0,001
Grup içi	0,174	780	0,000		
Toplam	0,180	789			

Çizelge 8.31. Geliştirilen KSE algoritmalarının değişim katsayısına göre Tukey Testi ile karşılaştırılmasına ait sonuçlar

Grup (I)	Grup (J)	Ortalama Fark (I-J)	Std. Hata	Anlamlılık	%95 Güven Aralığı	
		Alt Sınır	Üst Sınır	Alt Sınır	Üst Sınır	Alt Sınır
KSE	KSE_D	0,0004	0,002	1,000	-0,007	0,008
	KSE_TB_1	0,0002	0,002	1,000	-0,007	0,008
	KSE_D_TB_1	0,0013	0,002	1,000	-0,006	0,009
	KSE_TB_{DBS}_1	-0,006	0,002	0,226	-0,014	0,001
	KSE_D_TB_{DBS}_1	0,001	0,002	1,000	-0,007	0,008
	KSE_TB_2	0,004	0,002	0,831	-0,004	0,011
	KSE_D_TB_2	0,004	0,002	0,874	-0,004	0,011
	KSE_TB_{DBS}_2	0,004	0,002	0,853	-0,004	0,011
	KSE_D_TB_{DBS}_2	0,004	0,002	0,898	-0,004	0,011
KSE_D	KSE_TB_1	-0,0002	0,002	1,000	-0,008	0,007
	KSE_D_TB_1	0,001	0,002	1,000	-0,007	0,008
	KSE_TB_{DBS}_1	-0,007	0,002	0,152	-0,014	0,001
	KSE_D_TB_{DBS}_1	0,001	0,002	1,000	-0,007	0,008
	KSE_TB_2	0,003	0,002	0,908	-0,004	0,011
	KSE_D_TB_2	0,003	0,002	0,937	-0,004	0,011
	KSE_TB_{DBS}_2	0,003	0,002	0,923	-0,004	0,011
	KSE_D_TB_{DBS}_2	0,003	0,002	0,952	-0,004	0,011
KSE_TB_1	KSE_D_TB_1	0,001	0,002	1,000	-0,006	0,009
	KSE_TB_{DBS}_1	-0,006	0,002	0,192	-0,014	0,001
	KSE_D_TB_{DBS}_1	0,001	0,002	1,000	-0,007	0,008
	KSE_TB_2	0,004	0,002	0,867	-0,004	0,011
	KSE_D_TB_2	0,004	0,002	0,904	-0,004	0,011
	KSE_TB_{DBS}_2	0,004	0,002	0,886	-0,004	0,011
	KSE_D_TB_{DBS}_2	0,003	0,002	0,924	-0,004	0,011
KSE_D_TB_1	KSE_TB_{DBS}_1	-0,008	0,002	0,055	-0,015	0,000
	KSE_D_TB_{DBS}_1	-0,0004	0,002	1,000	-0,008	0,007
	KSE_TB_2	0,003	0,002	0,987	-0,005	0,010
	KSE_D_TB_2	0,002	0,002	0,993	-0,005	0,010
	KSE_TB_{DBS}_2	0,003	0,002	0,990	-0,005	0,010
	KSE_D_TB_{DBS}_2	0,002	0,002	0,996	-0,005	0,010
KSE_TB_{DBS}_1	KSE_D_TB_{DBS}_1	0,007	0,002	0,092	-0,001	0,015
	KSE_TB_2	0,010(*)	0,002	0,001	0,003	0,018
	KSE_TB_{DBS}_2	0,010(*)	0,002	0,001	0,002	0,017
	KSE_D_TB_{DBS}_2	0,001(*)	0,002	0,002	0,002	0,017
KSE_D_TB_{DBS}_1	KSE_TB_2	0,003	0,002	0,962	-0,005	0,011
	KSE_D_TB_2	0,003	0,002	0,977	-0,005	0,010
	KSE_TB_{DBS}_2	0,003	0,002	0,970	-0,005	0,010
	KSE_D_TB_{DBS}_2	0,003	0,002	0,984	-0,005	0,010
KSE_TB_2	KSE_D_TB_2	-0,0002	0,002	1,000	-0,008	0,007
	KSE_TB_{DBS}_2	-0,0001	0,002	1,000	-0,008	0,007

Çizelge 8.31. (Devam) Geliştirilen KSE algoritmalarının değişim katsayısına göre Tukey Testi ile karşılaştırılmasına ait sonuçlar

Grup (I)	Grup (J)	Ortalama Fark (I-J)	Std. Hata	Anlamlılık	%95 Güven Aralığı	
		Alt Sınır	Üst Sınır	Alt Sınır	Üst Sınır	Alt Sınır
	$KSE_D\_TB_{DBS}\_2$	-0,0004	0,002	1,000	-0,008	0,007
$KSE_D\_TB\_2$	$KSE\_TB_{DBS}\_2$	0,0001	0,002	1,000	-0,007	0,008
	$KSE_D\_TB_{DBS}\_2$	-0,0001	0,002	1,000	-0,008	0,007
$KSE\_TB_{DBS}\_2$	$KSE_D\_TB_{DBS}\_2$	-0,0003	0,002	1,000	-0,008	0,007

* Ortalama farklar 0,05 düzeyinde anlamlıdır.

Çizelge 8.32. Geliştirilen KKE algoritmalarının değişim katsayısına göre elde edilen ANOVA testi sonuçları

	Kareler Toplamı	Serbestlik Derecesi	Ortalama Kare	F Değeri	P Değeri
Gruplar arası	0,011	9	0,001	5,765	0,000
Grup içi	0,165	780	0,000		
Toplam	0,176	789			

ANOVA testi %95 güvenilirlik düzeyinde gerçekleştirilmiştir ve P değeri 0,05'ten küçük olduğundan geliştirilen KKE algoritmaları arasında değişim katsayısına göre anlamlı bir fark vardır.

Çizelge 8.33. Geliştirilen KSE algoritmalarının değişim katsayısına göre Tukey Testi ile karşılaştırılmasına ait sonuçlar

