T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GEZGİN SATICI PROBLEMLERİNİN ÇÖZÜMÜ İÇİN ELEKTROMANYETİZMA SEZGİSELİNİN UYARLANMASI

BURAK TOPCU

YÜKSEK LİSANS TEZİ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI
ENDÜSTRİ MÜHENDİSLİĞİ PROGRAMI

DANIŞMAN
YRD. DOÇ. DR VİLDAN ÇETİNSAYA ÖZKIR

İSTANBUL, 2014

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

GEZGİN SATICI PROBLEMLERİNİN ÇÖZÜMÜ İÇİN ELEKTROMANYETİZMA SEZGİSELİNİN UYARLANMASI

Burak TOPCU tarafından hazırlanan tez çalışması . .2014 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Fen Bilimleri Endüstri Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Yrd. Doç. Dr. Vildan Ç.ÖZKIR

Yıldız Teknik Üniversitesi

Jüri Üyeleri

Yrd. Doç. Dr. Vildan Ç.ÖZKIR

Yıldız Teknik Üniversitesi _____

Prof. Dr. İlker BİRBİL

Sabancı Üniversitesi _____

Yrd. Doç. Dr. Tuğba EFENDİGİL

Yıldız Teknik Üniversitesi _____

ÖNSÖZ

Günümüz rekabet dünyasında araç rotalama stratejileri büyük önem taşımaktadır. Bir ürünü üretme aşamasında kullanılan optimizasyon teknikleri kadar, o ürün için kullanılacak ham maddenin toplanması ve bitmiş ürünün dağıtımı aşamasının optimizasyonu da aynı şekilde çok önemlidir. Zira akaryakıt, araç, insan ve zaman gibi kaynakların ciddi oranda kullanıldığı bu aşamalar, verimli bir endüstri profili çizilmesi açısından hayati önem taşımaktadır.

Bugüne kadar rota ağlarının optimizasyonunda çeşitli yöntemler kullanılmıştır. Farklı sezgisel ve meta-sezgisel algoritmaların sıklıkla kullanıldığı bu araştırma sahasında yaptığım çalışmada Elektromanyetizma Sezgisel Algoritmasını kullanılmıştır. Araç rotalama problemlerinin bir çeşidi olan gezgin satıcı problemlerine yeni bir yaklaşım getirmek adına ele alınmıştır.

Bu tezimde bana her daim yardımcı olan başta tez danışmanım Yrd. Doç. Dr. Vildan ÇETİNSAYA ÖZKIR hocama, tezimin özellikle yazılım aşamasında verdikleri fikirlerle yeni ufuklar edinmeme vesile olan Salih TOPCU, Erdal DEMİRCİ ve Muhammed PARLAK'a, her zaman yanımda olan aileme teşekkürü borç bilirim.

Haziran, 2014

Burak TOPCU

İÇİNDEKİLER

Sayfa

SİMGE LİSTESİ

n	düğüm noktası
b	dallanma faktörü
d	dallanma ağacının maksimum derinliği
m	parçacık sayısı
k	boyut indisi
i	parçacık indisi
j	parçacık indisi
λ	rassal adım uzunluğu
δ	yerel arama çarpanı
msf	mesafe

KISALTMA LİSTESİ

2-HTSP	İki Depolu Heterojen GSP'de
ÇGSP	Çoklu Gezgin Satıcı Problemi
EMA	Electromagnetism Algorithm
FIFO	First Input First Output
GA	Genetik Algoritma
GSP	Gezgin Satıcı Problemi
İYA	İteratif Yerel Arama Algoritması
KA	Kanguru Algoritması
KKA	Karınca Kolonisi Algoritması
LSITER	Local Search Iteration
MAXITER	Maximum Iteration
MBP	Maximum Betweenness Problem
NP-hard	Non-deterministic Polynomial-time hard
NP-Tam	Non-deterministic Polynomial-time tam
RK	Random Key (Rassal Anahtar)
ŞKG	Şu Kısıtlara Göre
TA	Tabu Araması
TB	Tavlama Benzetimi
TSP	Travelling Salesman Problem
TT	Tepe Tırmanma
YAK	Yapay Arı Kolonisi Algoritması
ZPGSP	Zaman Pencereli Gezgin Satıcı Problemi

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

Sayfa

ÖZET

GEZGİN SATICI PROBLEMLERİNİN ÇÖZÜMÜ İÇİN ELEKTROMANYETİZMA SEZGİSELİNİN UYARLANMASI

Burak TOPCU

Endüstri Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Tez Danışmanı: Yrd.Doç.Dr. Vildan ÇETİNSAYA ÖZKIR

Günümüzde özellikle lojistik alanında karşılaşılan araç rotalama ve gezici satıcı problemi gibi karmaşık ve kombinatoryal problemlerin çözümünde, doğrusal modelleme gibi yöntemlerin dışında çeşitli yapay zeka yöntemleri ile birlikte farklı sezgisel algoritmalar da önerilmiştir. Özellikle araç rotalama problemlerinin çok kombinasyonlu yapısı çözüm yöntemlerinin seçiminde en önemli etkenlerden biridir. Bu sebeple matematiksel modellerin oluşturulmasının güç olduğu durumlarda kullanılan sezgisel algoritmalar stokastik ve iterasyonlu yapıları sayesinde optimal çözümün bulunmasında veya optimal çözüme yeterince yakınsanmasında büyük öneme sahiptirler. Amaç fonksiyonunu optimize etmek amacıyla bu sezgiseller tek başına kullanılabildikleri gibi farklı arama algoritmaları ve hatta farklı sezgisel algoritmalar ile melezlenerek de kullanılabilmekte ve bu sayede çözümü bulma yetenekleri de sürekli olarak geliştirilmektedir. Bu çalışmada ele alınan Elektromanyetizma Sezgisel Algoritması son yıllarda kullanılmaya başlanmış olup sürekli fonksiyonlar üzerinde optimizasyon amacı ile geliştirilmiştir. Öncelikle, bu algoritma çeşitli 3 boyutlu fonksiyonlar üzerinde test edilmiş ve uygun hesaplama zamanları içinde doğru çözümler verebildiği gözlemlenmiştir. Sürekli fonksiyonlar üzerinde alınan olumlu sonuçların ardından, günümüzde karşımıza çıkan çok parametreli, çok değişkenli, kesikli ve kısıtlı problemler üzerinde de çalışabilmesi için elektromanyetizma sezgisel algoritması Rassal Anahtar (Random Key) yöntemi ile birleştirilerek geliştirilen güncel mekanizma ile sıralama, atama, çizelgeleme, araç rotalama gibi kesikli problemlerin çözümüne elverişli hale getirilmiştir.

Kapasite kısıtsız bir gezici satıcı problemi üzerinde uygulanan algoritmanın uygun çözüm süresi içinde optimal sonuçlar verdiği gözlemlenmiştir.

Anahtar Kelimeler: Elektromanyetizma sezgisel algoritması, meta-sezgisel algoritmalar, rassal anahtar yöntemi, kapasite kısıtsız gezici satıcı problemleri.

ABSTRACT

ADAPTING ELECTROMAGNETİSM-LIKE HEURISTICS FOR SOLVING TRAVELLING SALESMAN PROBLEMS

Burak TOCU

Department of Industrial Engineering

MSc. Thesis

Advisor: Assist. Prof. Dr. Vildan ÇETİNSAYA ÖZKIR

Nowadays, in order to solve such combinatorial and complicated problems like vehicle routing and sales traveling problems which are seen especially logistics fields, different heuristic algorithms and artificial intelligence methods are proposed as an alternative for mathematical programming methods. Especially, the multi combinational structure of vehicle routing problems is one of the most important factors in the choice of solving methods. Thus, owing to stochastic and iterative structure, meta-heuristic algorithms have a big practicality to find or converge the optimal solution in the situations when it is difficult to create mathematical model. Heuristics algorithms can be used solely to find the optimal value of objective function; however they can be applied by hybridizing them with different searching algorithms and different heuristics in order to develop their ability of finding optimal solution in different problem types. We handled Electromagnetism-like heuristic algorithms which is started to use in recent years. Actually, this algorithm has been introduced for global optimization problems. First, we applied this applied onto 3 dimensional functions and observed that this algorithm is successfully able to find the global optimum point in suitable computational times. After getting good results from continuous functions, this algorithm has been hybridized by integrating the Random Key mechanism in order to get it convenient to solve actual multi-parametric, multivariate, discrete and constrained optimization problems such as scheduling, sequencing, assignment, vehicle routing problems.

The computational results have been showed that, the proposed Electromagnetism-like Algorithm gives good results on travelling salesman problems within suitable computational times with respect to different meta-heuristic algorithms.

Keywords: Electromagnetism heuristic algorithm, meta-heuristic algorithms, random key methods, incapacitated travelling salesman problems.

BÖLÜM 1

GİRİŞ

Endüstrilerde üretim süreçleri kadar, üretim işlemlerinde kullanılacak ham maddenin tedarik edilmesi ve üretilen ürünün müşteriye ulaştırılması aşamalarının da verimli hale getirilmesi günümüz rekabet dünyasında rakip firmalara karşı üstünlük sağlamak açısından çok önemlidir. İmalat süreçleri ne kadar iyileştirilirse iyileştirilsin, tedarik ve teslimat ağlarının optimize edilmemesi durumunda ürünün toplam maliyeti hem yüksek çıkacaktır hem de sarkan teslim alma ve teslim etme süreleri üreticiyi müşkül konuma getirecektir. Bu problemin önüne geçmek için araç rotalama süreçlerininin optimizasyonu firmaya ciddi rekabet avantajı kazandıracaktır.

1.1 Literatür Özeti

Son yıllarda kombinatoryal optimizasyon yapay zeka konusunun önemli bir araştırma sahası haline gelmiştir. Karmaşık optimizasyon problemlerinin çözülebilmesi için farklı zamanalrda çeşitli sezgisel ve meta-sezgisel algoritmalar önerilmiştir. Kompinatoryal problemlerin teori ve uygulamadaki karmaşıklıklarından dolayı, bir çok araştırmacı farklı sezgisel algoritmaları birlikte kullanarak melez meta-sezgisel algoritmaların geliştirimesi yoluna da gitmişlerdir.

Gezgin satıcı problemi araştırmacıların üzerinde sıklıkla çalıştıkları n adet düğüm noktasından oluşan bir kombinatoryal optimizasyon problemidir. Başlangıç düğümü haricindeki tüm düğümlere yalnızca 1 kere uğranılır. Oluşturulan tur başlancık düğümünde biter. Bu tür problemler polinomal sürede çözülemezler NP-hard problemler olarak tanımlanırlar[1][2].

Bugüne kadar gezgin satıcı problemleri farklı meta-sezgisel algoritma kullanılarak çözülmüştür. Örneğin, tavlama benzetimi [3], karınca kolonisi [4], tabu araması [5],[6], genetik algoritma [7], parçacık sürü algoritması [8], genetik algoritma, tavlama benzetimi, karınca kolonisi ve parçacık sürü algoritmalarını birleştirerek geliştirilen melez bir yapı [9], gezgin satıcı problemlerinin çözümünde kullanılmıştır.

Bu tez çalışmasında gezgin satıcı problemlerinin çözümünde kullanılmak üzere Elektromanyetizma sezgisel algoritmasını önereceğiz. Bu algoritma ilk olarak global optimizasyon problemlerinde kullanılmak üzere geliştirilmiştir [10],[11]. Bir takım dönüşümler yaparak ve diğer sezgisel algoritmalar ile gerçekleştirilen melezlemeler ile araştırmacılar kesikli ve kısıtlı problemlerin çözümü için yeni yapılar geliştirebilmektedirler. Son birkaç çalışmada, elektromanyetizma sezgisel algoritmasının, simetrik olmayan matrislerde eigen değeri ve eigen ve eigen vektörünün hesaplanamsında [12], araç rotalamada [13],[14], hücre tasarımı ve düzenlemesinde [15], maksimum arasındalık probleminin (MBP) çözümünde [16], hemşire çizelgeleme probleminde [17], proje çizelgelemede [18], bir makina çizelgelemede [19][20][21][22]. Başarı ile kullanılmıştır. Bunun yanında, EMA gezgin satıcı problemleri içinde kullanılmış. Fakat bizim çalışmamızdan farklı olarak kesikli ve ikili değişkenli (0,1) versiyonu önerilmiştir [23].

1.2 Tezin Amacı

Günümüz dünyasında artan rekabet baskısında, firmalar etkinliklerini koruyabilmek için optimizasyon faaliyetlerine ağırlık vermek zorundadır. Yüksek nakliye, insan ve zaman maliyetinden dolayı araç rotalama ve diğer lojistik faaliyetleri üzerinde yapılacak iyileştirme faaliyetleri de büyük önem arz etmektedir.

Gezgin satıcı problemleri daha önce de belirttiğimiz gibi araştırmacılar tarafından çokça incelenmiş ve bu problemlerin farklı tipleri farklı sezgisel algoritmalarla çözülmeye çalışılmıştır.

Bu tez çalışmasında Elekromanyetizma sezgisel algoritması kullanılarak bu sezgisel algoritmanın farklı boyutlardaki gezgin satıcı problemlerini çözme yeteneği incelenecektir. Açgözlü arama algoritması ve rassal anahtar yöntemleri ile birleştirerek melez bir yapı önerilecektir.

1.3 Hipotez

Bu tez çalışmasında, yöneylem araştırması ve teorik bilgisayar bilimi alanlarında sıklıkla incelenen araç rotalama problemlerinin bir alt sınıfı olan gezgin satıcı problemleri ele alınacak ve nispeten yeni bir meta-sezgisel algoritma olan elektromanyetizma sezgisel algoritmasının gezgin satıcı problemleri üzerindeki çözüm kalitesi ve performansı incelenecektir.

Bu amaçla başladığım tez çalışmasında, ikinci bölümünde gezgin satıcı problemlerinin tarihçesi, tanımı ve yapısı anlatılmıştır. Problemin kombinatoryal yapısından dolayı ortaya çıkan çok ihtimallilik ve karmaşıklıktan ötürü kullanılan farklı çözüm teknikleri anlatılmıştır. Üçüncü bölümde gezgin satıcı probleminin çözümünde kullanacağımız Elektromanyetizma algoritmasının mekanizmasının teorisi, kullandığı analoji ve sözde kod yapısından bahsedilecektir. Bununla birlikte, bu algoritmanın sürekli fonksiyonlarda nasıl çalıştığı ve optimal çözüme nasıl yakınsadığı anlatılacaktır. Dördüncü bölümde sürekli fonksiyonlarda kullanılan elektromanyetizma sezgiselini kesikli ve çok kısıtlı modellere uyarlanabilmesi için kullanılacak olan rassal anahtar yöntemi tanıtılmış ve açıklanmıştır. Daha sonra rassal anahtar yönteminin elektromanyetizma algoritmasında uygulanma aşamaları açıklanmış ve safha safha gösterilmiştir. Beşinci bölümde elektromanyetizma algoritması gezgin satıcı problemlerine uygulanarak çözüm kalitesi doğrusal programlama yöntemi ve tavlama benzetimi ile karşılaştırılmıştır. Altıncı bölümde elde edilen bulgulardan ve önerilerden bahsedilmiştir.

BÖLÜM 2

GEZİCİ SATICI PROBLEMİ

Gezgin Satıcı Problemi (GSP), aralarındaki uzaklıklar bilinen n adet noktanın (şehir, parça veya düğüm gibi) her birisinden yalnız bir kez geçen en kısa veya en az maliyetli turun bulunmasını hedefleyen bir problemdir.

Yada GSP, n adet şehir arasındaki mesafelerin bilindiği durumda, şehirlerin her birine yalnız bir kez uğramak şartıyla, başlangıç noktasına geri dönülmesi esasına dayalı, tur boyunca kat edilen toplam yolun en kısa olduğu şehir sıralamasının (optimal rota) bulunmasının amaçlandığı bir problemdir. Dağıtım, rotalama, kuruluş yeri belirleme, planlama, lojistik gibi problemlerde geniş bir uygulama alanına sahip olan gezgin satıcı problemi, aynı zamanda optimizasyon alanında, araştırmacılar tarafından üzerinde uzun yıllardır çalışmalar yapılan NP-hard (çözümü zor) sınıfında yer alan bir problemdir.

Ayrık ve Kombinatoryal Eniyileme (Combinatorial Optimization) problemlerinin kapsamına girer. Problemdeki maliyet, uzaklık, zaman veya para gibi unsurlardan biri olabilir. Eğer şehirler düğümlerle, yollar ise hatlar ile gösterilirse problem çizge üzerinde minimum maliyetli kapalı yolun bulunmasına karşılık gelmektedir [24]. Çizge teorisinde ise, gezgin satıcı problemi “Verilen bir ağırlıklı çizgede (düğümler yani köşeler şehirleri; kenarlar ise şehirlerin arasındaki yolları göstermek; ağırlıklar da yolun maliyeti veya uzunluğu olmak üzere), en düşük maliyetli Hamilton Çevrimi’nin bulunması” şeklinde tanımlanabilir.

Problemde düğüm sayısı arttıkça problemin çözümünde harcanan zaman üstel olarak artmaktadır. GSP'de artan düğüm sayısına paralel olarak çözüm zamanının üstel olarak artışı Çizelge 1'de gösterilmektedir.

Çizelge 2.1 Hamilton Döngülerinin Değerlendirilmesi [24].

Düğüm Sayısı	**Döngü Sayısı (n-1)!**	**Gerekli Zaman**
12	39.916.800	0.004 saniye
13	479.001.600	0.05 saniye
14	6.227.020.800	1 saniye
15	87.178.291.200	9 saniye
16	1.307.647.368.000	2 dakika
17	2.1 * 1013	35 dakika
18	3.6 * 1014	10 saat
19	6.4 * 1015	7.5 gün
20	1.2 * 1017	140 gün
21	2.4 * 1018	7.5 yıl
22	5.1 * 1019	160 yıl
23	1.1 * 1021	3.500 yıl
24	2.6 * 1022	82.000 yıl
25	6.2 * 1023	2 milyon yıl

Problemde başlangıç şehri verilmişse, mümkün olan Hamilton yolları sayısı geriye kalan (n-1) adet şehrin yer değişmesine, yani (n-1)!'e eşit olmaktadır. Bu durumda problem basit olmasına rağmen, problemin çözümünü çözüm uzayının tamamını taramakla bulmak çok iyi bir yaklaşım değildir. Problemin en azından bir basit çözümünün olacağı kesindir. Bu sebeple GSP problemlerinin çözümünde sezgisel ve metasezgisel tekniklerin kullanılması etkin bir yoldur [25].