Grup (I)	Grup (J)	Ortalama Fark (I-J)	Std. Hata	Anlamlılık	%95 Güven Aralığı	
		Alt Sınır	Üst Sınır	Alt Sınır	Üst Sınır	Alt Sınır
TB_KKE	TB_KKE_D	-0,005	0,002	0,470	-0,012	0,002
	KKE_TB	-0,004	0,002	0,689	-0,012	0,003
	KKE_D_TB	-0,010(*)	0,002	0,001	-0,017	-0,002
	KKE_TB_{DBS}	-0,004	0,002	0,663	-0,012	0,003
	$KKE_D\_TB_{DBS}$	-0,010(*)	0,002	0,001	-0,017	-0,003
	TB_KKE_TB	0,002	0,002	1,000	-0,006	0,009
	$TB\_KKE_D\_TB$	-0,005	0,002	0,501	-0,012	0,002
	$TB\_KKE\_TB_{DBS}$	0,0002	0,002	1,000	-0,007	0,008
	$TB\_KKE_D\_TB_{DBS}$	-0,006	0,002	0,330	-0,013	0,002
TB_KKE_D	KKE_TB	0,001	0,002	1,000	-0,007	0,008
	KKE_D_TB	-0,005	0,002	0,555	-0,012	0,003
	KKE_TB_{DBS}	0,001	0,002	1,000	-0,007	0,008
	$KKE_D\_TB_{DBS}$	-0,005	0,002	0,462	-0,012	0,002
	TB_KKE_TB	0,007	0,002	0,131	-0,001	0,014
	$TB\_KKE_D\_TB$	0,0001	0,002	1,000	-0,007	0,007
	$TB\_KKE\_TB_{DBS}$	0,005	0,002	0,417	-0,002	0,013
	$TB\_KKE_D\_TB_{DBS}$	-0,0005	0,002	1,000	-0,008	0,007
KKE_TB	KKE_D_TB	-0,0055	0,002	0,343	-0,013	0,002
	KKE_TB_{DBS}	-0,0001	0,002	1,000	-0,007	0,007

Çizelge 8.33. (Devam) Geliştirilen KSE algoritmalarının değişim katsayısına göre Tukey Testi ile karşılaştırılmasına ait sonuçlar

Grup (I)	Grup (J)	Ortalama Fark (I-J)	Std. Hata	Anlamlılık	%95 Güven Aralığı	
		Alt Sınır	Üst Sınır	Alt Sınır	Üst Sınır	Alt Sınır
	$KKE_{D}\_TB_{DBS}$	-0,0058	0,002	0,267	-0,013	0,002
	TB_KKE_TB	0,0058	0,002	0,264	-0,002	0,013
	$TB\_KKE_{D}\_TB$	-0,00062	0,002	1,000	-0,008	0,007
	$TB\_KKE\_TB_{DBS}$	0,00451	0,002	0,636	-0,003	0,012
	$TB\_KKE_{D}\_TB_{DBS}$	-0,00122	0,002	1,000	-0,009	0,006
KKE_{D}_TB	KKE_TB_{DBS}	0,0054	0,002	0,367	-0,002	0,013
	$KKE_{D}\_TB_{DBS}$	-0,0003	0,002	1,000	-0,008	0,007
	TB_KKE_TB	0,0113(*)	0,002	0,000	0,004	0,019
	$TB\_KKE_{D}\_TB$	0,0049	0,002	0,524	-0,002	0,012
	$TB\_KKE\_TB_{DBS}$	0,010(*)	0,002	0,001	0,003	0,017
	$TB\_KKE_{D}\_TB_{DBS}$	0,0043	0,002	0,704	-0,003	0,012
KKE_TB_{DBS}	$KKE_{D}\_TB_{DBS}$	-0,00571	0,002	0,288	-0,013	0,002
	TB_KKE_TB	0,0059	0,002	0,244	-0,001	0,013
	$TB\_KKE_{D}\_TB$	-0,0005	0,002	1,000	-0,008	0,007
	$TB\_KKE\_TB_{DBS}$	0,0046	0,002	0,610	-0,003	0,012
	$TB\_KKE_{D}\_TB_{DBS}$	-0,00113	0,002	1,000	-0,008	0,006
$KKE_{D}\_TB_{DBS}$	TB_KKE_TB	0,01161(*)	0,002	0,000	0,004	0,019
	$TB\_KKE_{D}\_TB$	0,0052	0,002	0,432	-0,002	0,013
	$TB\_KKE\_TB_{DBS}$	0,0103(*)	0,002	0,000	0,003	0,018
	$TB\_KKE_{D}\_TB_{DBS}$	0,0046	0,002	0,613	-0,003	0,012
TB_KKE_TB	$TB\_KKE_{D}\_TB$	-0,0064	0,002	0,146	-0,014	0,001
	$TB\_KKE\_TB_{DBS}$	-0,0013	0,002	1,000	-0,009	0,006
	$TB\_KKE_{D}\_TB_{DBS}$	-0,0070	0,002	0,074	-0,014	0,000
$TB\_KKE_{D}\_TB$	$TB\_KKE\_TB_{DBS}$	0,00513	0,002	0,447	-0,002	0,012
	$TB\_KKE_{D}\_TB_{DBS}$	-0,0006	0,002	1,000	-0,008	0,007
$TB\_KKE\_TB_{DBS}$	$TB\_KKE_{D}\_TB_{DBS}$	-0,006	0,002	0,285	-0,013	0,002

* Ortalama farklar 0,05 düzeyinde anlamlıdır.

8.4.3. Geliştirilen algoritmaların çözüm zamanına göre değerlendirilmesi

Çalışmanın bu bölümünde geliştirilen algoritmaların performansları çözüm zamanları açısından değerlendirilecektir. Çözüm zamanına ilişkin değerlendirmeler sırasıyla DKİ, KSE ve KKE algoritmaları için yapılacaktır. Daha sonra her bir algoritmanın en iyi varyasyonu birbiriyle karşılaştırılacaktır.

Çizelge 8.34'te geliştirilen DKİ algoritmalarının tam bağlı ve tam bağlı olmayan şebeke problemlerini çözüm sürelerine ilişkin veriler yer almaktadır. Bu çizelgedeki değerler incelendiğinde TL_DKİ algoritmasının ortalama olarak daha kısa sürede çözüme gittiği anlaşılmaktadır.

Çizelge 8.35'te geliştirilen KSE algoritmalarının tam bağlı ve tam bağlı olmayan şebeke problemlerini çözüm sürerleri yer almaktadır. Bu çizelgedeki değerler incelendiğinde $KSE_D\_TB_{DBS}\_1$ algoritmasının en kısa sürede, $KSE_D\_TB\_1$ algoritmasının ortalama olarak en uzun sürede çözüme ulaştığı görülmektedir. Ayrıca tavlama benzetimi algoritmasının başlangıç sıcaklığının her iterasyonda azaltılması algoritmayı hızlandırmaktadır.

Çizelge 8.36'da geliştirilen KKE algoritmalarının tam bağlı ve tam bağlı olmayan şebeke problemlerinin çözüm sürelerine ait değerler yer almaktadır. Bu değerler incelendiğinde KKE_TB_{DBS} algoritmasının ortalama olarak en kısa sürede, $TB\_KKE_D\_TB$ algoritmasının da en uzun sürede çözüme ulaştığı görülmektedir.

Çizelge 8.37 ve Şekil 8.6 incelendiğinde DKİ'nin ortalama olarak en kısa sürede çözüme ulaştığı, daha sonra $KSE_D\_TB_{DBS}\_1$ algoritmasının ve sonuncu olarak KKE_TB_{DBS} algoritmasının geldiği görülmektedir.