Problemin matematiksel ifadesi şu şekilde yapılabilir [26]:

Verilen bir "maliyet matrisi" C = (c_{ij}) için - ki burada c_{ij} şehir i'den şehir j'ye gitme maliyetini temsil eder (i,j = 1,..., n) - 1'den n'e kadar tamsayıların aşağıdaki niceliği minimize eden bir permütasyonunu (i1, i2, i3, ..., in) bulunuz:

$$c_{i_1 i_2} + c_{i_2 i_3} + \ldots + c_{i_n i_1}$$

Maliyet matrisi C'nin özellikleri problemlerin sınıflandırılmasında kullanılır:

Tüm i ve j değerleri için $c_{ij} = c_{ji}$ ise problem simetriktir ve Simetrik GSP adını alır; aksi takdirde ise asimetriktir ve Asimetrik GSP adını alır. Problemin en yaygın şekli Simetrik GSP'dir. Ancak bazı durumlarda A şehrinden B şehrine gitmenin maliyeti ile, B şehrinden A şehrine gitmenin maliyeti farklıdır. Bazı şehirlerarasında tek yön yolların olması, gidiş ve geliş yönlerindeki yolların trafik sıkışıklığının getireceği süre farklılıkları gibi durumlar dikkate alındığında Asimetrik GSP devreye girer.

Eğer üçgen eşitsizliği sağlanıyorsa (tüm i,j ve k değerleri için $c_{ik} \leq c_{ij} + c_{jk}$ ise), problem metriktir ve Metrik GSP adını alır.

Eğer c_{ij} düzlemdeki noktalar arasındaki Öklit uzaklığı ise problem Öklit tabanlıdır ve Öklit Tabanlı GSP adını alır. Problemin bu tipi için düğümler, R^2 (daha genel anlamda herhangi bir d için, R^d şeklinde) uzayındadır. Öklit Tabanlı GSP doğal olarak hem simetrik hem de metriktir.

2.1 Gezgin Satıcı Problemlerinin Çeşitleri

Literatürde klasik GSP'nin amaç fonksiyonunun değiştirilmesi ve/veya kısıtlarının farklılaştırılması ile elde edilen farklı çeşitleri bulunmaktadır.

Simetrik GSP, noktalar arası mesafelerin gidiş-dönüş için aynı olduğu durumlarda, Asimetrik GSP ise farklı olduğu durumlarda söz konusudur [27].

Kârlı GSP, bütün düğüm noktalarının ziyaret edilme zorunluluğunun bulunmadığı GSP'nin genelleştirilmiş halidir. Her noktaya ait bir kâr değeri vardır. Amaç, toplanan kâr ile yapılan yol masraflarının eşzamanlı en iyilenmesidir. Bu iki eniyileme kriteri ya amaç fonksiyonunda bulunmakta ya da kısıt olarak yazılmaktadır [28].

Zaman Pencereli GSP (ZPGSP), her şehrin önceden belirlenen zaman pencereleri içinde ziyaret edilmesi kısıtını ekleyerek oluşturulur [29].

Belirsiz GSP'de gerçek hayattaki pek çok belirsizliğin hesaplamalara dâhil edilmesi söz konusudur.

Bu tip GSP problemlerine örnek olarak, trafik, hava durumu gibi etkenlerin tur boyunca yollarda geçecek süre üzerinde belirsizlik oluşturması verilebilir.

Araştırmacılar ağırlıklandırmalar yaparak ve belirsizlik teoremini kullanarak Belirsiz GSP üzerinde çözümler aramaktadırlar [30].

İki Depolu Heterojen GSP'de (2-HTSP), bir hedef seti üzerinde, iki farklı merkezden hareket eden ve birbirinden farklı olan araçlar tüm hedeflere uğrayarak araç başına minimum mesafeyle turlarını tamamlarlar [31].

Çoklu GSP'de (ÇGSP), n adet şehir, her biri ayrı bir satıcıya atanmak üzere m adet tura bölünmektedir. ÇGSP, GSP'den daha zor bir problemdir, zira çözümü için öncelikle her bir satıcıya hangi şehirlerin atanacağının ve satıcıların turları üzerindeki şehirlerin optimal sıralamasının belirlenmesi gerekmektedir [32].

Dinamik GSP, zaman içerisindeki değişimleri dikkate alan ve sürekli yeni bir optimum bulmayı amaçlayan GSP çeşididir. Bu değişkenlik problemin çözümünü karmaşık hale getirmektedir [32].

Genelleştirilmiş GSP'de, bir gezgin satıcı s salkımlı, n düğümlü bir ağda bir başlangıç noktasından başlayıp her salkımdan bir düğüme sadece bir defa uğrayıp başladığı yere dönmek durumundadır. Uğrayacağı yerlerin sıralarını belirlerken de kat edeceği toplam mesafenin veya yapacağı harcamanın en küçük olmasını amaçlamaktadır. Uçaklar için havaalanı rotalaması, elektronik devre tasarımı, posta kutusuna dağıtım problemleri, malzeme akış sistemleri tasarımı genelleştirilmiş GSP'ye örnek olarak verilebilir [33].

Açık Döngülü GSP'de, gezgin satıcı her noktaya uğramakta ancak başladığı noktaya geri dönme zorunluluğu bulunmamaktadır. Her şehir gezgin satıcı tarafından ziyaret edilmiş olmaktadır[34].

2.2 Gezgin Satıcı Probleminin Önemi

GSP'nin önemi, yolculuk mesafelerini minimize etmek isteyen satıcıların büyük ihtiyaçlarından doğmaz. Asıl önem pek çoğu yolculuk rotalarıyla ilişkili görünmeyen çok sayıda başka uygulamadan ileri gelir [26].

Örneğin, şu süreç planlama problemini ele alalım: Belirli bir sayıda işin tek bir makine üzerinde yapılması gerekmektedir. Makine aynı anda yalnızca bir iş yürütebilmektedir. Bir işin yapılmasına başlanmadan önce makine hazırlanmalıdır (temizlenme, ayarlanma vb). Her bir işin yürütülme zamanının ve bir işten diğerine geçerken kaybolan zamanın verildiği durumda, amaç toplam yürütülme zamanını olabildiğince kısa tutacak bir işletim sırası bulmaktır.

Bu problemin bir GSP örneği olduğu kolayca görülebilir. Burada c_{ij} i işinden sonra j işini tamamlamak için gereken süreyi temsil eder (iki iş arasındaki geçiş zamanı ve j işini yapmak için gereken zaman). Yürütülme zamanı 0 olan sözde bir iş makine için başlama ve bitiş durumunu belirtir.

GSP, optimizasyon teknikleri üzerine çalışma yapan araştırmacılar için ideal bir test ortamı oluşturmaktadır. Araştırmacılar geliştirdikleri yöntem(ler)in üstünlüklerini ispat etmek üzere literatürde sıklıkla kullanılan GSP'den faydalanmaktadır. GSP birçok gerçek hayat problemine uygulanmakla birlikte en çok yol ve rota planlama gibi konularda kullanılmaktadır [24].

GSP'nin kullanım alanları şu şekilde sıralanabilir[35][36].

- GSM operatörlerinin baz istasyonlarının yerleşim yerlerinin belirlenmesi,
- Birçok ulaşım ve lojistik uygulamaları,
- Malzeme akış sistem tasarımı,
- Araç rotalama problemleri,
- Depolardaki vinç güzergâhlarının programlaması,
- Stok alanındaki malzeme toplama problemleri,
- Uçaklar için havaalanı rotalaması,

- Elektronik devre tasarımı
- Röportaj zamanlama,
- Matbaa zamanlama,
- Bilgisayar kablolama,
- Ekip planlama,
- Misyon planlama,
- Navigasyon uydu sisteminin ölçme ağlarının tasarımı,
- Sipariş toplama.

GSP türünün, yani kombinatoryal eniyilemenin tipik bir problemidir. Bu, GSP üzerinde yapılan çalışmalardan edinilen teorik ve pratik kavrayışın, bu alandaki diğer problemlerin çözümü için de sıklıkla yarar sağlayabileceği anlamına gelir. Aslında kombinatoryal eniyilemedeki birçok gelişmenin GSP üzerindeki araştırmalara kadar izi sürülebilir. Şu anda iyi bilinen bir hesaplama yöntemi olan dallandır ve bağla (branch and bound) ilk dekomfa GSP kapsamında kullanılmıştır [37]. Ayrıca şunu da belirtmek gerekir ki, GSP üzerindeki araştırmalar 1970lerin başında hesaplama karmaşıklığı (computational complexity) teorisinin geliştirilmesinde de önemli bir itici güç olmuştur [38].

Fakat GSP'ye duyulan ilgi yalnızca pratikte ve teorideki öneminden kaynaklanmamaktadır. Problemi çözmenin zihinsel zorluğu da büyük rol oynar. Basit tanımına rağmen GSP çözülmesi güç bir problemdir. Zorluk, olası turların sayısı düşünüldüğünde - görece az sayıda şehir için bile astronomik bir değer - aşikar hale gelir. n şehirlik simetrik bir problem için (n-1)!/2 olası tur bulunmaktadır. n = 20 ise 10^{18}'den fazla tur var demektir. Örnek olarak Helsgaun un (2000) çalışmasındaki 7397 şehirlik problem içinse 10^{25000}'den fazla tur mevcuttur. Karşılaştırma yapmayı kolaylaştırmak adına, evrendeki temel parçacıkların sayısının yalnızca (!) 10^{87} olduğunu belirtmek gerekir.

2.3 Gezgin Satıcı Problemi için Çözüm Algoritmaları

GSP çözümünde ilk olarak klasik yöntemler uygulanmıştır. Bu yöntemler kesin ve sezgisel yöntemlerden oluşmaktadır.

Doğrusal Programlama [39], dinamik programlama [40], dal-kesim yöntemi [41],[42] gibi kesin yöntemler küçük problemlerin çözümünde kullanılmıştır.

2opt [43], 3opt [44], Markov zinciri [45], tavlama benzetimi [46][47]), tabu arama [48][49] gibi sezgisel yöntemler büyük problemlerin çözümü için kullanılmıştır.

Bunun yanında daha etkin çözümler için agresif prensiplere dayalı en yakın komşu [50], minimum kapsayan ağaç [51] gibi yöntemler de uygulanabilmektedir.

Çözüm uzayının çok büyük olduğu durumlarda klasik metotlar GSP çözümünde yetersiz kalmaktadır. Bu yetersizliklerin üstesinden gelmek için yeni yöntemlere ihtiyaç duyulmuştur.

Populasyon temelli optimizasyon algoritmaları, genellikle tabiattan ilham alınarak geliştirilmiş tekniklerdir [52].

Tabiatta yaşayan varlıklar, işleyen ve gelişen doğal sistemler; bilim, teknoloji vb. farklı alanlarda yapılan tasarımlar ve icatlar için ilginç ve değerli bir ilham kaynağıdır.

Genetik Algoritma [53],[54], Karınca Kolonisi Algoritması [55][56], Arı Kolonisi Algoritması [57][58], Yapay Sinir Ağları [59][60], Yapay Bağışıklık Sistemi [61][62], Parçacık Sürü Optimizasyonu [63], Akıllı Su Damlaları [64], Elektromagnetizma Benzetimli Mekanik Algoritması [65] bu alanda yer alan GSP çözüm tekniklerindendir.

GSP'nin NP-tam problemler kümesinin elemanı olduğu ispatlanmıştır. Bu, zaman karmaşıklıkları üstel olan çok zor problemlerden oluşan bir sınıftır.

Sınıfın elemanları birbiriyle ilişkilidir, yani bir problem için bir polinomsal zaman bulunursa, tüm problemler için polinomsal zaman algoritmaları mevcut olabilir. Ancak yaygın kanı böyle bir polinomsal algoritmanın olmadığı yönündedir. Bu yüzden GSP için optimal çözümler bulmak için genel bir algoritma oluşturma girişimlerinin tümü (muhtemelen) başarısız olacaktır.

Çünkü, böyle bir algoritma dahilinde üretilecek problem örnekleri için çalışma zamanı, girdinin büyüklüğüyle orantılı olarak en azından üstel olarak büyüyecektir. Elbette, buradaki zaman karmaşıklığının herhangi bir algoritmanın en kötü senaryolardaki davranışına tekabül ettiğine dikkat edilmelidir. Ortalama çalışma zamanları polinomsal olan algoritmaların bulunabileceği göz ardı edilemez. Bu tür algoritmaların varlığı ise hala cevaplanmamış bir sorudur.

En kısa güzergahı (turu) bulmanın en basit yolu, verilen N adet şehir için tüm şehir permütasyonlarını listeleyerek her bir olası güzergahın toplam yol uzunluğunu hesaplamaya dayanmaktadır [66].

En küçük değere sahip olan güzergah veya güzergahlar kesin çözümdür. Ancak bu yöntemin zaman karmaşıklığı O(n!)'dir. Dolaşılacak 6 şehir varsa, toplam 6! =720 farklı olası güzergahın toplam yol uzunluğunun hesaplanması gerekecektir. 40 şehir olduğu durumda bile, permütasyonların sayısı, bilgisayarın kısa sürede çözemeyeceği kadar büyük olmaktadır. Bu nedenle, kesin yöntemlerin yanında, kısa sürede iyi çözümlerin bulunmasını sağlayan yaklaşım yöntemleri de günümüzde birçok alanda kullanılmaktadır. Problem boyutunun büyük olduğu durumlarda hızlı bir şekilde iyi çözümlerin bulunmasını sağlayan sezgisel (heuristics) ve yaklaşım algoritmaları, 1950'lerden bu yana, gezgin satıcı problemlerinde kullanılmaktadır [67].

Bu ön bilgiyle birlikte GSP için çözüm algoritmaları Kesin algoritmalar ve yaklaşık algoritmalar olarak iki sınıfa ayrılabilir.

2.3.1 Kesin Algoritmalar

Kesin algoritmalar, optimal çözümü sınırlı sayıda adımda bulmayı garanti eder. Günümüzde birkaç bin şehirlik simetrik problemler için kesin sonuçlar bulunabildiği gibi, daha fazla sayıda şehir içeren problemlerin çözüm haberleri de gelmektedir.

En etkili kesin algoritmalar, kesit düzlem (cutting-plane) veya yüzey bulma (facet-finding) algoritmalarıdır [68],[69],[70]. Bu algoritmalar oldukça karmaşıktır ve 10.000 satır düzeyinde koda sahiptirler. Ayrıca algoritmalar yüksek bilgisayar gücüne ihtiyaç duyarlar. Örneğin, 2392 şehirlik bir simetrik problemin kesin çözümü güçlü bir süper bilgisayar üzerinde 27 saatten uzun bir süreçte bulunabilmiştir [69].

7397 şehirlik bir problemin kesin çözümü ise çok geniş bir bilgisayar ağı üzerinde yaklaşık 3-4 yıllık CPU zamanı almıştır [70]. Simetrik problemlerin çözülmesi genellikle asimetrik problemlerden daha zordur [71].

2.3.2 Yaklaşık Algoritmalar

Kesin algoritmaların aksine, yaklaşık algoritmalar iyi çözümler elde etmelerine rağmen optimal çözümün bulunacağını garanti etmezler.

Bu algoritmalar genellikle oldukça basittir ve görece kısa çalışma zamanlarına sahiptir. Bazı algoritmalar ortalama olarak optimal çözümden yalnızca küçük bir yüzdeyle farklı olan çözümler verir. Bundan dolayı, optimum değerden küçük bir sapma kabul edilebilirse, yaklaşık algoritmaları kullanmak uygun olabilir.

Sezgisel algoritmalar sınıfı üç alt sınıfa ayrılabilir:

- Tur oluşturma algoritmaları
- Tur iyileştirme algoritmaları
- Karma algoritmalar

Tur oluşturma algoritmaları her adımda yeni bir şehir ekleyerek bir güzergahı kademeli olarak meydana getirir. Tur iyileştirme algoritmaları çeşitli değiş tokuşlarla bir tur üzerinde iyileştirmeler yapar. Karma algoritmalar ise bu iki özelliği birleştirir.

Basit bir tur oluşturma algoritması örneği olarak en yakın komşu algoritması [72] gösterilebilir. Bu algoritmada rastgele seçilmiş bir şehirden başlanır. Ziyaret edilmemiş şehir kaldığı müddetçe turda henüz belirmemiş en yakın şehir ziyaret edilir. Son olarak ilk şehre geri dönülür.

Bu yaklaşım basit ancak genellikle gereğinden fazla açgözlüdür. Oluşturma sürecindeki ilk mesafeler makul kısalıktadır ancak sürecin sonundaki mesafeler genellikle daha ziyade uzun olacaktır. Bu soruna çare bulmak için pek çok başka tur oluşturma algoritması da geliştirilmiştir [73].

Bununla birlikte, en büyük başarıyı tur iyileştirme algoritmaları yakalamıştır. Bu tip algoritmaların temel bir örneği 2-opt algoritmasıdır. Verilen tur ile başlanır.

Turdaki iki bağlantı başka iki bağlantı ile yeni tur uzunluğu daha kısa olacak şekilde yer değiştirilir. Daha fazla iyileştirme yapılması mümkün olmayıncaya kadar bu şekilde devam edilir.

Bu yöntemin genelleştirilmesi simetrik GSP'yi çözmede en etkili yaklaşık algoritmalardan biri olan Lin-Kernighan algoritmasına temel teşkil etmektedir [2]. Lin ve Kernighan tarafından ilk olarak 1971 yılında gerçekleştirildiği şekliyle orijinal algoritma ortalama $O(n^{2.2})$ çalışma zaman karmaşıklığındadır ve 100 şehirden az şehirlik problemlerin çoğu için optimal çözümleri bulabilmektedir. Yine de işbu algoritmanın gerçekleştirimi kolay değildir. 1989'da yapılan bir çalışmada [74] yazarlar algoritmanın o tarihlerdeki başka hiçbir uygulamasının, Lin ve Kernighan tarafından elde edilen etkinliği veremediğini belirtmişlerdir.

2.4 Gezgin Satıcı Problemlerinde Kullanılan Meta-Sezgisel Yöntemler

GSP'nin çözümünde kullanılan, tek noktadan arama yapan Tepe Tırmanma, İteratif Yerel Arama, Tavlama Benzetimi, Tabu Arama ve Kanguru Algoritmaları ile popülasyon temelli yöntemlerden Yapay Arı Kolonisi, Genetik Algoritma ve Karınca Kolonisi Algoritması bölüm alt başlıklarında açıklanmıştır.

2.4.1 Tepe Tırmanma Algoritması (TT)

TT algoritması, yerel arama sınıfında yer alan iteratif bir optimizasyon yöntemidir. Tanımlanan kurallar doğrultusunda bir çözümden diğer komşu çözüme ulaşma temeline dayanmaktadır [75].

TT algoritmasında iyi bir komşuluk yapısı seçilmesinin, metodun etkinliğinde önemi büyüktür. Algoritmanın zayıf yanı yerel ve global arasında ayrım yapamamasından kaynaklı yerel optimuma takılmasıdır [76].

TT algoritmasında kopya işlemi ile başlangıç rotası hafızaya alınmakta ve tweak işlemi ile kopyalanan rota üzerinde değişiklikler yapılarak yeni bir rota elde edilmektedir. Algoritma boyunca mevcut rota ile oluşturulan rotanın kaliteleri (uygunluk) karşılaştırıldığı için rotaları bozmamak adına kopyalama işlemi yapılmaktadır [76].

2.4.2 İteratif Yerel Arama Algoritması (İYA)

İYA, matematik ve bilgisayar bilimlerinde uygulanan yerel arama, tepe tırmanma gibi metotların geliştirilmiş halidir.1980'lerde ortaya çıkmıştır. Yerel arama metotları bazen yerel minimum veya maksimuma takılabilir [77].

İYA'nın genel fikri; tüm çözüm uzayını aramaya odaklanmaktansa daha yerel optimumlar tarafından çevrili küçük bir alt uzaya odaklanmaktır. İYA ile bir optimizasyon problemiyle bir uygulamayı ele almak 4 temel birleşenle gerçekleşir.

Bunlar şu şekildedir;

1. Rassal bir başlangıç çözümü oluşturmak
2. Yerel arama
3. Perturb yapmak
4. Kabul kriterini uygulamak

Probleme rassal bir başlangıç çözümüyle başlanmaktadır. Yerel arama safhasında hafızaya kopyalanarak alınan başlangıç rotası üzerinde küçük bir değişiklik (tweak) yapılarak yeni bir çözüm elde edilmekte ve kalite olarak başlangıç çözüm ile karşılaştırılmaktadır. İzleyen aşamada mevcut rota üzerinde büyük değişiklikler yapan perturb işlemi yapılarak çözüm kalitesi artırılmaya çalışılmaktadır. Büyük değişiklikler ile ifade edilen işlem, yerel çözüm alanından sıçrama yaptırabilecek değişikliklerdir. Örneğin, GSP probleminde uzak kaydırma işleminin düğüm sayısına bağlı olarak bir defadan çok daha fazla yapılmasıdır.

Son olarak kabul kriteri uygulanarak döngü tamamlanmaktadır. Bu bileşenler arası etkileşimler hesaplanmalı ve metodun kalitesi arttırılmaya çalışılmadır [78].

İYA; grafik çizimlerinde, Steiner minimum yayılan ağaç problemi çözümünde, GSP'de ve maksimum gerçeklenebilirlik problemi gibi problemlerin çözümünde kullanılmıştır.

2.4.3 Tavlama Benzetimi Algoritması (TB)

TB algoritması, ilk olarak 1983 yılında Kirkpatrick, Gelatt ve Vechi [79] tarafından sunulmuş olup, optimizasyon problemlerinin çözümü için geliştirilmiş bir yerel arama algoritmasıdır [80][81].

TB algoritması adını erimiş metalin soğutulması işlemi olan, tavlama işleminden almaktadır [82]. Bu işlemde metalik yapıdaki kusurları azaltmak için bir materyal ısıtılır, daha büyük kristal boyuta ve minimum enerji ile katı kristal duruma yavaşça soğutulur. Tavlama işlemi, sıcaklığın ve soğuma katsayısının dikkatlice kontrolünü gerektirir [80]. Tavlama işlemi sonucunda oluşan kristalleşme, metalin mekanik özelliklerini iyileştiren moleküler yapısındaki değişikliklerle oluşmaktadır [81]. Tavlama işlemindeki ısının davranışı, optimizasyondaki kontrol parametresiyle aynı gibi görülür. Isının, daha iyi sonuçlara doğru algoritmaya rehberlik eden bir rolü vardır. Bu durum ancak kontrollü bir tutum içinde, ısının kademeli olarak düşürülmesiyle yapılabilir. Eğer ısı aniden düşürülürse, algoritma yerel minimum ile durur [83].

TB algoritması; birçok değişkene sahip fonksiyonların maksimum veya minimum değerlerinin bulunması için, özellikle de birçok yerel minimuma sahip doğrusal olmayan fonksiyonların minimum değerlerinin bulunması için tasarlanmıştır [81].

Tavlama benzetimi algoritmasında; sıcaklık, sıcaklığın düşürülmesi, tekrar işlemi (döngü) gibi parametreler vardır. Algoritmada bir başlangıç aday çözümü belirlenip, bu çözüm başlangıçta en iyi çözüm olarak kabul edilir.

Başlangıç aday çözümünün kopyasına "tweak" işlemi (başlangıç aday çözümüne ufak bir değişiklik yaparak yeni bir çözüm üretme işlemi) uygulanır. Daha sonra ki adımlar da ise en iyi çözüm kabul edilen başlangıç aday çözümü ile yeni üretilen çözümün hangisinin daha iyi olduğu (kaliteleri) veya 0-1 arasında üretilen bir rassal sayının < $e^{((\text{Kalite(Y)}-\ \text{Kalite(B)})/\text{sıcaklık})}$ durumu araştırılır [82]. Eğer yeni sonuç daha iyi ise en iyi çözüm olarak yeni çözüm atanır. Sıcaklık parametresi, tavlama yönteminde başlangıçta belirlenen bir değerdir. Bu değer, oluşturulan döngünün sonunda, belirli bir oranda (sıcaklığın düşürülme parametresi kadar) azaltılır. Bu işlemler en iyi çözüm bulunana kadar veya algoritmanın çalıştırılması için belirlenen bir süre bitene kadar veya sıcaklık parametresi sıfır veya sıfırdan küçük olana kadar sürdürülebilir.

2.4.4 Tabu Arama Algoritması (TA)

Glover [84] tarafından ortaya atılan ve Hansen [85] tarafından türetilmiş versiyonları bulunan TA algoritması, temelde son çözüme götüren adımın dairesel hareketler oluşturmasını engellemek için cezalandırılarak bir sonraki döngüde tekrar edilmesinin yasaklanması üzerine kurgulanmıştır [86].

TA algoritması döngü ya da çalışma süresi boyunca üretilen çözümler ile ilgili bilgileri saklamak üzere tasarlanmış dinamik bir hafızaya sahiptir. Tabu listesi olarak da adlandırılan bu hafızada saklanan bilgiler, araştırma uzayında yeni çözüm kümelerinin oluşturulması için kullanılır [87]. TA algoritması, mevcut çözümlerden küçük bir değişim ile (tweak) elde ettiği denenmemiş bir çözüm kümesi üreterek optimizasyona başlamaktadır.

TA algoritmasında, çözüm uzayında bir yerel minimum noktada takılmayı engelleyebilmek için oluşturulan yeni çözüme, o anki çözümden daha kötü olsa bile müsaade edilmektedir. Ancak kötü çözüme izin verilmesi algoritmayı bir kısır döngü içerisine sokabilecektir. Algoritmanın kısır döngüye girmesine engel olmak için, bir tabu listesi oluşturulur ve o anki çözüme uygulanmasına izin verilmeyen tüm yasaklı hareketler tabu listesinde saklanır.

Bir hareketin tabu listesine alınıp alınmayacağını belirlemek için, tabu kısıtlamaları adı verilen bazı kriterler kullanılmaktadır. Tabu listesinin kullanılması, belirli bir sayısınca daha önce denenmiş çözümlerin tekrar edilmesini engellediği için arama esnasında bir bölgede takılma ihtimalini azaltmaktadır [88].

TA, mümkün bir çözüm ile başlar. Bu çözüm, problemin matematiksel ifadesinde geçen kısıtları tatmin eden bir çözümdür. Tabu aramanın performansı başlangıç çözümüne bağlıdır. Bu nedenle mümkün olduğunca iyi olan bir çözüm ile başlamak gerekir.

Tabu listesi ilk giren ilk çıkar (first in first out = FIFO) mantığında çalışan bir listedir. Algoritmada belirlenen karakteristik özelliklere göre tabu listesi sürekli olarak yukarıdan doldurulmaya başlar. Bulunan elemanların sayısı liste uzunluğunu (/) aştığında yeni gelen elemanın listeye eklenmesi için listenin en sonundaki eleman listeden çıkarılır [82].

2.4.5 Kanguru Algoritması (KA)

KA, ilk olarak Pollard [89] tarafından kesikli logaritmik optimizasyon problemlerinin çözümü için geliştirilmiş bir algoritmadır. Van Oorschot ve Wiener [90] yılında yapmış oldukları çalışmada Pollard tarafından ileri sürülen kanguru metodunun paralel halini geliştirmişlerdir.

Bir iteratif iyileştirme metodu olan KA; TA, TB gibi sezgisellerden ilham alınarak geliştirilmiştir. Komşu arama algoritması olarak da sınıflandırılan algoritmada, kesikli optimizasyon problemlerinin çözümünde belli bir noktadan çözüme başlamakta, yakın komşuluklar belli bir komşuluk fonksiyonuna göre aranmaktadır. Bu aşamaya descent denmektedir.

Başlangıç çözümü *u*, komşu çözümler *u'* dür. Mevcut en iyi çözüm *u** dür ve aranan komşu çözümlerde *u** bulunursa bu çözüm *u* yerine geçmektedir. Belirli sayıda (A) iterasyonun tamamlanması halinde amaç fonksiyonu değerinde iyileşme gözlemlenmiyorsa algoritmanın ikinci adımına geçilir. İkinci adım jump olarak adlandırılır. jump prosedürü çözümün yerel optimuma takılmasına engel olmak için geliştirilmiştir. Bu aşamada çözüm uzayının farklı noktalarını taramak için çözüm seti tesadüfi olarak değiştirilmektedir. Belirlenen adım sayısı kadar gelişme olmazsa bulunan en iyi jump noktasından yakın komşu arama prosedürüne, descent, geçilir. Her iki prosedürde de mevcut en iyi çözümden daha iyisi bulunduğunda sayaç (*t*) sıfırlanır ve *t*, A'dan büyük olana kadar arama adımlarına devam edilir [91].

A parametresi, mevcut çözümde herhangi bir gelişme olmaksızın gerçekleştirilebilecek maksimum iterasyon sayısıdır. descent prosedüründe, A kadar iterasyonda gelişme olmazsa jump prosedürüne geçiş yapılır. Aynı şekilde jump prosedüründe A kadar iterasyonda iyileşme olmazsa descent prosedürüne dönülür. Bu süreç KA' ya ait bir döngü olup durdurma kriteri sağlanan kadar devam eder [92].

2.4.6 Yapay Arı Kolonisi Algoritması (YAK)

YAK algoritması, arılardaki yiyecek arama davranışları temel alınarak geliştirilen bir algoritmadır [93]. Doğada arılar yiyecek kaynaklarından nektar toplama, bulunan nektarı zamanı ve yolu minimize ederek en verimli şekilde kovana getirme işini içgüdüsel olarak yaparlar. Kaliteli bir yiyecek kaynağı bulan arılar, buldukları kaynaklardan toplayabildiği miktardaki nektarı kovana getirdikten sonra tekrar kaynağa dönmeden önce "dans alanında" (dancing area) "salınım dansı" (waggle dance) ile kendi kaynağı hakkındaki yön, uzaklık ve nektar miktarı bilgilerini diğer arılarla paylaşırlar. Bu başarılı mekanizma sayesinde koloni, kaliteli yiyecek kaynaklarının olduğu bölgelere yönlendirilebilmektedir. YAK sistemi, üç temel bileşenden oluşmaktadır. Bunlar, yiyecek kaynakları, arılar ve kovan.

Yiyecek kaynakları, arıların yiyecek bulmak için gittikleri kovan etrafındaki nektar kaynaklarıdır.

Bir yiyecek kaynağının değeri, kaynağın çeşidi, yuvaya olan uzaklığı, nektar miktarı veya nektarın çıkarılmasının kolaylığı gibi birçok faktöre bağlıdır. Yiyecek kaynakları optimize edilmeye çalışılan problemin olası çözümlerine karşılık gelmektedir [58]. Bir kaynağa ait nektar miktarı o kaynakla ifade edilen çözümün kalite değerini ifade eder.

YAK Algoritmasında bir kolonide üç grup arı bulunmaktadır: işçi arılar, gözcü arılar ve kaşif arılar [58]. Kâşif Arılar, arı sistemi içerisinde tamamen bağımsız davranarak herhangi bir ön bilgi kullanmadan yeni yiyecek kaynaklarını arayan arı grubudur. Bir kovandaki kâşif arıların kovandaki tüm arılara oranı %5-10 civarındadır [93]. Rasgele yiyecek kaynağı ararlar. Buldukları yeni yiyecek kaynağı bir önceki yiyecek kaynağından daha iyi ise, yeni yiyecek kaynağını hafızalarında güncellerler. Kovana dönerler, dans alanında salınım dansı yaparak gözcü arılarla yeni yiyecek kaynağının bilgilerini paylaşırlar. İşçi Arılar, belirli bir yiyecek kaynağından nektar getiren arılardır. Mevcut durumda nektar kaynağından faydalanmaya ve çalışmaya devam ederler. Aynı zamanda nektar topladıkları yiyecek kaynaklarına giderken komşu yiyecek kaynaklarını (komşu çözümler) da araştırırlar. Belirlenen hata olasılığına [94] göre karşılaştıkları yeni daha iyi ya da daha kötü yiyecek kaynağını kabul eder veya reddederler.

Eğer yeni çözümü kabul ederlerse ziyaret sayaçları sıfırlanır ve yeni yiyecek kaynağı hafızalarında güncellenir. Kovana döndüklerinde dans alanında salınım dansı yaparak gözcü arılarla yeni yiyecek kaynağının bilgilerini paylaşırlar. Aynı yiyecek kaynağına her gittiklerinde ziyaret sayaçları bir artırılır. Arının ziyaret sayısı, verilen maksimum ziyaret sayısını aştığında arı kaynağı terk eder ve gözcü arıya dönüşür. Gözcü Arılar, kâşif ve işçi arıların salınım dansını izleyerek yiyecek kaynakları hakkında bilgi edinen ve daha sonra bu bilgiye göre hareket ederek yiyecek kaynaklarına ulaşan arılardır. Daha iyi yiyecek kaynağı bulan kâşif veya işçi arılar dans alanında salınım dansı yaptığında gözcü arılar yeni yiyecek kaynağını belirlenen ikna olasılığına [94] göre kabul eder veya reddederler. Kabul ettikleri takdirde yeni çözüm hafızalarında güncellenir.

Kovan, arıların barındıkları, yiyeceklerini depoladıkları ve bilgi paylaştıkları yapılardır. Arılar arasındaki bilgi değişimi ortak bilginin sağlanmasındaki en önemli olaydır. Bu bilgi değişiminin gerçekleştiği dans alanı kovanın en önemli bölümüdür.

Dans alanında yapılan salınım dansı ile yiyecek kaynaklarının yer ve kalite bilgileri paylaşılır [94].

2.4.7 Genetik Algoritma (GA)

GA doğrusal olmayan, çok değişkenli, zor problemlerin çözümü için geliştirilmiş, popülasyon temelli sezgisel bir yöntemdir [95],[96]. Önbilgi ve varsayımlar olmadan, amaç fonksiyonu ile çalışabilmektedir. Problem değişkenleri, kromozom denen dizilerde, genlerle temsil edilmektedir. Her bir değişken kodlama biçimine bağlı olarak tek ya da bir grup genle tanımlanmaktadır. Seçim, çarprazlama ve mutasyon olarak adlandırılan genetik operatörlerle iterasyonlar boyunca kromozomlarda bir takım değişiklikler yapılmakta ve en iyi çözüm seti aranmaktadır.

GA'da ilk olarak kodlama biçimine karar verilir. Genellikle ikili kodlama, permutasyon kodlama ve gerçek değerli kodlama kullanılmaktadır. İkili kodlama 1 ve 0 değerlerinden oluşup değişkenler değer aralığına göre belirlenen sayıda genden oluşan ikili düzende temsil edilmektedir. Permutasyon kodlama, sıralamanın önemli olduğu ve tekrarın mümkün olmadığı, en kısa yol, gezgin satıcı, araç rotalama vb. problemlerin çözümünde kullanılır. Gerçek değerli kodlama ise değişkenlerin doğrudan kendi değerleriyle temsil edildikleri kodlama biçimidir.