Çizelge 8.38 ve Şekil 8.7'de büyük boyutlu problemler için KSE_TB_2 algoritması ve KKE_TB_KKE algoritmasının çözüm zamanı değerleri yer almaktadır. Söz konusu çizelge ve şekil incelendiğinde KSE_TB_2 algoritmasının tüm problem boyutlarında TB_KKE_TB algoritmasına göre çözüm zamanı bakımından daha iyi performans gösterdiği görülmektedir.

Çizelge 8.34. Geliştirilen DKİ algoritmalarının tam bağlı ve tam bağlı olmayan şebeke problemlerini çözüm zamanı

Geliştirilen DKİ Algoritması	Şebeke Boyutu									Genel Ortalama
	G(6,15)	G(7,21)	G(8,28)	G(9,36)	G(10,45)	G(11,55)	G(14,21)	G(16,24)	G(20,30)	
DKİ	0,174	0,540	0,707	1,460	1,734	5,092	38,462	20,219	24,328	10,302
TL_DKİ	0,172	0,400	0,588	1,125	2,238	3,164	29,462	12,119	14,819	7,121

Çizelge 8.35. Geliştirilen KSE algoritmalarının tam bağlı ve tam bağlı olmayan şebeke problemlerini çözüm zamanı

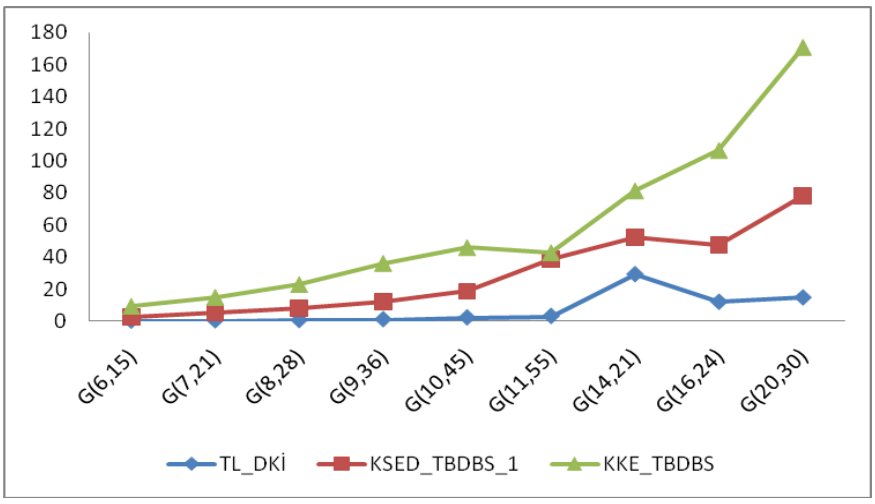
Geliştirilen KSE Algoritması	Şebeke Boyutu									Genel Ortalama
	G(6,15)	G(7,21)	G(8,28)	G(9,36)	G(10,45)	G(11,55)	G(14,21)	G(16,24)	G(20,30)	
Temel KSE	3,099	7,165	6,408	7,680	8,452	11,684	212,019	199,263	260,659	79,603
KSE_D	3,335	6,970	5,875	7,810	8,868	11,628	210,999	198,304	259,405	79,244
KSE_TB_1	12,395	27,889	52,898	103,867	199,131	436,106	230,978	215,603	353,947	181,424
KSE_D_TB_1	11,853	30,705	59,809	118,466	230,309	516,094	235,578	215,603	353,947	196,929
KSE_TB_{DBS}_1	3,056	6,984	10,146	14,900	19,423	38,255	56,580	51,782	149,709	38,982
KSE_D_TB_{DBS}_1	2,726	5,520	8,392	12,502	18,817	38,700	52,280	47,457	78,149	29,394
KSE_TB_2	10,291	17,462	16,373	20,141	14,195	18,512	81,836	82,657	173,749	48,357
KSE_D_TB_2	8,184	15,836	16,063	19,921	15,348	23,647	87,734	101,106	169,943	50,865
KSE_TB_{DBS}_2	7,658	16,793	16,417	18,782	12,514	16,114	62,389	68,441	115,038	37,127
KSE_D_TB_{DBS}_2	7,464	14,740	14,478	16,277	10,490	20,817	76,588	86,371	125,920	41,461

Çizelge 8.36. Geliştirilen KKE algoritmalarının tam bağlı ve tam bağlı olmayan şebeke problemlerini çözüm zamanı

Geliştirilen KKE Algoritması	Şebeke Boyutu									Genel Ortalama
	G(6,15)	G(7,21)	G(8,28)	G(9,36)	G(10,45)	G(11,55)	G(14,21)	G(16,24)	G(20,30)	
TB_KKE	11,295	19,501	25,459	41,131	61,516	76,152	90,719	109,578	222,578	73,103
TB_KKE$_D$	12,373	18,737	27,271	48,397	75,382	73,152	95,470	125,578	222,578	77,660
KKE_TB	9,711	15,820	22,920	36,184	47,695	44,827	84,170	105,278	180,578	60,798
KKE$_D$_TB	11,600	19,674	26,481	41,795	59,175	46,827	87,170	109,278	274,578	75,175
KKE_TB$_{DBS}$	9,252	14,836	22,857	36,043	45,807	42,645	81,170	106,278	170,578	58,830
KKE$_D$_TB$_{DBS}$	12,286	17,967	26,096	28,430	62,292	48,645	80,270	101,978	167,578	60,616
TB_KKE_TB	20,422	29,584	41,943	67,183	88,292	95,219	113,398	136,973	178,223	85,693
TB_KKE$_D$_TB	21,935	34,848	47,229	79,605	119,919	152,813	145,150	175,325	356,125	125,883
TB_KKE_TB$_{DBS}$	20,983	31,106	41,427	65,492	92,861	85,228	127,006	153,409	199,609	90,791
TB_KKE$_D$_TB$_{DBS}$	22,538	36,403	48,860	84,297	119,034	88,232	130,010	156,413	202,613	98,711

Çizelge 8.37. Geliştirilen en iyi algoritmaların tam bağlı ve tam bağlı olmayan şebeke problemlerini çözüm zamanı değerleri