Popülasyon temelli olan GA'da çözüm uzayındaki arama tek bir noktadan değil, noktalar kümesinden yapılmaktadır. Uygulayıcı tarafından belirlenen miktardaki kromozom, popülasyonuoluşturmaktadır.Çözüme tesadüfiolarak veya basit tekniklerle oluşturulabilen başlangıç populasyonu ile başlanır. Genlerdeki değişken değerleri, fonksiyonda yerine konularak kromozomun uygunluk değeri elde edilir. Genetik operatörlerden ilk olarak seçim operatörü uygulanır. Amaç, popülasyonda daha iyi bireylerin çoğaltılması, uygunluğu düşük olan bireylerin elenmesidir. Birçok seçim yöntemi vardır. Bunlara, rulet tekerleği seçimi, turnuva seçimi, genel stokastik örnekleme ve sıralı seçim örnek olarak verilebilir [97]. Seçim sonrası çaprazlama operatörü uygulanmaktadır. Çaprazlamada amaç, iki bireyin farklı birtakım özelliklerini taşıyan ve daha iyi bireyler elde etmektir. Böylelikle, uygunluğu daha yüksek çözüm alternatifleri üretilmeye çalışılır.

İkili kodlamada genellikle tek nokta [98], iki nokta ve çok noktalı çaprazlama, permutasyon kodlamada pozisyona dayalı, sıralı [95] ve dairesel çaprazlama [99] kullanılmaktadır. Gerçek değerli kodlamaya ise aritmetik çaprazlama, kesikli üretim, çizgi üretim örnek olarak verilebilir. Mutasyon operatörü, bir daha ulaşılması mümkün olmayan çözümlerin kaybına karşı koruma sağlamaktadır [95]. Düşük bir olasılıkla herhangi bir gen üzerinde yapılan tesadüfi değişikliklerdir. İkili düzende, genin değeri 1 ise 0'a, 0 ise 1'e dönüştürülmesi şeklinde gerçekleştirilmektedir. Permutasyon kodlamada yakın kaydırma, uzak kaydırma, toplu kaydırma, tesadüfi değişim, sıralı değişim gibi birçok mutasyon çeşidi vardır. Gerçek değerli kodlamada ise mevcut değişken değerinin belirlenen mutasyon adımı miktarınca azaltılması veya eşit olasılıkla arttırılması şeklinde mutasyon uygulanmaktadır.

Genetik operatörler başlangıç popülasyonuna uygulanır ve yeni bir jenerasyon elde edilir.

Tamamlanma kriteri sağlanana kadar genetik operatörlerle yeni popülasyonlar üretilir. Döngü tamamlandığında algoritma durdurulmakta ve mevcut en iyi çözüm sonuç olarak belirlenmektedir [100].

2.4.8 Karınca Kolonisi Algoritması (KKA)

KKA, karıncaların yön seçme duyularından ve besin kaynaklarına ulaşma mantıklarından esinlenerek geliştirilmiş bir metasezgisel yöntemdir. Gerçek karıncaların yuvaları ile yiyecek topladıkları noktalar arasındaki mesafeyi en küçükleyen rotayı, salgıladıkları feromon kimyasalı sayesinde belirlemeleri üzerine kurgulanmıştır [101].

Gerçek karıncalar yiyecek aramak üzere yuvalarından ayrıldıklarında izleyecekleri rotayı yoldan daha önce geçen karıncaların salgıladıkları feromon sayesinde Feromon karıncaların bacaklarından salgılanan bir kimyasal olup, belirli bir süre boyunca yol üzerinde kalmakta ve zamanla buharlaşarak yok olmaktadır.

Başlangıçta rassal olarak yiyecek aramaya çıkan karıncalardan en kısa mesafeli yolu kullanan karıncalar daha kısa sürede yuvaya dönüşe geçerek, yol üzerine daha fazla feromon bırakmış olmaktadır.

Yiyecek arama işlemi devam ettikçe, kısa mesafeler üzerinde feromon miktarı yoğunlaşmakta, sürenin de kısalığına paralel olarak feromon buharlaşma oranı da azalmaktadır. Aynı şekilde daha uzun mesafeli rotalar üzerinde başlangıçta daha az olan feromon nedeniyle karıncalar tarafından tercih edilme oranı azalmakta ve bir süre sonra yol üzerinde bulunan feromon tamamen buharlaşarak hiçbir karıncanın kullanmayacağı bir yol olmaktadır [102]. Gerçek karıncaların yiyecek arama davranışlarından esinlenerek geliştirilen KKA ilk olarak 1992 yılında yayımlanan bir doktora tezinde [103] GSP üzerinde uygulanarak literatüre geçmiştir. Yapay karıncalardan oluşan KKA'da başlangıç çözümü olarak karıncalar rassal yollar üzerinden turlarına başlamakta ve bir turu tamamlayarak yuvaya dönmektedirler. Tur boyunca karıncaların takip ettikleri yollar üzerine feromon eklemesi yapılmaktadır.

İlk iterasyonun tamamlanmasının ardından ikinci iterasyonda daha kısa mesafelerde nispeten daha fazla feromon bulunacağı için yapay karıncaların bir kısmı izledikleri rota yerine daha çok feromon bulunan yeni rotayı takip etmeye başlayacaktır. Her iterasyon sonunda yollar üzerinde bulunan feromon miktarları kullanıcının belirlediği bir oranda buharlaştırılarak local güncelleme yapılmaktadır. Bu sayede çok kötü çözümlere izin verilmediği gibi, feromon miktarının nispeten fazla olduğu rotalar da en iyi çözüm olarak kabul edilmemektedir. Bu durum algoritmanın yerel optimumlara takılmasına mani olmaktadır.

Global feromon güncellemesi aşamasında ise en iyi çözümü üreten rota ek feromon eklemesi yapılmak suretiyle yapay karıncalar için daha cazip kılınmaktadır [104].

BÖLÜM 3

ELEKTROMANYETİZMA SEZGİSELİ TABANLI ÇÖZÜM YAKLAŞIMI

Geçtiğimiz yıllarda global optimizasyon hızla gelişen bir alan haline geldi. Gerçek hayatta, fizik, kimya moleküler biyoloji [105],[106] gibi alanlarda karşımıza çıkan problemler doğrusal olmayan fonksiyonlar ve birçok değişkeni yapısında bulundurmaktadır. Bu problem tipleri konvansiyonel matematiksel yöntemlerle çözümü zor yapılardır. Çünkü nitelikleri itibariyle değişkenlik ve kesikli yapılar içermektedirler.

Yukarda bahsedilen zorlukların üstesinden gelmek üzere Stokastik arama yöntemleri geliştirilmiş ve özellikle 1980'li yıllar itibari ile bilgisayar gücünden yararlanılmaya başlanmıştır. Rassal arama algoritmaları, yüksek boyutlu ve düzenli olmayan problemlerde diğer algoritmalara göre daha kullanılabilir ve tutarlı sonuçlar vermiştir.

Bu bölümde; orijinal elektromanyetizma sezgiseli ve rassal anahtar yöntemi ile ilgili detaylı bilgi sunulmaktadır. Kısıtsız optimizasyon problemlerinin çözümünde başarılı bir algoritma olan elektromanyetizma sezgiseli, rassal anahtar yöntemiyle birleştirilerek kesikli ve kısıtlı optimizasyon problemlerinin çözümü için elverişli hale getirilmiştir.

3.1 Elektromanyetizma Sezgiseli Genel Yapısı

Birbil ve Fang [11] EM Sezgiselini içinde sınırlandırılmış değişken yapılarını içeren özel bir optimizasyon sınıfı için önermişlerdir.

$$Min\ f(x)$$

$$X\ \in\ [l,u]$$

$$[l,u] = \{x\ \in\ R^n | l_k \leq\ x_k \leq u_k, k = 1, \dots n\}.$$

Stokastik optimizasyon yöntemlerinde, popülasyon tabanlı algoritmalar genellikle fizibil bölgede tanımlanan rassal noktaların belirlenmesi ile başlar. Belirlenen noktaların sahip oldukları amaç fonksiyon değerlerine göre birbirleriyle etkileşim faaliyetleri belirlenir ve kullanılacak algoritmanın amaç fonksiyon değerini iyileştirici mekanizmaları devreye girer. Genetik Algoritmadaki bu mekanizmalar, reprodüksiyon, çaprazlama ve mutasyon [107] işlemleridir.

Benzer şekilde, EM algoritmasında parçacıkların çekici vadilere doğru yakınsandıkları, itici dik tepe bölgelerden de uzaklaştırıldıkları bir mekanizma kullanılmıştır. Parçacıkların daha iyi çözüm alanlarına doğru hareket etmelerini sağlama fikrinden yola çıkarak fizikteki elektromanyetizma teorisini baz alan bir analoji kullanılmaktadır.

Elektromanyetizma teorisine benzer şekilde her örnek nokta; fizibil uzayda elektriksel bir yükse sahip parçacıklar olarak ele alınmaktadır. Bu yaklaşımda, her bir parçacığın sahip olduğu amaç fonksiyon değerine göre bir yük değeri vardır ve amaç bu amaç fonksiyonunun optimize etmektir.

Ayrıca parçacıkların sahip oldukları amaç fonksiyon değerleri o parçacığın, uzayda bulunan parçacık popülasyonundaki diğer parçacıklara uyguladığı itme ve çekme kuvvetinin büyüklüğünü de belirlemektedir. Yani amaç fonksiyonu daha yüksek olan parçacığın oluşturduğu çekme kuvveti daha büyüktür.

Hesaplanan yük değerleri mevcut parçacıkların, sonraki itereasyonlarda ne yönde hareket ettiklerini saptamak için kullanılır. Bir parçacık üzerine etkiyen kuvvetlerin bileşkesi, o parçacığın ne tarafa doğru hareket edeceğini de belirler. Tıpkı elektromanyetizma teorisinde olduğu gibi bu yöntemde de kuvvetler vektöreldir ve her bir parçacık üzerine etkiyen kuvvetler ayrı ayrı hesaplanır.

Özellikle belirtmek gerekirse bu sezgiselin fizikteki elektromanyetizma teorisi ile yüksek benzerliğe sahip olduğu gibi bir takım farklılıkları da vardır. Bu farklılıklara, algoritma açıklanırken yeri geldikçe değinilecektir.

Son olarak, popülasyon tabanlı algoritmalara benzer şekilde [108],[109], uzayda bulunan parçacıklara bir yerel arama prosedürü uygulanarak, her iterasyonda daha iyi bir amaç fonksiyon değeri elde edilebilmektedir.

Algoritmanın en temel parametreleri şunlardır:

n: problemin boyutu

u_k: k. boyutun üst sınırı

l_k: k. boyutun alt sınırı

f(x): minimize edilmek istenen amaç fonksiyonu

Elektromanyetizma sezgiseli 4 aşamadan oluşmaktadır. Bunlar, *Başlangıç*, her bir parçacığa uygulanan *toplam kuvvetin hesaplanması*, uygulanan kuvvetin doğrultusunda parçacığın *hareket* etmesi ve daha iyi bir yerel minimum bulma amacıyla kullanılan yerel arama prosedürüdür.

Şekil 3.1 Elektromanyetizma Sezgiseli Genel Akışı

EM Sezgiselinin genel yapısı Algoritma 1 olarak gösterilebilir.

ALGORİTMA 1. EM(*m*, *MAXITER*, *LSIT ER*, *δ*)

m: uzaydaki parçacık sayısı

MAXITER: maksimum iterasyon sayısı

LSITER: maksimum yerel arama sayısı

δ: yerel arama parametresi, $\delta \in [0, 1]$

1: Initialize()
2: iteration ←1
3: **while** iteration <*MAXITER* **do**
4: Local(*LSIT ER*, *δ*)
5: **F** ←CalcF()
6: Move(**F**)
7: iteration ←iteration + 1
8: **end while**

3.1.1 Başlangıç

Başlangıç noktası, fizibil alan içerisinde rassal oalrak *m* adet noktanın seçilmesi ile başlar. Fizibil alan denilen bölge, n boyuta sahip hiper-kübik bir yapıdadır. Yani 3 boyuttan daha fazla fiziksel boyuta sahip olabilir. Her bir parçacığın koordinatları boyutların alt ve üst sınırları arasında değişen rassal bir değere sahiptir. Bir parçacık uzayda oluşturulduktan sonra o noktacığın amaç fonksiyon değeri hesaplanır(Algoritma 2, Satır 6). Başlangıç prosedürü m adet noktanın belirlenip içlerinde en iyi amaç fonksiyonuna sahip olanı (x^{best}) tespit edilerek sonlandırılır (Satır 8).

ALGORITHM 2. Initialize()

1: **for** i = 1 to m **do**
2: **for** k = 1 to n **do**
3: λ ← U (0, 1)
4: xik← lk + λ(uk − lk)
5: **end for**
6: Calculate f (xi)
7: **end for**
8: xbest ← argmin{f (xi), ∀i}

3.1.2 Yerel arama

Yerel arama prosedürü x_i parçacığı için yerel bilgi toplama amaçlı uygulanır. Prosedür içinde geçen *LSITER* ve δ parametreleri ise yerel arama sayısı ve komşu arama çarpanı anlamına gelir.

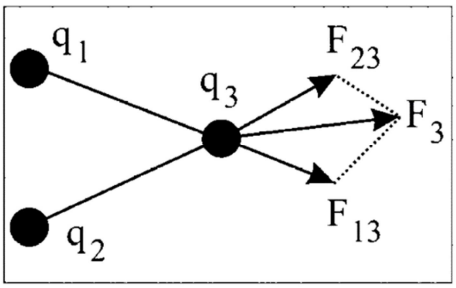
Yerel arama prosedürünün çalışma şekli şöyledir: Öncelikle δ parametresine göre, maksimum fizibil adım uzunluğu *(Length)* hesaplanır (Algoritma 3, Adım 2). Daha sonra, her bir *k* boyutunda x^i için adım adım arama yapılır (Adım 5-13).

Seçilen bir koordinat için x^i geçici bir değişken y içine atanır. Bu geçici değişken parçacığın sahip olduğu ilk amaç fonksiyon değerinin saklanması açısından önemlidir. Çünkü yerel arama prosedürü kullanılırken, arama esnasında parçacığa komşu olan alanların amaç fonksiyon değeri ile parçacığın ilk amaç fonksiyon değeri karşılaştırılacaktır.

Daha sonra, üretilen bir rassal sayı adım uzaklığı olarak atanır ve *y* noktası bu doğrultuda hareket eder. *LSITER* adet iterasyon içinde *y* noktası daha iyi bir nokta keşfederse x^i devreye girerek *y'*nin yerleşmiş olduğu daha iyi noktaya yerleşir ve i. Parçacık için yerel arama safhası sona ermiş olur (satır 14-17). Böylelikle *geçerli en iyi nokta* x^{best} güncellenmiş olur (satır 22).

İşin özünde yerel arama algoritması bir parçacığın sahip olduğu amaç fonksiyon değeri için her bir koordinat için doğrusal arama yapan bir yöntemdir. Yani ilk başta bir parçacık seçilen tek bir koordinatta adım adım hareket ettirilir ve her bir adımda amaç fonksiyon değerinin değişim durumu kontrol edilir. İyileşme varsa parçacık o noktaya yerleşir. Daha sonra aynı parçacık bir diğer koordinat üzerinde arama yapmaya başlar. Yerel arama algoritması her bir parçacık için *n* adet koordinat için de tekrarlanır. Yapısı itibariyle bu prosedür yerel arama yapmak için gradyan bilgisine ihtiyaç duymaz. Yola çıkmış olduğu nokta ile varmış olduğu nokta arasındaki fonksiyon değerlerini karşılaştırır. Bu aşama için aslında [110] çok daha güçlü yerel arama yöntemleri kullanılabilir. Fakat optimizasyonun aslını elektromanyetizma yöntemi gerçekleştireceğinden dolayı bu aşamada daha basit bir yerel arama yöntemi kullanılmıştır.

Önemli olan, her bir iterasyon adımında, mevcut durumdan daha iyi bir durumun var olup olmadığının araştırılmış olmasıdır. Bu sebeple daha basit yapıdaki bir arama prosedürünün kullanılması algoritmanın toplam çalışma süresi açısından avantaj sağlamaktadır.



Şekil 3.2 Süperpozisyon ilkesi

ALGORITHM 3. Local(LSITER, δ)

1: counter ←1
2: Length← δ(maxk{uk − lk})
3: for i = 1 to m do
4: for k = 1 to n do
5: λ1 ← U (0, 1)
6: while counter <LSITER do
7: y ← xi
8: λ2 ← U (0, 1)
9: if λ1 > 0.5 then
10: yk ← yk + λ2(Length)
11: else
12: yk ← yk − λ2(Length)
13:end if
14: if f (y) < f(xi) then
15: xi ← y
16: counter← LSITER − 1
17: end if
18: counter ←counter + 1
19: end while
20: end for
21: end for
22: xbest ← argmin{f (xi), ∀i}

3.1.3 Toplam Kuvvet Vektörünün Hesaplanması

Elektromanyetizma teorisindeki süperpozisyon ilkesine göre, manyetik parçacıkların birbirlerine uygulamış oldukları kuvvet, parçacıkların birbirlerine olan uzaklıkları ile ters orantılıdır. Yani birbirine kuvvet uygulayan iki parçacık arasındaki mesafe artarsa uygulanan kuvvet azalır, iki parçacık arasındaki mesafe azalırsa parçacıkların birbirlerine uyguladıkları kuvvetler artar. Bununlar birlikte, elektromanyetik kuvvetin büyüklüğü birbirlerine kuvvet uygulayan parçacıkların sahip oldukları yükün büyüklüğü ile doğru orantılıdır. Büyük yükler büyük, küçük kuvvetler ise küçük kuvvet uygularlar (Cowan, 1968).

Elektromanyetizma algoritmasını temel elektromanyetizma teorisinden farklı kılan taraflardan bir tanesi de popülasyon içerisindeki parçacıkların sahip oldukları yük değerlerinin sabit olmamasıdır. Zira her bir iterasyonda parçacıkların sahip oldukları yük değeri amaç fonksiyonlarına göre tekrar hesaplanır. Parçacıkların sahip oldukları yük değerleri, hem kendi amaç fonksiyonlarına hem de en iyi parçacığın amaç fonksiyon değerine bağlıdır.