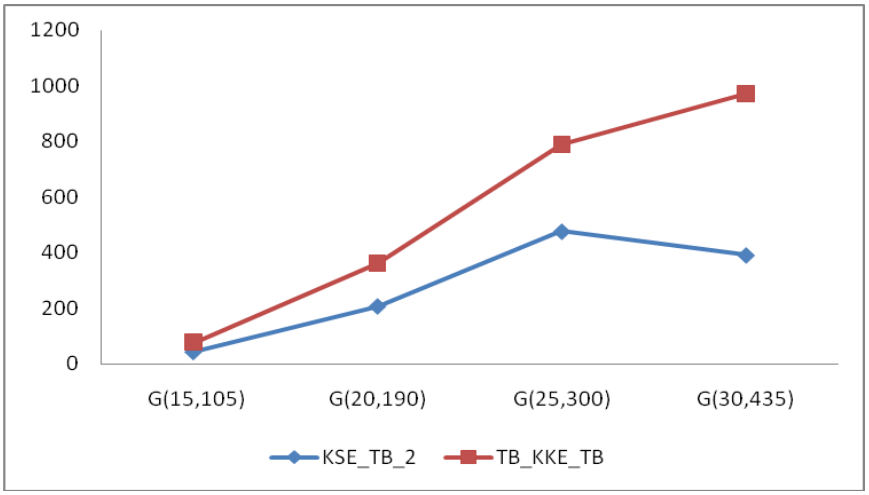
Geliştirilen Algoritmalar	Şebeke Boyutu									Genel Ortalama
	G(6,15)	G(7,21)	G(8,28)	G(9,36)	G(10,45)	G(11,55)	G(14,21)	G(16,24)	G(20,30)	
TL_DKİ	0,172	0,400	0,588	1,125	2,238	3,164	29,462	12,119	14,819	7,121
KSE$_D$_TB$_{DBS}$_1	2,726	5,520	8,392	12,502	18,817	38,700	52,280	47,457	78,149	29,394
KKE_TB$_{DBS}$	9,252	14,836	22,857	36,043	45,807	42,645	81,170	106,278	170,578	58,830



Şekil 8.6. Geliştirilen en iyi algoritmaların tam bağlı ve tam bağlı olmayan şebeke problemlerini çözüm zamanı değerleri

Çizelge 8.38. Büyük boyutlu test problemlerinde KSE_TB_2 ve TB_KKE_TB algoritmalarıyla elde edilen ortalama çözüm zamanı eğerleri

Geliştirilen Algoritmalar	Şebeke Boyutu				
	G(15,105)	G(20,190)	G(25,300)	G(30,435)	G(40,780)
KSE_TB_2	43,235	207,818	477,040	391,488	1745,781
TB_KKE_TB	76,121	361,501	788,590	971,616	-



Şekil 8.7. Büyük boyutlu test problemlerinde KSE_TB_2 ve TB_KKE_TB algoritmalarıyla elde edilen ortalama çözüm zamanı değerlerinin karşılaştırılması

Son olarak, KSE ve KKE ile geliştirilen algoritmaların kendi aralarında çözüm zamanına göre anlamlı bir fark olup olmadığı incelenmiştir. ANOVA testi kullanılarak yapılan analizde KSE ve KKE ile geliştirilen algoritmaların çözüm zamanına göre kendi aralarında anlamlı bir farklılığa sahip olduğu görülmüştür. Çizelge 8.39'da geliştirilen KSE algoritmaları ile gerçekleştirilen optimumdan sapma oranına göre gerçekleştirilen ANOVA testine ilişkin sonuçlar, Çizelge 8.40'ta Tukey testi sonuçları ve Çizelge 8.41'de geliştirilen KSE algoritmaları ile optimumdan sapma oranına göre gerçekleştirilen ANOVA testi sonuçları ve Çizelge 8.42'de Tukey testi sonuçları yer almaktadır.

Çizelge 8.39. Geliştirilen KSE algoritmalarının çözüm zamanına göre elde edilen ANOVA testi sonuçları

	Kareler Toplamı	Serbestlik Derecesi	Ortalama Kare	F Değeri	P Değeri
Gruplar arası	0,010	9	0,001	4,179	0,000
Grup içi	0,199	780	0,000		
Toplam	0,208	789			

ANOVA testi %95 güvenilirlik düzeyinde gerçekleştirilmiştir ve P değeri 0,05'ten küçük olduğundan geliştirilen KSE algoritmaları arasında çözüm zamanına göre anlamlı bir fark vardır.

Çizelge 8.40. Geliştirilen KSE algoritmalarının çözüm zamanına göre Tukey Testi ile karşılaştırılmasına ait sonuçlar

Grup (I)	Grup (J)	Ortalama Fark (I-J)	Std. Hata	Anlamlılık	%95 Güven Aralığı	
		Alt Sınır	Üst Sınır	Alt Sınır	Üst Sınır	Alt Sınır
KSE	KSE_D	0,001	0,003	1,000	-0,007	0,009
	KSE_TB_1	0,001	0,003	1,000	-0,007	0,009
	KSE_D_TB_1	-0,003	0,003	1,000	-0,008	0,008
	KSE_TB_{DBS}_1	-0,0101(*)	0,003	0,003	-0,018	-0,002
	KSE_D_TB_{DBS}_1	-0,0002	0,003	1,000	-0,008	0,008
	KSE_TB_2	0,002	0,003	0,996	-0,006	0,010
	KSE_D_TB_2	0,002	0,003	0,999	-0,006	0,010
	KSE_TB_{DBS}_2	0,002	0,003	0,999	-0,006	0,010
	KSE_D_TB_{DBS}_2	0,002	0,003	0,995	-0,006	0,010
KSE_D	KSE_TB_1	0,000	0,003	1,000	-0,008	0,008
	KSE_D_TB_1	-0,001	0,003	1,000	-0,009	0,007
	KSE_TB_{DBS}_1	-0,011(*)	0,003	0,001	-0,019	-0,003
	KSE_D_TB_{DBS}_1	-0,001	0,003	1,000	-0,009	0,007
	KSE_TB_2	0,002	0,003	1,000	-0,006	0,010
	KSE_D_TB_2	0,001	0,003	1,000	-0,007	0,009

Çizelge 8.40. (Devam) Geliştirilen KSE algoritmalarının çözüm zamanına göre Tukey Testi ile karşılaştırılmasına ait sonuçlar

Grup (I)	Grup (J)	Ortalama Fark (I-J)	Std. Hata	Anlamlılık	%95 Güven Aralığı	
		Alt Sınır	Üst Sınır	Alt Sınır	Üst Sınır	Alt Sınır
	$KSE\_TB_{DBS}\_2$	0,001	0,003	1,000	-0,007	0,009
	$KSE_D\_TB_{DBS}\_2$	0,002	0,003	1,000	-0,006	0,010
KSE_TB_1	KSE_D_TB_1	-0,001	0,003	1,000	-0,009	0,007
	$KSE\_TB_{DBS}\_1$	-0,011(*)	0,003	0,001	-0,019	-0,003
	$KSE_D\_TB_{DBS}\_1$	-0,001	0,003	1,000	-0,009	0,007
	KSE_TB_2	0,002	0,003	1,000	-0,006	0,010
	KSE_D_TB_2	0,001	0,003	1,000	-0,007	0,009
	$KSE\_TB_{DBS}\_2$	0,001	0,003	1,000	-0,007	0,009
	$KSE_D\_TB_{DBS}\_2$	0,002	0,003	1,000	-0,006	0,010
KSE_D_TB_1	$KSE\_TB_{DBS}\_1$	-0,010(*)	0,003	0,005	-0,018	-0,002
	$KSE_D\_TB_{DBS}\_1$	0,000	0,003	1,000	-0,008	0,008
	KSE_TB_2	0,003	0,003	0,988	-0,005	0,011
	KSE_D_TB_2	0,002	0,003	0,997	-0,006	0,010
	$KSE\_TB_{DBS}\_2$	0,002	0,003	0,997	-0,006	0,010
	$KSE_D\_TB_{DBS}\_2$	0,003	0,003	0,987	-0,005	0,011
$KSE\_TB_{DBS}\_1$	$KSE_D\_TB_{DBS}\_1$	0,010(*)	0,003	0,004	0,002	0,018
	KSE_TB_2	0,0124(*)	0,003	0,000	0,004	0,020
	KSE_D_TB_2	0,0120(*)	0,003	0,000	0,004	0,020
	$KSE\_TB_{DBS}\_2$	0,0120(*)	0,003	0,000	0,004	0,020
	$KSE_D\_TB_{DBS}\_2$	0,0125(*)	0,003	0,000	0,004	0,021
$KSE_D\_TB_{DBS}\_1$	KSE_TB_2	0,003	0,003	0,992	-0,006	0,011
	KSE_D_TB_2	0,002	0,003	0,998	-0,006	0,010
	$KSE\_TB_{DBS}\_2$	0,002	0,003	0,998	-0,006	0,010
	$KSE_D\_TB_{DBS}\_2$	0,003	0,003	0,991	-0,005	0,011
KSE_TB_2	KSE_D_TB_2	0,000	0,003	1,000	-0,008	0,008
	$KSE\_TB_{DBS}\_2$	0,000	0,003	1,000	-0,008	0,008
	$KSE_D\_TB_{DBS}\_2$	0,000	0,003	1,000	-0,008	0,008
KSE_D_TB_2	$KSE\_TB_{DBS}\_2$	0,000	0,003	1,000	-0,008	0,008
	$KSE_D\_TB_{DBS}\_2$	0,001	0,003	1,000	-0,008	0,009
$KSE\_TB_{DBS}\_2$	$KSE_D\_TB_{DBS}\_2$	0,001	0,003	1,000	-0,008	0,009

* Ortalama farklar 0,05 düzeyinde anlamlıdır.

Çizelge 8.41. Geliştirilen KKE algoritmalarının çözüm zamanına göre elde edilen ANOVA testi sonuçları

	Kareler Toplamı	Serbestlik Derecesi	Ortalama Kare	F Değeri	P Değeri
Gruplar arası	0,010	9	0,001	5,964	0,000
Grup içi	0,150	780	0,000		
Toplam	0,160	789			

ANOVA testi %95 güvenilirlik düzeyinde gerçekleştirilmiştir ve P değeri 0,05'ten küçük olduğundan geliştirilen KKE algoritmaları arasında çözüm zamanına göre anlamlı bir fark vardır.

Çizelge 8.42. Geliştirilen KKE algoritmalarının çözüm zamanına göre Tukey Testi ile karşılaştırılmasına ait sonuçlar

Grup (I)	Grup (J)	Ortalama Fark (I-J)	Std. Hata	Anlamlılık	%95 Güven Aralığı	
		Alt Sınır	Üst Sınır	Alt Sınır	Üst Sınır	Alt Sınır
TB_KKE	TB_KKE_D	-0,0057	0,002	0,226	-0,013	0,001
	KKE_TB	-0,0039	0,002	0,758	-0,011	0,003
	KKE_D_TB	-0,0107(*)	0,002	0,000	-0,018	-0,004
	KKE_TB_{DBS}	-0,0037	0,002	0,811	-0,011	0,003
	$KKE_D\_TB_{DBS}$	-0,0083(*)	0,002	0,007	-0,015	-0,001
	TB_KKE_TB	0,0010	0,002	1,000	-0,006	0,008
	$TB\_KKE_D\_TB$	-0,0061	0,002	0,151	-0,013	0,001
	$TB\_KKE\_TB_{DBS}$	-0,00003	0,002	1,000	-0,007	0,007
	$TB\_KKE_D\_TB_{DBS}$	-0,0058	0,002	0,200	-0,013	0,001
TB_KKE_D	KKE_TB	0,0018	0,002	0,998	-0,005	0,009
	KKE_D_TB	-0,0050	0,002	0,410	-0,012	0,002
	KKE_TB_{DBS}	0,0020	0,002	0,996	-0,005	0,009
	$KKE_D\_TB_{DBS}$	-0,0026	0,002	0,974	-0,010	0,004
	TB_KKE_TB	0,0067	0,002	0,070	0,000	0,014
	$TB\_KKE_D\_TB$	-0,0004	0,002	1,000	-0,007	0,007
	$TB\_KKE\_TB_{DBS}$	0,0057	0,002	0,232	-0,001	0,013
	$TB\_KKE_D\_TB_{DBS}$	-0,0001	0,002	1,000	-0,007	0,007
KKE_TB	KKE_D_TB	-0,0068	0,002	0,064	-0,014	0,000
	KKE_TB_{DBS}	0,0002	0,002	1,000	-0,007	0,007
	$KKE_D\_TB_{DBS}$	-0,0044	0,002	0,592	-0,011	0,003
	TB_KKE_TB	0,0049	0,002	0,434	-0,002	0,012
	$TB\_KKE_D\_TB$	-0,0022	0,002	0,992	-0,009	0,005
	$TB\_KKE\_TB_{DBS}$	0,0039	0,002	0,765	-0,003	0,011
	$TB\_KKE_D\_TB_{DBS}$	-0,0019	0,002	0,997	-0,009	0,005
KKE_D_TB	KKE_TB_{DBS}	0,007(*)	0,002	0,049	0,000	0,014
	$KKE_D\_TB_{DBS}$	0,0024	0,002	0,987	-0,005	0,009
	TB_KKE_TB	0,0117(*)	0,002	0,000	0,005	0,019
	$TB\_KKE_D\_TB$	0,0046	0,002	0,534	-0,002	0,012
	$TB\_KKE\_TB_{DBS}$	0,0107(*)	0,002	0,000	0,004	0,018
	$TB\_KKE_D\_TB_{DBS}$	0,0049	0,002	0,450	-0,002	0,012
KKE_TB_{DBS}	$KKE_D\_TB_{DBS}$	-0,0046	0,002	0,526	-0,012	0,002
	TB_KKE_TB	0,0047	0,002	0,498	-0,002	0,012
	$TB\_KKE_D\_TB$	-0,0024	0,002	0,985	-0,009	0,005
	$TB\_KKE\_TB_{DBS}$	0,0037	0,002	0,817	-0,003	0,011
	$TB\_KKE_D\_TB_{DBS}$	-0,0021	0,002	0,994	-0,009	0,005
$KKE_D\_TB_{DBS}$	TB_KKE_TB	0,009(*)	0,002	0,001	0,002	0,016
	$TB\_KKE_D\_TB$	0,0022	0,002	0,992	-0,005	0,009
	$TB\_KKE\_TB_{DBS}$	0,008(*)	0,002	0,007	0,001	0,015
	$TB\_KKE_D\_TB_{DBS}$	0,0025	0,002	0,981	-0,005	0,009
TB_KKE_TB	$TB\_KKE_D\_TB$	-0,0071(*)	0,002	0,042	-0,014	0,000
	$TB\_KKE\_TB_{DBS}$	-0,0011	0,002	1,000	-0,008	0,006
	$TB\_KKE_D\_TB_{DBS}$	-0,0069	0,002	0,060	-0,014	0,000
$TB\_KKE_D\_TB$	$TB\_KKE\_TB_{DBS}$	0,0060	0,002	0,155	-0,001	0,013
	$TB\_KKE_D\_TB_{DBS}$	0,0003	0,002	1,000	-0,007	0,007
$TB\_KKE\_TB_{DBS}$	$TB\_KKE_D\_TB_{DBS}$	-0,0058	0,002	0,205	-0,013	0,001

* Ortalama farklar 0,05 düzeyinde anlamlıdır.