Popülasyon içindeki i. parçacığın sahip olduğu yük değeri (q^i) , o parçacığın sahip olduğu itme ve çekme gücünü gelirler.

$$q^i = \exp\left(-n \frac{f(x^i) - f(x^{best})}{\sum_{k=1}^{m}(f(x^i) - f(x^{best}))}\right) \tag{3.1}$$

Daha iyi amaç fonksiyonuna sahip olan nokta, diğer noktalara göre daha yüksek bir yük değerine sahip olmaktadır. Yük hesaplama formülü içinde bulunan kesirli kısım n ile çarpılmaktadır. Bunun sebebi, büyük boyutlu problemlerde, kullanılacak parçacık sayısının yüksek olması sebebiyle kesirli kısım çok küçük çıkacaktır ve bu durum formülün üssünü alırken taşma sorunu doğuracaktır. Bu sorunun önüne geçmek için kesir n büyüklüğü ile çarpılmaktadır.

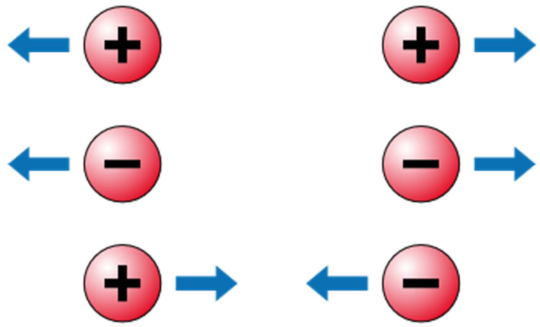
Popülasyon içindeki her bir parçacığın sahip olduğu q^i yük değerini, sahip olduğu amaç fonksiyon değerinin kısmi etkisi olarak tanımlanabilir, aynı amacı sağlayan alternatif bir formül de yük hesabında kullanılabilirdi. Fakat yapılan testler neticesinde bu formülün, algoritmanın işleyişi açısından son derece yeterli olduğu görülmüştür.

Özellikle dikkat edilmelidir ki, temel elektromanyetizma teorisinden farklı olarak, bu denkleme göre bir yükün negatif çıkma ihtimali yoktur. Bununla birlikte, iki parçacığa etkiyen kuvvetlerin yönlerine o parçacıkların amaç fonksiyon değerleri karşılaştırılarak karar verilir. Bundan dolayı, bir parçacığa etki eden kuvvetin hesaplanması için aşağıdaki eşitlik kullanılmaktadır:

$$F^i = \sum_{j \neq i}^{m} \begin{cases} (x^i - x^j)\dfrac{q^i q^j}{\|(x^i - x^j\|^2} & if\ f(x^j) \leq f(x^i) \\ (x^j - x^i)\dfrac{q^i q^j}{\|(x^i - x^j\|^2} & else \end{cases} \quad ,\forall i \qquad (3.2)$$

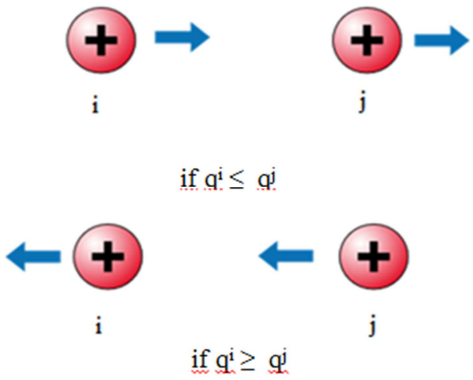
2 parçacıktan amaç fonksiyon değeri daha iyi olan diğerini çeker (Algoritma 4, satır 7-8). 2 parçacık içinde daha kötü bir amaç fonksiyonuna sahip olan parçacık ise diğerini iter (Algoritma 4, satır 9-10). X^{best} en iyi amaç fonksiyon değerine sahip parçacık olduğu için mutlak çekim noktası gibi hareket eder ve popülasyon içindeki diğer tüm noktaları çeker.

Denklem (3.2), j. parçacığı tarafından i. parçacığı üzerine uygulanan kuvvetin büyüklüğünün bu iki parçacık arasındaki Öklid mesafesi ile ters orantılı olduğunu göstermektedir. Bununla birlikte uygulanan kuvvetin büyüklüğü parçacıkların sahip oldukları yük büyüklükleri ile doğru orantılıdır. Yani yük değeri yüksek parçacıklar arasında yüksek, düşük olan parçacıklar arasında ise düşük kuvvetler oluşur. EM algoritmasının kullanmış olduğu itme çekme sistemi elektrik yüklü parçacıkların elektrostatik etkileşimini açıklayan Coulomb yasasından biraz farklıdır. Elektrik yüklü iki parçacığın birbirlerine uyguladıkları yükler Şekil 3.3'de görüldüğü gibi her zaman ters yönlüdür:



Şekil 3.3 Coulomb Yasası'na göre itme-çekme sistemi

Coulomb Yasası'na göre aynı yüklü parçacıklar birbirlerini iterlerken, farklı yüklü parçacıklar birbirlerini çekmektedirler. Bu durum EM sezgisel algoritmasında daha farklıdır. Algoritmanın doğası gereği elektrik yüklü parçacıkların birbirlerine uyguladıkları yük vektörleri her zaman aynı yöndedir. Coulomb Yasası'ndan farklı olarak EM algoritmasında yük değeri yüksek olan parçacık diğer parçacığı her zaman kendine doğru çeker bununla birlikte yük değeri düşük olan parçacık ise yüksek yük değerine sahip olan parçacığı itmek isteyecektir. Şekil 3.4'de de görüldüğü gibi j. parçacığın yük değerinin i. parçacığın yük değerinden büyük olduğu durumlarda j parçacığı i parçacığını çekerken, i parçacığı da j parçacığını itmektedir ve parçacıkların birbirlerine uygulamış oldukları itme-çekme kuvvet vektörlerinin yönleri aynıdır.



Şekil 3.4 EM algoritmasına göre itme-çekme sistemi

ALGORITHM 4. CalcF():**F**

1: **for** i = 1 to m **do**
2: $q^i \leftarrow \exp\left(-n \frac{f(x^i)-f(x^{best})}{\sum_{k=1}^{m}(f(x^i)-f(x^{best}))}\right)$
3: Fi ← 0
4: **end for**
5**: for** i = 1 to m **do**
6: **for** j = 1 to m **do**
7: **if** f (xj) < f(xi) **then**
8: $(x^i - x^j)\frac{q^i q^j}{\|(x^i-x^j\|^2} \quad \{Attraction\}$
9:**else**
10:$(x^j - x^i)\frac{q^i q^j}{\|(x^i-x^j\|^2} \quad \{Repulsion\}$
11: **end if**
12: **end for**
13: **end for**

3.1.4 Toplam Kuvvete Göre Hareket Etme

Her *i* parçacığı için toplam F^i hesaplandıktan sonra, *i* parçacığı kendine uygulanan bileşke kuvvetin yönünde Denklem (3.3)'te verilen rassal adım uzunluğu (λ) kadar hareket eder. (λ) sayısı 0 ve 1 arasında düzgün dağılan bir rassal sayıdır. Adım uzunluğunun hesaplanmasında daha farklı dağılımlar da mutlaka kullanılabilir fakat hesap kolaylığı teşkil etmesi açısından bu çalışmada uniform dağılım seçilmesi uygundur.

Adım uzunluğunun rastgele seçilmesinin en önemli sebeplerinden biri, her iterasyonda aynı adım uzunluğunun kullanmak fizibil uzayda uğranmayan alan kalma riski oluşturur. Dolayısı ile bütünüyle rastgele olarak geçilen adım uzunluğu sayesinde önceden belirlenen fizibil uzayın her bir noktası aynı ziyaret edilme imkanına sahip olunur.

Denklem (3.3) teki RNG, ilgili eksen üzerinde, eksenin üst sınırı u^k ve alt sınırı l^k arasında fizibil hareket etmeyi sağlayan vektör olarak tanımlanmaktadır. (Algoritma 5, Satır 6-10)

Bununla birlikte her bir parçacık üzerine etkiyen kuvvetler, fizibilite kurallarının sağlanabilmesi için normalize edilir. Böylece,

$$x^i = x^i + \lambda * \frac{F^i}{\|F^i\|}(RNG) \quad \forall i \tag{3.3}$$

5. Algoritmada *Hareket* prosedürünün *sözde* kodları verilip algoritma adımları detaylı bir şekilde anlatılacaktır. Bu aşamadan algoritmanın sağlıklı bir şekilde çalışması için çok önemli bir nokta vardır; en iyi amaç fonksiyon değerine sahip olan nokta x^{best} içinde bulunduğu iterasyon adımında hareket etmez ve bir sonraki iterasyon değerine aynı amaç fonksiyon değeri ile geçer (Satır 2). Bu durum bize hesaplama ve algoritmanın işlem zamanı açısından büyük avantaj sağlamaktadır. Çünkü bu sayede x^{best} üzerine etkiyen kuvvetleri hesaplamak gerekmemektedir. Bununla bitlikte, x^{best} haricindeki her bir parçacık kendi üzerine etkiyen bileşke kuvvet yönünde hareket etmektedir. Ama bir iterasyondaki x^{best} parçacığının algoritmanın en sonuna kadar sabit kalacağı anlamına gelmez. Çünkü bir sonraki iterasyonda başka bir parçacık daha iyi bir noktaya rastlarsa, oraya yerleşir ve yeni x^{best} noktası o parçacık olur.

ALGORITHM 5. Move(**F**)

1: **for** *i* = 1 to *m* **do**
2: **if** *i* ≠ best **then**
3: $\lambda \leftarrow U(0, 1)$
4: $F^i \leftarrow \frac{F^i}{\|F^i\|}$
5: **for** *k* = 1 to *n* **do**
6: **if** $F_k^i > 0$ **then**
7: $x_k^i \leftarrow x_k^i + \lambda F_i^k \ (u_k - x_k^i)$
8: **else**
9: $x_k^i \leftarrow x_k^i + \lambda F_i^k \ (x_k^i - l_k)$
10: **end if**
11: **end for**
12: **end if**
13: **end for**

3.1.5 Tamamlanma Kriteri

Elektromanyetizma sezgiseli için algoritmanın tamamlanması için maksimum iterasyon sayısı parametresinin (MAXITER) kullanılması önerilmiştir [10]. Maksimum iterasyon sayısının belirlenmesinde çeşitli ve farklı yöntemler kullanılabilir. Fakat yapılan testlerde *n* boyutlu bir problemin çözümünde *MAXITER*=25n şeklinde bir kullanımın ortalama karmaşıklığa sahip bir fonksiyon için optimal noktaya yakınsanma açısından son derece yeterli ve uygun olduğu tespit edilmiştir [10].

Bir diğer tamamlanma kriteri ise x^{best} parçacığının değişmediği başarılı iterasyon sayısı olarak da kullanılabilir. Örneğin bir fonksiyonun en iyi amaç fonksiyon değeri son 20 iterasyon boyunca değişmiyorsa burada algoritmayı tamamlayıcı kriter olarak 20 alınabilir. Fakat bu kararı alırken çok dikkatli davranılmalıdır. Çünkü işin içinde algoritmanın optimum noktayı bulmadan durdurulması riski olduğu gibi gereksiz iterasyon denemelerinden de kaçınmak gerekmektedir.

Literatürde bu gibi algoritmaların durdurulması için çeşitli yöntemlerin önerildiği görülebilir [111]. Sıklıkla kullanılan bir diğer durdurma kriteri ise optimize edilmek istenen fonksiyon değerinin, optimal değere ε hata payı kadar yaklaşımını baz almaktadır [112]. Fakat bu yaklaşım Elektromanyetizma sezgiseli için çok kullanılabilecek bir yöntem değildir çünkü global optimum noktası bilinmemektedir.

3.1.6 Algoritmanın Sürekli bir Fonksiyon Üzerinde Uygulanması

Elektromanyetizma sezgiseli yapısı itibari ile sürekli ve kısıtsız optimizasyon alanında kullanılmak üzere geliştirilmiş bir algoritmadır. Bu sezgiselin işleyişini ve çözüm üretme performansını adım adım göstermek için *Rastrigin* fonksiyonu kullanılmıştır. *Rastrigin* fonksiyonu global optimizasyon yöntemlerinde test amaçlı sıklıkla kullanılan ve üzerinde çalışılması nispeten zor denebilecek bir konfigürasyona sahiptir.

3.1.6.1 Rastrigin Fonksiyonu'nun Tanıtımı

Matematiksel optimizasyonda sıklıkla kullanılan Rastrigin Fonksiyonu konveks olmayan bir yapıya sahiptir. Doğrusal olmayan çok modlu fonksiyon yapısı vardır. İlk olarak Rastrigin [113] tarafından 1974 yılında ortaya atılmış ve Mühlenbein [114] tarafından genel halini almıştır.

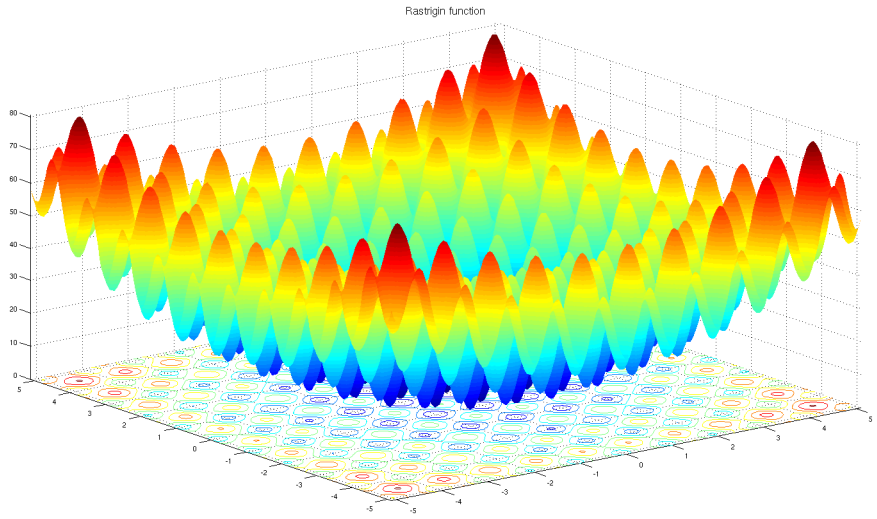
Büyük ölçekli bir yapıya sahip olması ve içinde çok fazla yerel minimum nokta bulunması sebebiyle bu fonksiyon tipinin global minimum noktasının bulunması oldukça zordur.

Bu sebeple Genetik Algoritmalar veya Tavlama Benzetimi gibi sezgisellerin de test edilmesi aşamasında sıklıkla başvurulur. Fonksiyonun ana şekli fonksiyon (3.4) olarak verilmiştir:

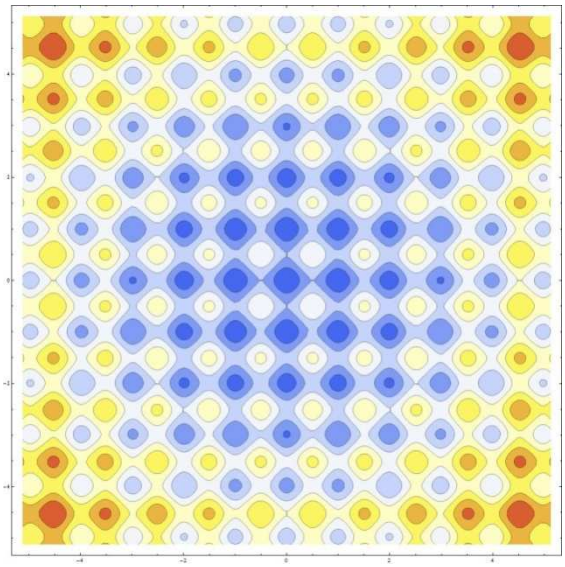
$$f(x) = An + \sum_{i=1}^{n} [\, x_i^2 - Acos(2\pi x_i)] \qquad (3.4)$$

$A = 10$ ve $x_i \in [-5.12, 5.12]$

Rastrigin fonksiyonunun global optimum noktası 0 noktasında bulunur ve $f\ (0) = 0$ 'dır.



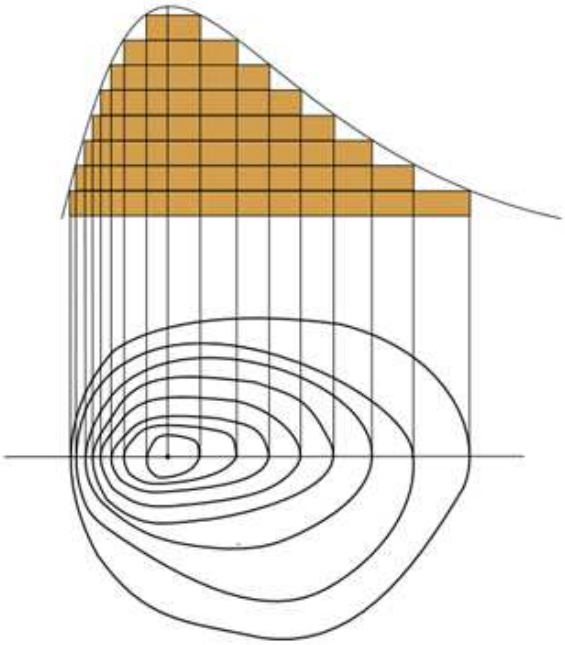
Şekil 3.5 Rastrigin Fonksiyonunun 3D görüntüsü [115]



Şekil 3.6 Rastrigin Fonksiyonunun Kontur Düzlemindeki görüntüsü [115]

3.1.6.2 Fonksiyon Konturları:

Fonksiyon konturları iki değişkenli fonksiyonların irtifa seviyelerinin bir düzlem üzerindeki iz düşüm eğrileridir. Kartografi 'de kontur çizgilerinin renkleri deniz seviyesi referans (0) alınarak şekillendirilir. Oluşturulan çizgilerin aralığı fonksiyonun eğimlerini gösterir. Oluşturulan kontur çizgileri üç boyuta sahip bir fonksiyonu iki boyutlu bir düzlem üzerinde gözlemleme ve inceleme fırsatı sunmaktadır.