9. SONUÇ VE ÖNERİLER

Bu tezde, NP-zor problem sınıfına giren güvenilirlik kısıtı altından minimum maliyetli haberleşme şebekelerinin tasarımı problemi ele alınmıştır. Bu problemde ana haberleşme şebekelerinin tasarımıyla ilgilenilmiştir. Ayrıca bu problemde elde edilen aday şebekelerin güvenilirliğini değerlendirmede Monte Carlo benzetimi kullanılmıştır. Bu problemin çözümünde 3 farklı genel amaçlı sezgisel ve bunların karma formları kullanılmıştır. Bu genel amaçlı sezgiseller değişken komşu iniş, Kuş Sürüsü Eniyilemesi ve Karınca Kolonisi Eniyilemesidir. Değişken Komşu İniş algoritmasında tabu listesi, Kuş Sürüsü Eniyilemesi ve karınca kolonisi eniyilemesi algoritmalarında ise bir stokastik arama yöntemi olan tavlama benzetimi kullanılarak karma formlar elde edilmiştir. Elde edilen 22 farklı algoritma toplam 154 test problemi üzerinde denenmiştir.

Yapılan denemeler sonucu elde edilen çözümler algoritmaların etkinliklerini değerlendirmede kullanılmıştır. Algoritmaların etkinlikleri en iyi değerden sapma oranı, değişim katsayısı ve çözüm zamanı açısından değerlendirilmiştir. Bölüm 8'de ayrıntılı değerlendirmeler yer almaktadır. Burada ise algoritmalar genel olarak değerlendirilecektir.

Çalışmanın sonucunda görülmüştür ki ele alınan 3 temel algoritmadan genel olarak en iyisi Kuş Sürüsü Eniyilemesi algoritmasıdır. Daha sonra ise Karınca Kolonisi Eniyilemesi ve Değişken Komşu İniş algoritması gelmektedir. Bunu yanında geliştirilen algoritmalar değişim katsayısı bakımından Dengiz ve arkadaşları [16] tarafından geliştirilen genetik algoritma sonuçlarıyla karşılaştırılmıştır ve bu çalışmada geliştirilen Kuş Sürüsü Eniyilemesi ve Karınca Kolonisi Eniyilemsi algoritmaları genetik algoritmadan daha iyi performans göstermiştir. Genetik algoritma ise değişken komşu iniş algoritmasından daha iyi performans vermiştir.

Kuş Sürüsü Eniyilemesi ve Karınca Kolonisi Eniyilemesi algoritmaları sürü zekası adı verilen sınıfta yer almaktadırlar. Yapılan çalışmada görülmüştür ki algoritmaların parametrelerinin doğru şekilde seçilmesi algoritmaların etkinliğini artırmaktadır.

Ayrıca bu algoritmalarla karma şekilde diğer genel amaçlı sezgiseller kolaylıkla kullanılabilmekte ve algoritmaların etkinliklerinin artırılmasına yardımcı olmaktadırlar. Her iki algoritmada çeşitli parametrelere dinamiklik özelliği katılarak da denemeler yapılmıştır. Yapılan denemeler sonucunda parametrelerin dinamikleştirilmesinin çözüm kalitesinin iyileşmesinde önemli bir etkiye sahip olmadığı görülmüştür.

Değişken Komşu İniş algoritmasının etkinliğinin ise bu çalışmada kullanılan diğer iki genel amaçlı sezgisele göre daha düşük olduğu görülmüştür. Bu durum seçilen komşuluk yapılarından kaynaklanmaktadır.

İleride bu konuda yapılacak olan çalışmalarda Değişken Komşu İniş algoritması için farklı komşuluk yapıları kullanılabilir ve Değişken Komşu İniş algoritmasının farklı türlerinin bu problem üzerinde nasıl sonuçlar verdiği gözlenebilir. Bunun yanı sıra bu algoritmalarla birlikte daha başka genel amaçlı sezgiseller karma biçimde kullanılarak farklı çalışmalar yapılabilir.

KAYNAKLAR

1. Amato, V., "CISCO Networking Essentials For Educational Institutions 1st ed.", ***CISCO Press,*** Indianapolis, 1-28 (2000).

2. Tanenbaum, A.S., "Computer Networks 4th ed.", ***Prentice Hall PTR*** Upper Saddle River, New Jersey, 1-26 (2003).

3. Gökçen, H., "Yönetim Bilgi Sistemleri Analiz ve Tasarım Perspektifi", ***EPİ Yayıncılık***, Ankara, 225-232 (2002).

4. Colbourn, C.J., "Telecommunications Network Planning" Sanso, B., Soriano, P., ***Kluwer Academic,*** Boston, 135-142 (1999).

5. McGreger, M., "CISCO CCIE Fundamentals: Network Design & Case Studies 2nd ed.", ***CISCO Press***, Indianapolis, 1-8 - 1-9 (1999)

6. Hayes, J.F., "Modeling and Analysis of Computer Communications Networks", ***Plenum Press,*** New York, 349-368 (1984).

7. Altıparmak, F., "Genetik Algoritma ile Haberleşme Şebekelerinin Topolojik Optimizasyonu", Doktora Tezi, ***Gazi Üniversitesi Fen Bilimleri Enstitüsü,*** Ankara, 15-22 (1996).

8. Konak, A., Smith,A.E., "Handbook of Optimization in Telecommunications 1st ed.", Resende, M.G.C., Pardalos, P.M., ***Springer Science+Business Media***, New York, 735-761 (2006).

9. Jan R.H., "Design of Reliable Networks", ***Computers and Operations Research***, 20: 25-34 (1993).

10. Yeh, M., Lin, J., Yeh, W., "A New Monte Carlo Method for Estimating Network Reliability", ***IEEE Transactions on Reliability,*** R-29 (1): 27-32 (1994).

11. Alabaş, Ç., "Tabu Arama ve Tavlama Benzetimi Algoritmalarıyla Bilgisayar Şebekelerinin Topolojik Optimizasyonu", Yüksek Lisans Tezi, ***Gazi Üniversitesi Fen Bilimleri Enstitüsü,*** Ankara, 22-26 (1999).