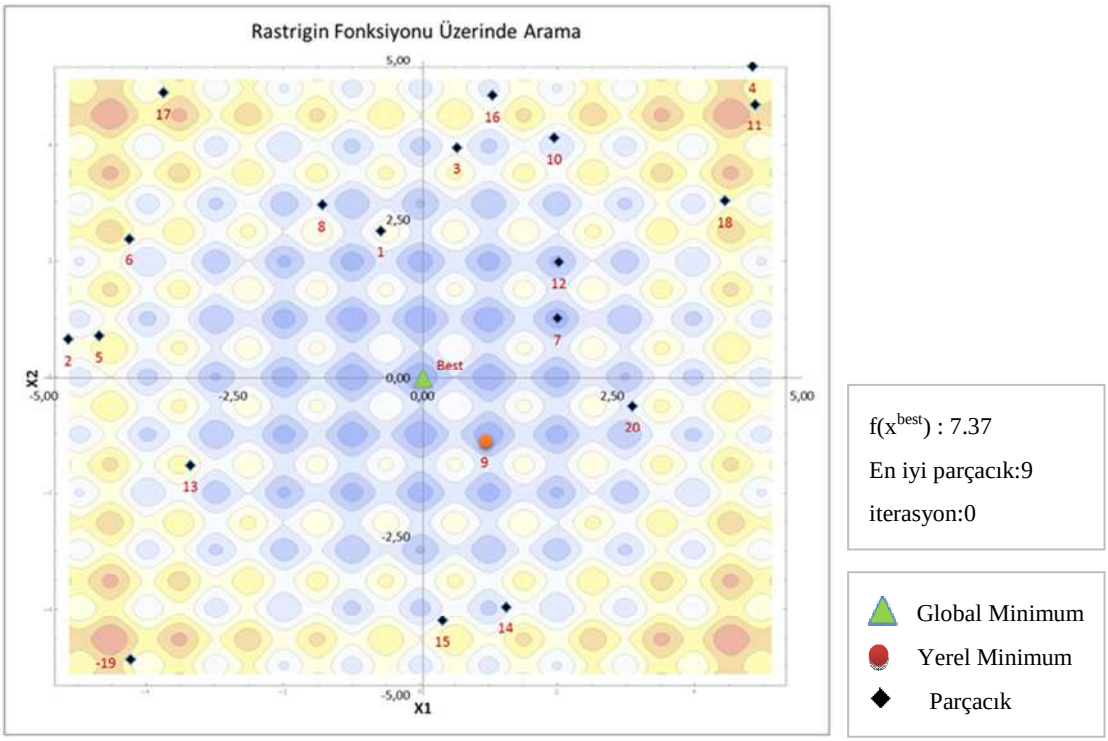


Şekil 3.7 Bir fonksiyonun Kontur Düzlemindeki görüntüsü.

Elektromanyetizma sezgisel algoritmasının çalışma prensibinin analizi ve tespiti açısından fonksiyon kontur düzlemi kavramının da açıklanması önemlidir. Zira Elektromanyetizma algoritmasının başlangıç safhasında rassal olarak üretilen *m* adet parçacığın hareket edecekleri fizibil alan işte bu kontur düzlemidir. Düzlem üzerindeki mavi alanlar amaç fonksiyon değeri görece düşük olan alanlara, kırmızı renkli alanlar ise amaç fonksiyonu görece daha yüksel kısımları temsil etmektedir.

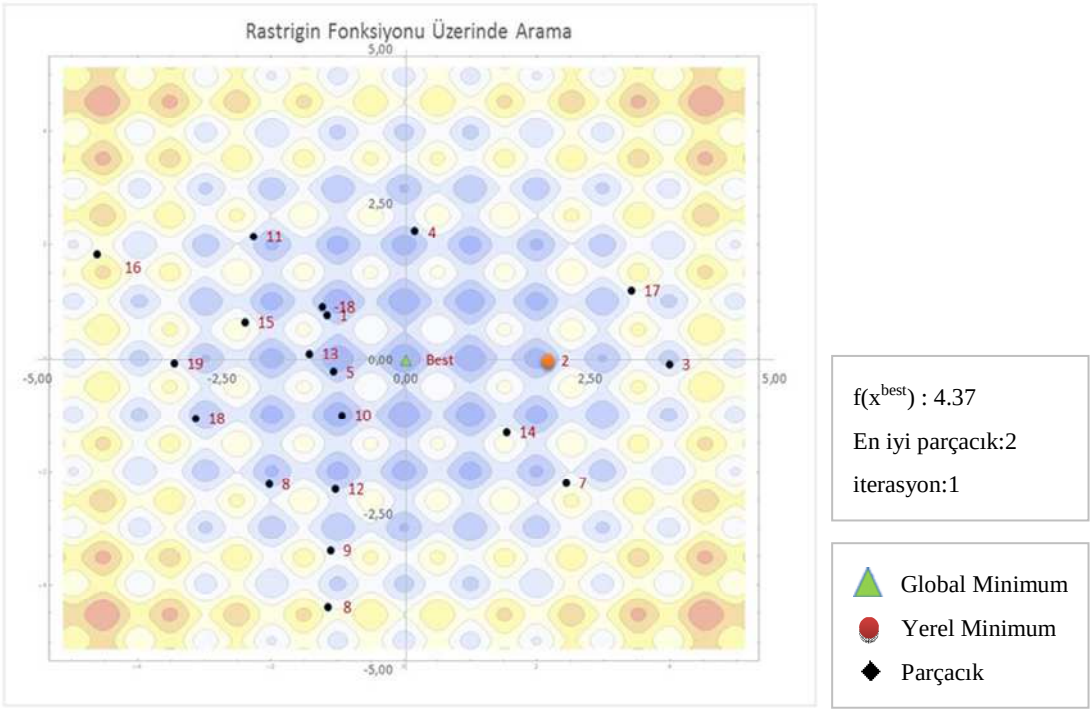
Gösterilecek örnekte ● şekli t. iterasyon için en iyi noktayı gösterir. ▲ ise global optimum noktasının konumudur. Algoritmanın işleyiş yöntemini göstermek adına 20 adet parçacık kullanılmıştır. Yani bu örnekte m=20'dir.

Algoritmanın işleyişi 20 adet rastgele parçacığın türetilmesi ile başlar. Bu durumda x^{best} noktası global optimum noktasının uzağındadır. Başlançık aşamasında Rastrigin fonksiyonunun kontur düzlemi üzerinde konumlandırılan parçacıklar henüz elektromanyetik ilşki içinde değildirler. Dolayısı ile amaç fonksiyon değerinin optimize edilmesi işlemlerine henüz başlanmamıştır.



Şekil 3.8 Rastrigin Fonskiyonu üzerinde EM algoritmasının çalışması (0. İterasyon)

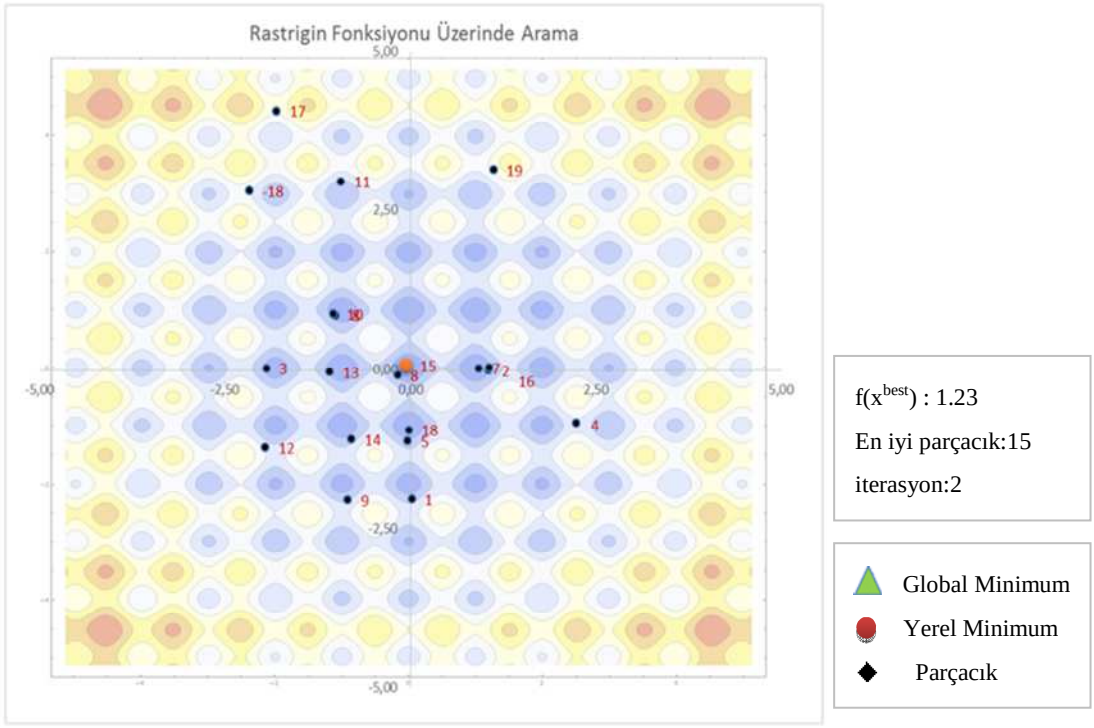
Şekil 3.8'de görüldüğü gibi 20 adet çözüm parçacığının fonksiyonun kontur düzlemi üzerinde rastgele oluşturulmuş ve oluşturulan parçacıklar düzlem üzerinde homojen bir şekilde dağılmıştır. Mevcut durumda 9. parçacık en iyi amaç fonksiyon değerine sahip olan parçacıktır ve amaç fonksiyon değeri 7.37'dir. Kırmızı noktalara denk gelen parçacıkların amaç fonksiyon değerleri son derece kötüdür. Mavi alanlara denk gelen parçacıkların ise nispeten daha iyi amaç fonksiyon değerlerine sahip oldukları açıktır. İtme-çekme mekanizması devreye girdikten sonra mavi alanlardaki parçacıklar kırmızı alanlardaki parçacıkları kendilerine doğru çekeceklerdir. Kırmızı alan üzerindeki parçacıklar ise kendilerini çeken iyi parçacıkları daha iyi çözüm alanları aramak amacıyla iteceklerdir. Fakat x^{best} noktası bu iterasyonda hareket etmeyecektir (Algoritma 5-Satır 2). Fakat hareket eden noktacıklar daha iyi alanlar bulurlarsa 9. Parçacık en iyi parçacık olma özelliğini kaybedecek ve diğer iterasyonlarda o da hareket edecektir.



Şekil 3.9 Rastrigin Fonskiyonu üzerinde EM algoritmasının çalışması (1. İterasyon)

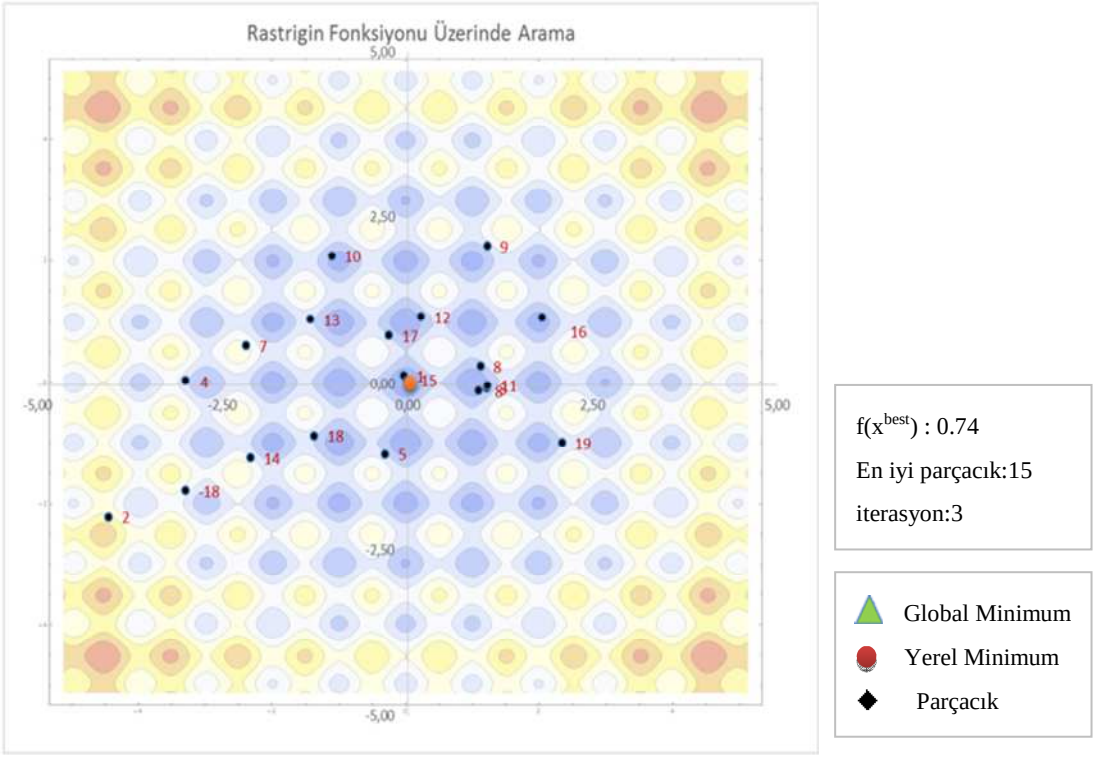
İtme-Çekme mekanizmasının devreye girmesiyle birlikte fizibil uzayda bulunan her parçacık üzerlerinde oluşan kuvvetlerin bileşkesi yönünde hareket eder ve hareket ettikleri doğrultu üzerinde daha iyi bir nokta olup olmadığını arar. Şekil 3.9'da 1. iterasyon sonundaki parçacık dağılımı görülmektedir. Buna göre, 1. iterasyon sonunda

2. parçacık en iyi parçacık olmuştur ve amaç fonksiyon değeri 4.37'dir. Bir önceki iterasyonda en iyi parçacığın amaç fonksiyon değeri 7.37'ydi. 1. iterasyon sonunda amaç fonksiyon değerinde 40,7% 'lik bir iyileşme gerçekleşmiştir.



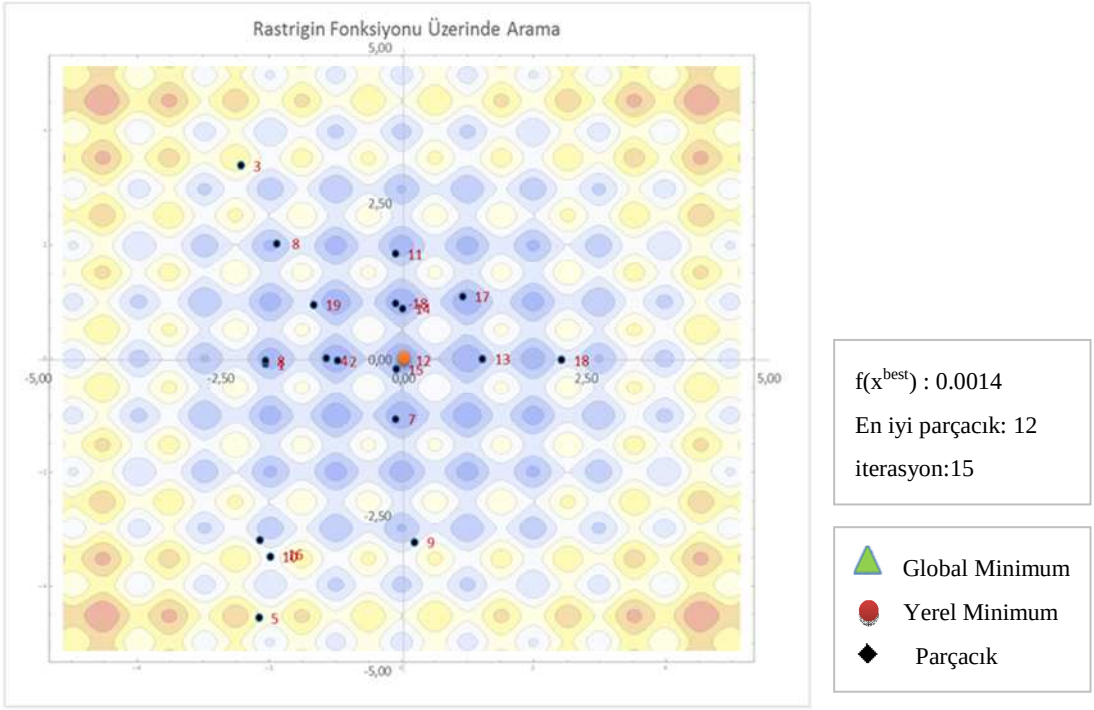
Şekil 3.10 Rastrigin Fonskiyonu üzerinde EM algoritmasının çalışması (2. İterasyon)

Şekil 3.10'da 2. iterasyon sonucunda çözüm parçacıklarının mavi renkli alanlar üzerinde yoğunlaştığı ve birbirlerine yaklaştıkları görülmektedir. Bir önceki iterasyonda en iyi parçacık 2. Parçacıkken 2. iterasyon sonucunda en iyi parçacık 15. parçacık olmuştur ve amaç fonksiyon değeri bir önceki iterasyonda 4.37 iken bu iterasyonda 1.23 olmuştur. 2. iterasyon sonunda 71,8'lik bir iyileşme gerçekleşmiştir.



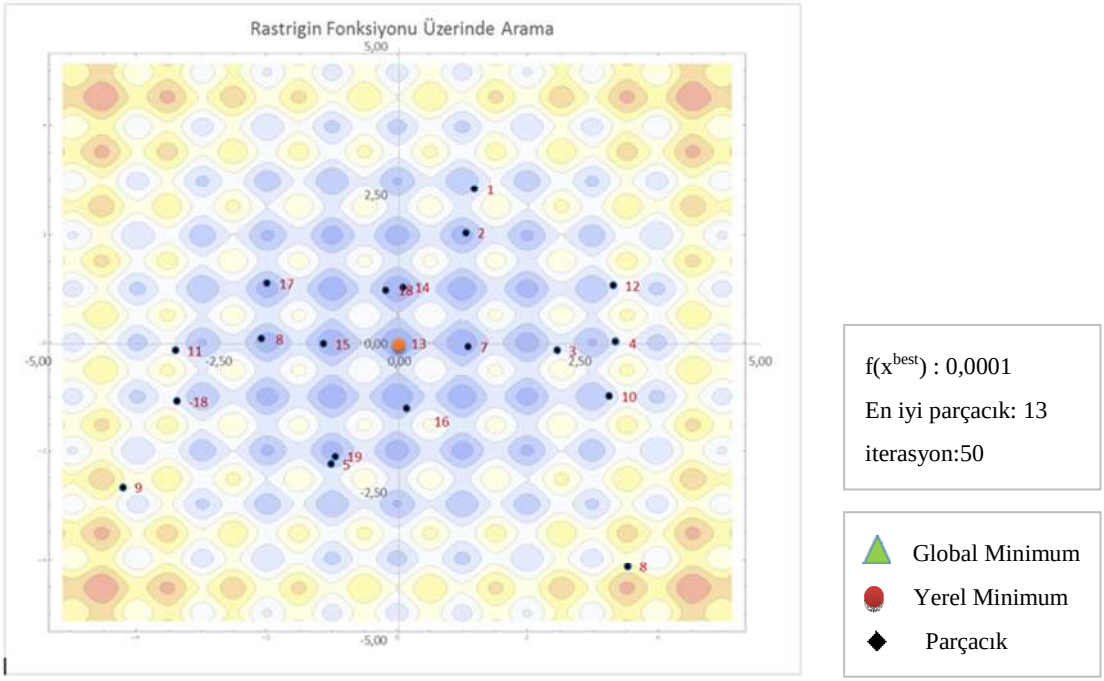
Şekil 3.11 Rastrigin Fonskiyonu üzerinde EM algoritmasının çalışması (3. İterasyon)

Şekil 3.11'de 2. iterasyonda sonundaki en iyi parçacık olan 15. parçacığın 3. İterasyon sonunda da değişmediği görülmektedir. Fakat sahip olmuş olduğu amaç fonksiyon değeri 1.23'ten 0.74'e düşmüştür. 3. İterasyonda, amaç fonksiyonunda 39.8%'lik bir iyileşme gerçekleşmiştir.



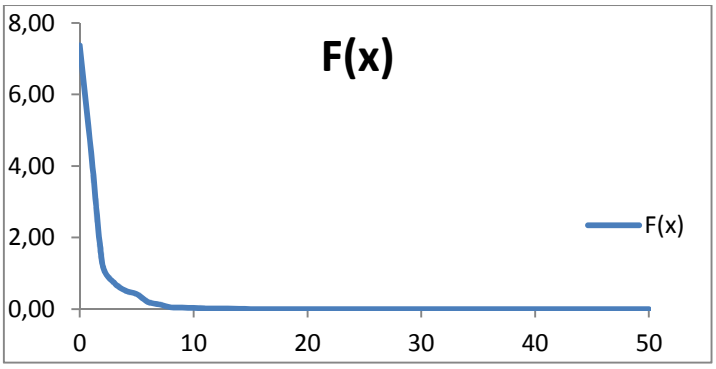
Şekil 3.12 Rastrigin Fonskiyonu üzerinde EM algoritmasının çalışması (15. İterasyon)

Şekil 3.12'de 15. iterasyon neticesinde en iyi amaç fonksiyon değerine sahip olan çözüm parçacığın 12. Parçacık olduğu ve sahip olduğu amaç fonksiyon değerinin 0.0014 olduğu görülmektedir. Yerel arama ve itme-çekme mekanizması sonucunda amaç fonksiyon değeri büyük ölçüde iyileşmiş ve global minimum noktasına yaklaşmıştır. Toplam iterasyon sayısı olan MAXITER parametresi 50 seçilmiştir. Bu da bundan sonraki iterasyon adımlarında daha küçük miktarlı iyileştirmelerin olacağının bir göstergesidir.



Şekil 3.13 Rastrigin Fonskiyonu üzerinde EM algoritmasının çalışması (50. İterasyon)

EM algoritmasının Rastrigin Fonksiyonu üzerindeki çalışma mekanizmasını göstermek adına 1.-2.-3.-15. ve 50. iterasyon neticesinde parçacıkların dağılımı ve x^{best} değerleri gözlemlenmiştir. Şekil 3.13'te görüldüğü üzere 50. İterasyon neticesinde 0.00 olan glabal minimum noktasına 0.0001 hata payı ile yakınsanarak optimal çözüm değeri elde edilmiştir. Bu açıdan değerlendirildiğinde algoritmanın optimal çözüme yaklaşma becerisi yüksektir. Şekil 3.13'te son olarak f(x) değerlerinin algoritmanın çalışması boyunca nasıl değiştiği gösterilmiştir.



Şekil 3.14 F(x) değerlerinin iterasyon bazlı değişimi

3.2 Rassal Anahtar Yöntemi

Elektromanyetizma sezgisel algoritması kısıtsız ve sürekli fonksiyonların optimizasyonu üzerinde kullanılabilen bir yöntemdir. Bu algoritmayı kesikli ve kısıt içeren problem tiplerine uygulayabilmek için bir takım eklentiler yapmak gerekmektedir. Bu amaçla, sıralama problemlerini genetik algoritma ile çözmek adına Bean [116] tarafından geliştirilmiş olan rassal anahtar tekniğinin EM algoritması ile birleştirilerek başarı ile kullanılabileceği tespit edilmiştir. Rassal anahtar tekniği basit ve kolay uygulanabilir bir yapıya sahiptir. Bir çözümün oluşması için k boyutlu bir dizinin oluşturulması gerekiyor olduğunu farz edilsin, bu bağlamda:

1. k boyutlu dizinin her bir hücresi için birer rassal sayı üretilir ve hücre içine atanır.
2. rassal sayılar küçükten büyüğe veya büyükten küçüğe doğru sıralanır.
3. Rassal sayılar sıralanırken, karşılık geldikleri hücre indisleri de sıralanır.

Rassal anahtar tekniği kullanılırken herhangi bir sıralama algoritması kullanılabilir. Zaman karmaşası ve işlem hızı açısından hızlı sıralama (quick sort) algoritmasının kullanılması çok uygun olmaktadır. Sıralama yapıldıktan sonra oluşan sıranın amaç fonksiyon değeri hesaplanır.

Çizelge 3.1'de 10 boyutlu bir dizi gösterilmektedir.

Çizelge 3.1 Rassal Anahtar Yönteminin uygulanışı

önce

1	2	3	4	5	6	7	8	9	10
0,71	0,23	0,83	0,34	0,52	0,04	0,10	0,98	0,64	0,17

sonra

6	7	10	2	4	5	9	1	3	8
0,04	0,10	0,17	0,23	0,34	0,52	0,64	0,71	0,83	0,98

Rassal sayı üreten mekanizma bu dizinin ilk üç hücresi için 0.71, 0.23 ve 0,83 sayılarını üretmiştir. Rassal Anahtar algoritması için bu örnekte küçükten büyüğe sıralama yöntemi seçildiği uygulanacaktır. Dolayısı ile 0.71 bu dizideki en küçük 8. rassal sayı olduğu için yeni sıralamada 8. sıraya atanacaktır ve ilk sıradaki indis değeri olan 1 de aynı sıraya yerleşecektir. Üretilen ikinci rassal sayı olan 0.23 en küçük 4. rassal sayıdır. Dolayısı ile yeni dizide 4. Sıraya yerleşir ve ilk dizideki indisi olan 2 de kendisini takip ederek aynı sıraya yerleşir. Üretilen üçüncü rassal sayı olan 0.83 en küçük 9. rassal sayıdır. Dolayısı ile yeni dizide 9. Sıraya yerleşir ve ilk dizideki indisi olan 3 de kendisini takip ederek aynı sıraya yerleşir. Böylelikle yeni sıralama olan {6,7,10,2,4,5,9,1,3,8} oluşur.

3.3 Rassal Anahtar Yönteminin EM Algoritmasında Uygulanması

EM algoritmasının kesikli ve kısıtlı problemleri üzerinde uygulanabilmesi için rassal anahtar yöntemi etkin bir şekilde kullanılabilmektedir. Her nasıl ki Japonya standartları için tasarlanmış bir fişin Türkiye'de kullanılabilmesi için bir ara soket kullanmak gerekiyorsa, global optimizasyon problemleri için tasarlanmış bir algoritmanın da yöneylem problemleri üzerinde kullanılabilmesi için de bir uyarlayıcı yöntemin kullanılması gerekli olacaktır.

Global optimizasyon alanında, sürekli fonksiyonlar üzerindeki optimum noktayı bulmak üzere tasarlanan Elektromanyetizma sezgiselinin sahip olduğu fonksiyonel mekanizmanın farklı bir alan olan optimizasyon problemlerine uygulanabilmesi için 3. Bölümde detayları ile anlatılan EM algoritmasının sahip olduğu mekanizmaların kesikli problemlerde uyarlanma adımları 4. bölümde örneklerle açıklanacaktır.

BÖLÜM 4

GEZGİN SATICI PROBLEMLERİNE UYGULANMASI

EM algoritmasının kesikli ve kısıtlı problem tipleri üzerindeki etkisini araştırmak üzere bir süt toplama ağı kurgulanmıştır. Kurgulanan ağ bir adet depo ve depodan çıkan tankerin sırayla uğradığı çiftliklerden oluşmaktadır. Algoritmanın bulduğu çözümün kalitesi ve çözüm sürelerini test etmek amacıyla her denemede çiftlik sayısı her defasında arttırılmış ve algoritmanın parametreleri üzerinde değişiklikler yaparak algoritmanın çalışma performansı detaylı olarak test edilmiştir. EM algoritmasının vermiş olduğu değerin doğruluğu GAMS (2013) programının vermiş olduğu sonuçlar baz alınarak değerlendirilmiştir. Fakat algoritmanın çalışma süresi performansı ise TB algoritması ile kıyaslanmıştır. Çünkü GAMS (2013) programının çözüm süresi sezgisel algoritmalara nazaran çok daha kısadır fakat GSP'nin başlıca problemi olan alt tur kısıtının yazılmasındaki zorluk doğrusal modelleme kullanılmasını ergonomik kılmamaktadır.

5.1 Süt Toplama Problemi

1. Rota depodan başlayıp depoda sona erecektir
2. N adet düğüm noktasına uğranacaktır
3. Toplama aracının kapasitesi her durumda yeterlidir
4. Her bir düğüme tek bir düğümden gidilebilir
5. Her bir düğümden ancak bir düğüme gidilebilir
6. Rota tamamlandığında kapalı tek bir döngü oluşmaktadır.

$$\text{Min} \qquad \sum_i \sum_j x_{ij} * d_{ij} \tag{4.1}$$

$Kısıtlar$:

$$\sum_j x_{ij} = 1 \qquad \sum_j x_{ij} = 1 \tag{4.2}$$

$$\sum_i x_{ij} = 1 \qquad \sum_i x_{ij} = 1 \tag{4.3}$$

$$\sum_{i \in S} \sum_{j \notin S} x_{ij} \geq 1 \qquad \forall S \tag{4.4}$$

$$xij \in \{0,1\} \tag{4.6}$$

Karar Değişkeni:

$$x_{ij} = \begin{cases} 1 & Araç\ i\ düğümünden\ j\ düğümüne\ gidiyorsa \\ 0 & değilse \end{cases} \qquad i, j \in n\ , i \neq j$$

$dij =\ i.ve\ j.$düğüm $arasındaki\ mesafe$

Bilindiği gibi GSP'ler alt tur engelleme kısıtlarına dayalı atama tabanlı doğrusal programlama modellerini kullanırlar. Bu doğrultuda bu çalışmada öncelikle GSP problem tipi belirlenmiş ve doğrusal modeli oluşturularak GAMS yazılımı ile kodlanmıştır. Gezgin satıcı probleminin matematiksel modeli aşağıda verilmiştir.

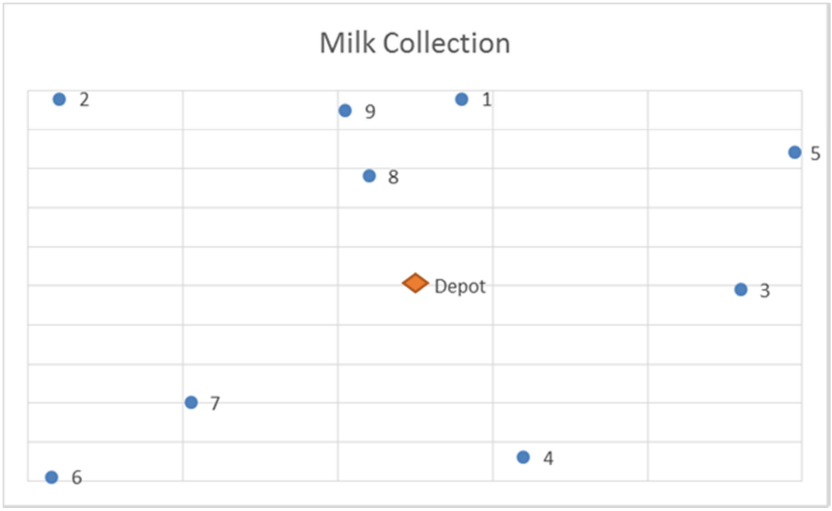
Oluşturulan matematiksel modelde amaç fonksiyonu (4.1) gidilen düğümler arasındaki mesafelerin toplamını minimize etmektedir. (4.2) ve (4.3) iki düğüm arasında yalnızca tek rota olmasını sağlar. (4.4) kısıtı alt-tur oluşumunu bertaraf eden kısıtlardır.

5.2 GAMS Uygulaması

Bir gezgin satıcı probleminde orijin noktasında (0,0) bulunan depodan çıkan bir süt toplama kamyonunun süt işleme tesisinin çevresindeki 9 ilçede bulunan süt üretim çiftliklerini dolaşması ve bu çiftliklerden süt toplaması gerektiği ele alınsın. 9 adet çiftliği dolşarak süt toplayan kamyon en sonunda süt toplama tesisine geri dönerek topladığı sütü boşaltmaktadır ve tüm çiftliklerde üretilen bütün sütleri taşıyacak büyüklükte kapasitesinin var olduğu kabul edilmektedir. Yani bu problem tipinde herhangi bir kapasite kısıtı yoktur.

Gidilmesi gereken tüm noktaları birbirlerine bağlayan noktaların özdeş özellikte bulunduğu durumlarda amaç fonksiyonu olarak toplam mesafenin minimizasyonu kullanılabilir. Zira toplam mesafeyi minimize etmekle hem nakliye hem de zaman maliyetleri eş zamanlı olarak minimize edilmiş olacaktır.

Depo ve süt üretim çiftliklerinin konumları Şekil 4.1'de gösterilmiştir.



Şekil 4.1 Depo ve Çiftlikler Konumları

Depo ile 9 adet toplama noktasının aralarındaki simetrik mesafe matrisi Çizelge 4.1'de gösterilmiştir. 0 no'lu indis deponun indisidir.

Çizelge 4.1 Depo ve toplama bölgeleri arasındaki mesafeler matrisi

msf(i,j)	0	1	2	3	4	5	6	7	8	9
0	0,00	48,37	66,48	42,01	46,17	59,64	67,90	41,73	28,64	45,89
1	48,37	0,00	52,00	60,80	92,35	45,22	110,54	85,49	23,32	15,30
2	66,48	52,00	0,00	100,72	109,84	96,03	97,01	79,83	44,72	37,12
3	42,01	60,80	100,72	0,00	51,31	35,69	101,12	76,69	56,08	68,68
4	46,17	92,35	109,84	51,31	0,00	85,49	61,20	45,22	74,73	91,92
5	59,64	45,22	96,03	35,69	85,49	0,00	126,91	100,90	55,33	59,03
6	67,90	110,54	97,01	101,12	61,20	126,91	0,00	26,17	87,24	101,39
7	41,73	85,49	79,83	76,69	45,22	100,90	26,17	0,00	62,39	77,62
8	28,64	23,32	44,72	56,08	74,73	55,33	87,24	62,39	0,00	17,26
9	45,89	15,30	37,12	68,68	91,92	59,03	101,39	77,62	17,26	0,00

EM algoritmasının performansının karşılaştırılması amacıyla GAMS (2013) yazılımı üzerinde GSP modellenmiştir (Şekil4.2).

```
variables

x(i,j)
z;

binary variables x(i,j);

equations

amac
k1(i)
k2(j)
;

amac .. z=e=sum((i,j),m(i,j)*x(i,j));
k1(i).. sum((j),x(i,j))=e=1;
k2(j).. sum((i),x(i,j))=e=1;

model GSP /all/;

solve GSP using MIP minimizing z;
```

Şekil 4.2 GSP'nin GAMS üzerinde modellenmesi (Alttur Kısıtsız)

Problemin modellenmesi aşamasında alttur oluşmasını engelleyici kısıtlar eklenmemiştir. Bu sebeple problemin çözümü neticesinde bir çok alt tur ile karşılaşılmıştır (Şekil 4.3).