12. Dengiz, B., Altıparmak, F., Smith, A.E., "Local Search Genetic Algorithm for Optimal Design of Reliable Networks", ***IEEE Transactions on Evolutionary Computation,*** 1 (3): 179-188 (1997).

13. Boorstyn R.R., Frank, H., "Large Scale Network Topological Optimization", ***IEEE Transactions on Communications*****,** 23: 29-37 (1977).

14. Aggarwal, K.K., Chopra, C., Bajwa, J.S., "Topological Layout of Links For Optimazing the s-t Reliability in a Computer Communication System", ***Microelectronics and Reliability***, 22 (3): 341-345 (1982).

15. Chopra Y.C., Sohi, B.S., Twari, R.K., Aggarwal, K.K., "Network Topology for Maximizing the Terminal Reliability in a Computer Communication Network", ***Microelectronics and Reliability***, 24: 911-913 (1984)

16. Aggarwal, K.K., Suresh, R., "Reliability Evaluation in Computer Communication Networks", ***IEEE Transactions on Reliability,*** R-30 (1): 32-35 (1981).

17. Venetsanopoulos, A.N., Singh, I., "Topologıcal Optımızatıon of Communıcatıon Networks Subject to Relıabılıty Constraınts", ***Problems of Control andInformation Theory,*** 15 (1): 63-78 (1986)

18. Atiquallah, M.M., Rao, S.S., "Reliability Optimization of Communication Networks Using Simulated Annealing", ***Microelectronics and Reliability***, 33 (9): 1303-1319 (1993).

19. Kumar A., Pathak, R.M., Gupta, Y.P., "Genetic Algorithm Based Reliability Optimization for Computer Network Expansion", ***IEEE Transactions on Reliability,*** 44: 63-72 (1995a).

20. Kumar A., Pathak, R.M., Gupta, Y.P., "Genetic Algorithm for Distributed System Topology Design", ***Computers and Industrial Engineering,*** 28 (3): 659-670 (1995b).

21. Deeter, D.L., Smith, A.E., "Heuristic Optimization of Network Design Considering All-Terminal Reliability", ***Annual Reliability and Maintainability Symposium,*** Philedelphia PA, 194-199 (1997).

22. Ahuja, S.P., "Performance Based Reliability Optimization for Computer Networks", ***Southeastcon '97,*** Blacksburg, VA, 121-125 (1997).

23. Costamagna, E., Fanni, A., Giacinto, G., "A Simulated Annealing Algorithm for the Optimazion of Communication Networks", ***International Symposium on Signals,Systems and Electronics,*** San Francisco, 405-408 (1995).

24. Pierre, S., Legault, G., "A Genetic Algorithm for Designing Distributed Computer Network Topologies", ***IEEE Transactions on Systems, Man and Sybernetics,*** Part B 28(2) 249-258 (1998).

25. Dengiz, B., Altıparmak, F., Smith, A.E., "Efficient Optimization of All-Terminal Reliable Networks Using an Evolutionary Approach", ***IEEE Transactions on Evolutionary Reliability***, 46 (1): 18-26 (1997).

26. Pierre, S., Elgibaoui, A., "A Tabu Search Approach for Designing Computer Network Topologies with Unreliable Components", ***IEEE Transactions on Reliability,*** 46 (3): 350-359 (1997).

27. Costamagna, E., Fanni, A., Giorgio, G., "A Tabu Search Algorithm for the Optimization of Telecommunication Networks", ***European Journal of Operational Research***, 106: 357-372 (1998).

28. Cheng, S.T., "Topological Optimization of a Reliable Communication Network", ***IEEE Transactions on Reliability,*** 47 (3): 225-233 (1998).

29. Deeter, D.L., Smith, A.E., "Economic Design of Reliable Networks", ***IEEE Transactions***, 30: 1161-1174 (1998).

30. Altıparmak, F., Dengiz, B., Smith, A.E., "Reliability Optimization of Computer Communication Networks Using Genetic Algorithms", ***International Conference on Systems, Man and Sybernetics,*** 5: 4676-4681 (1998).

31. Liu, B., Iwamura, K., "Topological Optimization Models for Communication Network with Mutiple Reliability Goals", ***Computers and Mathematics with Applications***, 39: 59-69 (2000).

32. Dengiz, B., Alabaş, Ç., "A Simulated Annealing Algorithm for Design of Computer Communication Networks", ***World Multiconference on Systemics, Cybernetics and Informatics***, Orlando, Florida, USA 5 (I): 188-193 (2001).

33. AboElFotoh, H.M.F., Al-Sumait, L.S., "A Neural Approach to Communication Networks with Reliability Constraint", ***IEEE Transactions on Reliability***, 50 (4): 397-408 (2001).

34. Fard, N., Lee, T-H., "Spanning Tree Approach in All Terminal Reliability Expansion", ***Computer Communications***, 24: 1348-1353 (2001).

35. Koide, T., Shinmori, S., Ishii, H., "Topological Optimization with a Network Reliability Constraint", ***Discrete Applied Mathematics***, 115: 135-149 (2001).

36. Altıparmak, F., Dengiz, B., Smith, A.E., "Optimal Design of Reliable Computer Networks: A Comparison of Metaheuristics", ***Journal of Heuristics,*** 9: 471-487 (2003).

37. Kumar, R., Parida, P.P., Gupta, M., "Topological Design of Communication Networks Using Multiobjective Genetic Optimization", ***Proceedings of the 2002 Congress on Evolutionary Computation,*** 1 (12-17): 425-430 (2002).

38. Srivaree-ratana, C., Konak, A., Smith, A.E., "Estimation of All Terminal Reliability Using an Artificial Neural Network", ***Computers and Operations Research,*** 29: 849-868 (2002).

39. Mandal, S., Saha, D., Mukherjee, R., Roy, A., "An Efficient Algorithm for Designing Optimal Backbone Topology for a Communication Networks", ***International Conference on Communication Technology***, Beijing, China 1: 103-106 (2003).

40. Altıparmak, F., Dengiz, B., Smith, A.E., "Reliability Estimation of Computer Communication Networks: ANN Models", ***8th IEEE International Symposium on Computers and Communication***, Kemer, Antalya, 2: 1-6 (2003).

41. Altıparmak, F., Gen, M., Dengiz, B., Smith, A.E., "A Genetic Algorithm with Fuzzy Logic Controller for Design of Communication Networks", ***IEEJ Transactions EIS,*** Tokyo 24 (10):1979-1985 (2004).

42. Shao, F-M., Shen, X., Ho, P-H., "Reliability Optimization of Distributed Access Networks With Constrained Total Cost", ***IEEE Transactions on Reliability,*** 54 (3): 421-430 (2005).

43. Xiong, J., Gong, W., "A Novel Algorithm on Network Reliability Estimation", ***Mathematical and Computer Modelling,*** 41: 119-133 (2005).

44. Konak, A., Bertolacci, M.R., "Designing survivable resilient networks:A stochastic hybrid genetic algorithm approach", ***The International Journal of Management Science,*** 35: 645-658 (2007).

45. Cancela, H, Petingi, L., "Properties of a Generalized Source-to-All-Terminal Network Reliability Model with Diameter Constarints", ***The International Journal of Management Science*** Omega 35: 659-670 (2007).

46. Khan, S.A., Engelbrecht, A:P., "A New Fuzzy Operator and Its Application to Topology Design of Distributed Local Area Networks" ***Information Science*****s** 177: 2692-2711 (2007).

47. Reichelt, D., Gmilkowsky, P., "A Study of an Iterated Local Search on the Reliable Communication Networks Design Problem", ***EvoWorkshop 2005*** Lausanne, 156-165 (2005).

48. Lucio, G.F., Reed, M.,J., Hanning, I.D., "Guided Local Search as a Network Planning Algorithm That Incorporates Uncertain Traffic Demands", ***Computer Networks,*** 51 (11): 3172-3196 (2007).

49. Hui,K-P., "Monte Carlo Network Reliability Ranking Estimation", ***IEEE Transactions on Reliability,*** 56 (1): 50-57 (2007).

50. Marseguerra, M., Zio, E., Podofillini, L., Coit, D.W., "Optimal Design of Reliable Network Systems in Presence of Uncertainity", ***IEEE Transactions on Reliability,*** 54 (2): 243-253 (2005).

51. Qoubutra, M., Premprayoon, P., Wardkein, P., "Topological Communication Network Design Using Improved Ant Colony Optimization", ***Networks and Communication Systems,*** Krabi, Tayland, 118-128 (2005).

52. Hansen, P., Mladenovic, N., "Variable Neighborhood Search: Principles and Applications", ***European Journal of Operational Research,*** 130 (3): 449-467 (2001).

53. Flezar, K., Hindi, K.S., "Solving the Resource-Constarined Project Scheduling Problem by a Variable Neighbourhood Search" ***Discrete Optimization, European Journal of Operational Research***, 155 (2): 402-413 (2004).

54. Magoulas, G.D., Eldabi, T., Paul, R.J., "Adaptive Stochastic Search Methods for Parameter Adaptation of Simulation Models", First International ***IEEE Symposium on Intelligent Systems,*** Las Alamitos, 2: 23-27 (2002).

55. Shi, Y., Eberhart, R.C., "Empirical Study of Particle Swarm Optimization", ***Congress on Evolutionary Computation,*** Washington, 3: 1945-1950 (1999).

56. Tretea, I.C., "The Particle Swarm Optimization Algorithms: Convergence Analysis and Parameter Selection", ***Information Processing Lettters*** 85: 317-325 (2003).

57. Jin, Y., Cheng, H., Yan, J., Zhang, L., "New Discrete Method for Particle Swarm Optimization and Its Application in Transmission Network Expansion Planning", ***Electric Power Systems Research*** 77: 227-233 (2007).

58. Goss, S., Aron, S., Deneubarg, J.L., Pasteels, J. M., "Self Organized Shortcuts in the Argentina Ant", ***Naturwissenschaften***, 76:579-581 (1989).

59. Dorigo, M., Di Caro, G., "New Ideas in Optimization", Corne, D., Dorigo, M., Glover, F., ***McGraw-Hill,*** Maidenhead, 1-8 (1999).

60. Dorigo, M., Stützle, T., "Metaheuristics Handbook", Glover, F., Kochenberger, G., ***International Series in Operations Research and Management Sciene Kluwer Academic*****,** Boston**,** 1-13 (2001).

61. Dorigo, M., Maniezzo, V., Colorni, A., "Ant System: Optimization By a Colony of Cooerating Agents", ***IEEE Transactions on Systems, Man and Cybernetics***, Part B 26 (1): 29-41 (1996).

62. Dorigo, M., Gambardella, L.M., "Ant Colony Ssytem: A Cooperative Learning Approach to the Traveling Salesman Problem", ***IEEE Transactions on Evolutionary Computation,*** 1(1): 53-66, (1997).

63. Stützle, T., Hoos, H.H., "MAX-MIN Ant System", ***Future Generation Computer Systems,*** 16 (8): 889-914 (2000).

64. Bullnheimer, B., Hartl, R.F., strauss, C., "A New Rank Based Version of the Ant System: A Computational Study", ***Central European Journal of Operations Research and Economics,*** 7 (1): 25-38 (1995).

65. Cordon, O., Vianna, I.F., Herrera, F., Moreno, L., "A New ACO Model Integrating Evolutionary Computation Concepts: The Best Worst Ant System", Dorigo, M., Middendorf, M., Stützle, T., ***Proceedings of Ants'2000,*** Bruxelles, Belgium, 22-29 (2000).

66. Ratnaweera, A., Halgamuge, S.K., Watson, H.C., "Self-Organizing Hierarchical Particle Swarm Optimizer with Time-Varying Acceleration Coefficients", ***IEEE Transactions on Evolutionary Computation*** 8 (3): 240-255 (2004).

67. Chatterjee, A., Siarry, P., "Nonlinear Inertia Weight Variation for Dynamic Adaptation in Particle Swarm Optimization", ***Computers and Operations Research*** 33: 859-871 (2006).

68. İnternet: Prof. Berna Dengiz's Home Page "Test Problems" www.baskent.edu.tr/~bdengiz/testproblems (2007).

69. İnternet: Prof. Dr. Fulya Altıparmak Kişisel Web Sitesi "Test Problems" w3.gazi.edu.tr/~fulyaal/testproblems (2007).

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: BELGİN, Önder
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 02.06.1981 Ankara
Medeni Hali	: Bekar
Telefon	: 0 (312) 467 55 90
Faks	: 0 (312) 427 30 22
e-mail	: onderbelgin@gmail.com

Eğitim

Derece	Eğitim Birimi	Mezuniyet tarihi
Yüksek lisans	Gazi Üniversitesi / Endüstri Müh.	2007
Lisans	Gazi Üniversitesi/ Endüstri Müh.	2004
Lise	Çankaya Lisesi	1999

İş Deneyimi

Yıl	Yer	Görev
2004-	Milli Prodüktivite Merkezi	Uzman Yardımcısı

Yabancı Dil

İngilizce

Yayınlar

1. Dengiz, B., Belgin, Ö., "KOBİlerde Benzetim ile Verimlilik Artışı: Türkiye ve AB KOBİ'lerinden Uygulamalar", II. KOBİ'ler ve Verimlilik Kongresi, İstanbul (2005).

2. Civek, E., Karakaya, Ş., Belgin, Ö., "Multi-Criteria Comparıson of Logistics Outsourcing Providers Using Fuzzy AHP" 3rd International Logistics and Supply Chain Congress, İstanbul (2005) .