Şekil 4.3 Alttur Probleminin Grafiksel Gösterimi

Alttur probleminin önüne geçmek için alt turu engelleyici kısıtlar eklenmiştir. Fakat problemin hacminden dolayı alt tur kısıtları da çok büyük olmuştur ve yazılması epey zaman almıştır(Şekil4.4).

```
variables
x(i,j)
z;
binary variables x(i,j);
equations
amac
k1(i)
k2(j)
ks1
ks2
ks3
ks4
ks5
ks6
;
amac .. z=e=sum((i,j),m(i,j)*x(i,j));
k1(i).. sum((j),x(i,j))=e=1;
k2(j).. sum((i),x(i,j))=e=1;
ks1 .. x('0','1')+x('0','2')+x('0','3')+x('0','5')+x('0','6')+x('0','7')+x('0','8')+x('0','9')+
       x('4','1')+x('4','2')+x('4','3')+x('4','5')+x('0','6')+x('0','7')+x('1','8')+x('1','9')=g=1;
ks2 .. x('1','0')+x('1','2')+x('1','3')+x('1','4')+x('1','5')+x('1','6')+x('1','7')+x('1','8')+
       x('9','0')+x('9','2')+x('9','3')+x('9','4')+x('9','5')+x('9','6')+x('9','7')+x('9','8')=g=1;
ks3 .. x('2','0')+x('2','1')+x('2','3')+x('2','4')+x('2','5')+x('2','6')+x('2','7')+x('2','9')+
       x('8','0')+x('8','1')+x('8','3')+x('8','4')+x('8','5')+x('8','6')+x('8','7')+x('8','9')=g=1;
ks4 .. x('3','0')+x('3','1')+x('3','2')+x('3','4')+x('3','6')+x('3','7')+x('3','8')+x('3','9')+
       x('5','0')+x('5','1')+x('5','2')+x('5','4')+x('5','6')+x('5','7')+x('5','8')+x('5','9')=g=1;
ks5 .. x('6','0')+x('6','1')+x('6','2')+x('6','3')+x('6','4')+x('6','5')+x('6','8')+x('6','9')+
       x('7','0')+x('7','1')+x('7','2')+x('7','3')+x('7','4')+x('7','5')+x('7','8')+x('7','9')=g=1;
ks6.. x('4','0')+x('6','0')+x('7','0')+x('4','1')+x('6','1')+x('7','1')+x('4','2')+x('6','2')+
       x('7','2')+x('4','3')+x('6','3')+x('7','3')+x('4','5')+x('6','5')+x('7','5')+x('4','8')+
       x('6','8')+x('7','8') +x('4','9')+x('6','9')+x('7','9') =g=1
```

Şekil 4.4 GSP'nin GAMS üzerinde modellenmesi (Alttur Kısıtlı)

5.3 EM Uygulaması

Bu çalışmada EM sezgisel algoritması rassal anahtar yöntemi ile birleştirilerek GSP problemleri üzerindeki çözüm performansı incelenmiştir.

GSP'lerin kesin çözüm yöntemleri ile çözümünü zor kılan en büyük engel alt tur probleinin önüne geçmektir. Küçük hacimi problemlerde alt tur probleminin önüne geçmek için eklenen alt tur engelleme kısıtlar nispeten daha basitken problemin hacmi büyüdükçe oluşan alt tur miktarları arttıkça bu kısıtı kontrol etmek gittikçe güçleşmektedir.

5.3.1 EM Algoritmasının Süt Toplama Problemi Üzerinde Manuel Çalıştırılması

Algoritmanın çalışma aşamaları ve prensiplerinin gösterilebilmesi için 3 adet çözüm parçacığı seçilmiştir. Her bir parçacığın bünyesinde depo'dan (0) başlayıp gene depo'da biten 11 düğüm noktası vardır. Dolayısı ile uygun çözümler arada bulunan 9 çiftlik düğümünün permütasyonları alınarak oluşur. Amaç fonksiyonunun nasıl oluştuğunu göstermek için oluşturulan 3 adet rassal parçacık Çizelge 4.2 'te gösterilmiştir.

Çizelge 4.2 Çözüm Parçacıklarının içerdiği dizi permütasyonları ve amaç değerleri

Çözüm 1	0	3	6	9	5	2	7	1	4	8	0
f(x)=760.61	42,01	101,12	101,39	59,03	96,03	79,83	85,49	92,35	74,73	28,64	
Çözüm 2	0	8	6	2	9	7	1	4	5	3	0
f(x)= 668.65	28.64	87.24	97.01	37.12	77.62	85.49	92.35	85.49	35.69	42.01	
Çözüm 3	0	9	1	2	6	5	4	8	7	3	0
f(x)=678.41	45.89	15.30	52.00	97.01	126.91	85.49	74.73	62.39	76.69	42.01	

Belirlenen 3 adet parçacığın sahip olduğu dizilim permütasyonları ve bu permütasyonların oluşturulmasına yarayan rassal anahtarlar Çizelge 4.3'de gösterilmiştir.

Görüldüğü üzere rassal anahtarlar oluşturulduktan sonra sıralama algoritması uygulanmış, rassal anahtarlar küçükten büyüğe sıralanmış ve her bir rassal anahtarın denk geldikleri çiftlik indisleri de rassal sayının büyüklük sırasına göre konumlanmıştır.

Çizelge 4.3 Parçacıkların sahip oldukları dizi permütasyonlarının gösterimi

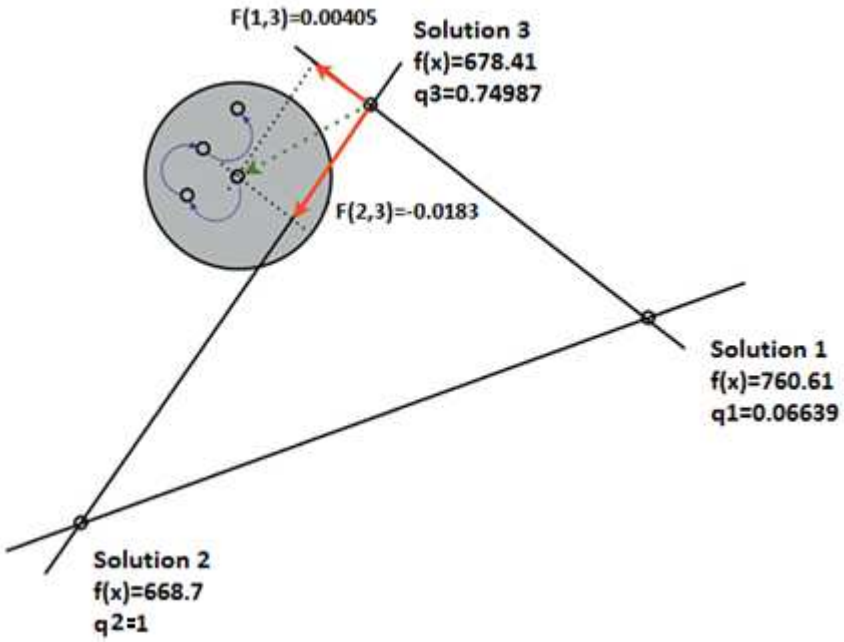
Konum	Çözüm 1		Çözüm 2		Çözüm 3	
	PM_1	RK_1	PM_2	RK_2	PM_3	RK_3
1	3	0,7238	8	0,4808	9	0,2280
2	6	0,3209	6	0,3564	1	0,3518
3	9	0,0211	2	0,7523	2	0,8097
4	5	0,7851	9	0,5438	6	0,6144
5	2	0,2650	7	0,6532	5	0,5095
6	7	0,1028	1	0,3543	4	0,4233
7	1	0,4044	4	0,4484	8	0,7193
8	4	0,8778	5	0,1282	7	0,6830
9	8	0,2475	3	0,4243	3	0,1352

Başlangıç çözümleri oluşturulduktan sonra, 2. parçacığın en iyi amaç fonksiyon değerine sahip olduğu görülmektedir f(2)= 668.65.

Daha önce de belirtildiği gibi EM metodolijisi, Coulomb Kanunu'nu baz alarak elektrostatik güçler mekanizmasındaki itme-çekme mekanizmasını kullanmaktadır. Buradaki en önemli nokta şudur, popülasyon içinde bulunan parçacıkların yükleri sabit değildir. Aksine, parçacık yükleri, en iyi amaç fonksiyonuna sahip parçacığa göre olan konumlarına göre göreli olarak değişkenlik gösterir. EM algoritmasının çalışması esnasında, iterasyonlar içindeki en iyi parçacık ve konumu değişkenlik gösterebileceği için, tüm parçacıkların yük değerleri de aynı şekilde değişkenlik gösterebilir. Bir parçacığın sahip olduğu yük değeri, popülasyon içindeki diğer parçacıklara uygulayacağı itme veya çekme miktarının büyüklüğünü belirler. Daha önce de belirtildiği gibi, popülasyon içindeki iki parçacık ele alındığında, amaç fonksiyon değeri daha iyi olan parçacığın, daha kötü olan parçacığı çekecektir. Amaç fonksiyon değeri daha kötü olan ise, iyi parçacığı itecektir. Fakat EM mekanizmasındaki en önemli noktalardan biri de en iyi parçacığın kendisinden daha iyi bir parçacık oluşana kadar yerinde sabit kalacağıdır. Bilindiği gibi yük değerleri (3.1) no'lu yük hesaplama denklemi ile hesaplanır.

Gösterdiğimiz örnekte, EM algoritmasının sıra permütasyonlarını içeren noktacıkların konumlanması, sahip oldukları yük değerleri ve yük değerlerinin neticesinde oluşan ve Denklem (3.2) ile hesaplanan itme ve çekme kuvvetlerinin büyüklükleri ve yönleri Şekil 4.5'de gösterilmiştir.

Görüldüğü üzere denklem (3.1)'e göre hesaplanan yük değerleri q^1=0.06639, q^2=1 ve q^3=0.074987'tür.



Şekil 4.5 İtme-Çekme Mekanizmasının iki boyutlu olarak grafiksel gösterimi

Rassal Anahtar yöntemi ile elektromanyetizma sezgisel algoritmasının birleştirilmesi amacıyla gösterilen örnekte öncelikle 3 adet parçacığın yükleri ve hesaplanmıştır. Daha sonra yük hesaplama ve uygulama prosedürünün uygulanışını açıklamak adına 3 no'lu parçacık üzerine etkiyen kuvvetler grafiksel olarak gösterilmiştir. Parçacıklar arasındaki mesafeler hesaplanırken öklidiyen uzaklık prensibi kullanılmıştır. x^k ve x^i arasındaki mesafe aşağıdaki denklemle hesaplanır:

$$d(x^i, x^k) = \sqrt{\left(x_1^k - x_1^i\right)^2 + \left(x_2^k - x_2^i\right)^2 + \cdots + \left(x_n^k - x_n^i\right)^2} = \sqrt{\sum_{j=1}^{n}\left(x_j^k - x_j^i\right)^2} \quad (4.6)$$

Bu denklem ile parçacıkların arasındaki mesafeler hesaplanırken klasik elektromanyetizma sezgisel algoritmasından farklı olarak her bir parçacık içindeki permütasyonları oluşturan rassal anahtarlar arasındaki mesafeler hesaplanır. Yani burada her bir rassal anahtar değeri bir koordinat görevi görür. Örnekte üzerine etkiyen kuvvetlerin hesaplandığı 3. Parçacık için de 1. ve 2. parçacıkların arasındaki mesafeler aşağıdaki çizelgede hesaplanmış ve Çizelge 4.4 'te gösterilmiştir.

Çizelge 4.4 Mesafeler

Konum		
1	0,0590	0,2458
2	0,0013	0,0010
3	0,5347	0,6219
4	0,0582	0,0291
5	0,1507	0,0598
6	0,0633	0,1027
7	0,0019	0,0992
8	0,5619	0,0379
9	0,0313	0,0126
Toplam	**1,4622**	**1,2100**
Mesafe	**1,2092**	**1,1000**

Örnekte, 3. Partikül üzerine etkiyen kuvvetler hesaplanmıştı. Bu amaçla 1 ve 3 ile 2 ve 3 nolu parçacıkların arasındaki mesafeler gösterilmişti. d(x2,x3)= 1.2092 ve de d(x1,x3)= 1.1000 olduğu çizelge z'de görülmektedir. 3. Parçacık üzerine etkiyen kuvvetlerin nasıl hesaplandığı Çizelge 4.5'da görülmektedir.

Çizelge 4.5 Kuvvet hesaplamaları

Konum	RK1	RK1	RK1	$F^{3,1}$	$F^{3,2}$	F^3	$F^3/\|F^3\|$	Yeni RK	RK'in Sıra
1	0,724	0,481	0,228	0,014	-2,24E-02	-0,009	-0,783	0,166	1
2	0,321	0,356	0,352	0,000	-2,08E-04	0,000	0,004	0,353	9
3	0,021	0,752	0,81	-0,003	2,60E-03	-0,001	-0,051	0,795	2
4	0,785	0,544	0,614	-0,004	3,20E-03	-0,001	-0,062	0,601	6
5	0,265	0,653	0,51	0,008	-6,50E-03	0,001	0,127	0,531	5
6	0,103	0,354	0,423	-0,004	3,12E-03	-0,001	-0,061	0,414	8
7	0,404	0,448	0,719	-0,015	1,23E-02	-0,003	-0,239	0,659	4
8	0,878	0,128	0,683	-0,030	2,51E-02	-0,005	-0,489	0,566	7
9	0,248	0,424	0,135	0,016	-1,31E-02	0,003	0,255	0,212	3
				-0,018	**4,05E-03**				

Garfiksel olarka Şekil 4.4'te görülen kuvvetlerin nasıl hesaplandıkları Çizelge 4.5'te detaylı bir şekilde anlatılmıştır. $F^{3,1}$ ve $F^{3,2}$ değerleri hesaplandıktan sonra aynı konuma ait kuvvetler skalar olarak toplanarak F^3 hesaplanır.

Tüm kuvvetler hesaplandıktan sonra sıra hesaplanan kuvvetler doğrultusunda parçacıkları hareket ettirmeye gelir. Yalnız burada dikkat edilmesi gereken nokta, en iyi noktanın kesinlikle hareket etmemesi gerektiğidir. Zira en iyi nokta da hareket ettirilirse mevcut iyi değerin bir sonraki iterasyonda kaybedilmesi riski doğar. Kuvvetler hesaplandıktan sonra kuvvetlerin doğrultusunda rassal adım uzunluğu λ kadar parçacıklar hareket eder. Bu örnekte rassal adım uzunluğu 0.35 olarak seçilmiştir. Burada dikkat edilmesi çok önemli bir nokta vardır. Elektromanyetizma sezgiselinin çalışma aşama ve prensplerinin anlatıldığı 3. Bölüm'de de detaylı olarak açıklandığı gibi fizibilite şartlarının korunabilmesi için çözüm parçacıkları yalnızca fizibil uzay sınırları içinde hareket edebilmektedir. Fizibil uzayın sınırlarını ise fonksiyon değişkenlerinin alt ve üst sınırları $(l_i \leq x_i \leq u_i)$ teşkil etmekteydi. Burada ise parçacığın hareket etmesi demek rassal anahtar değerlerinin değişmesi anlamına geldiği için her bir Rassal anahtarın alt ve üst sınırı [0,1]'dir. Yani denklem (3.3)'de belirtilen xij $\in$ [0,1]

$$x_j^i = \begin{matrix} x_j^i + \lambda \left| F_j^i \right| (u_j - x_j^i) & if\ F_j^i > 0 \\ x_j^i + \lambda \left| F_j^i \right| (x_j^i - l_i) & if\ F_j^i < 0 \end{matrix} \qquad (4.7)$$

Denklem (4.7) kullanılarak hesaplanan yeni rassal anahtarlar küçükten büyüğe sıralanarak yeni sıralama permütasyonu Çizelge 4.6'daki gibi oluşturulur.

Çizelge 4.6 Rassal Anahtar dizilimi ile yeni dizi permütasyonunun oluşturulması

RK	0,166	0,353	0,795	0,601	0,531	0,414	0,659	0,566	0,212	
Sıra	1	2	3	4	5	6	7	8	9	
Çiftlik	**1**	**9**	**2**	**6**	**5**	**8**	**4**	**7**	**3**	
	48,37	15,3	37,12	97,01	126,91	55,33	74,73	45,22	76,69	42,01	**618,69**

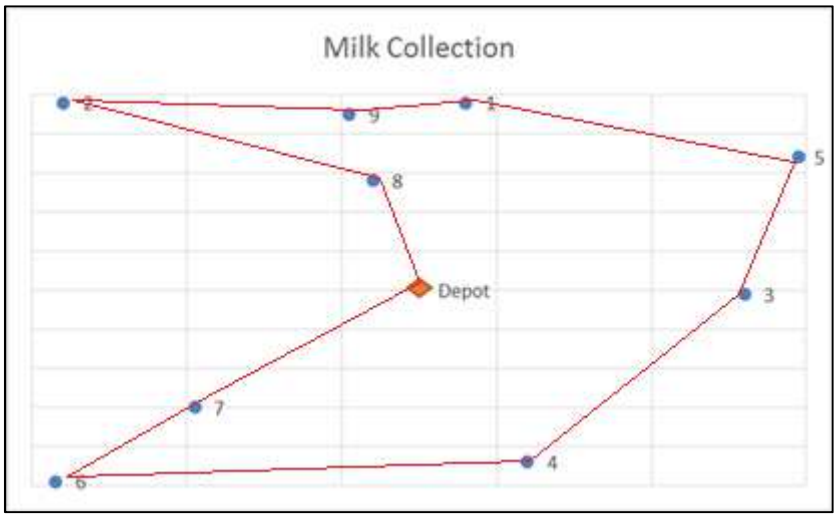
Elektromanyetizma algoritmasi ile rassal anahtar yöntemiin birleştirilmesi ile oluşturulan yeni yöntemle gösterilen örnekte, yeni rassal anahtarlar hesaplanmış, Çizelge 4.7 deki sıralama elde edilmiştir.

Bu sıralamanın amaç fonksiyon değeri ise 618.59'dUr. Görüldüğü üzere ilk permütasyonun amaç fonksiyon değeri 678.41'di. Dolayısıyla bu iterasyonda bir amaç 3. Parçacığın amaç fonksiyonu iyileşmiştir.

Çizelge 4.7 İterasyon sonunda oluşturulan ikinci parçacığa ait rota dizilimi

0	1	9	2	6	5	8	4	7	3	0

EM algoritmasının çalışmasını tamamladığında veriş olduğu optimal çözümün grafiksel gösterimi Şekil 4.6 'te gösterilmiş ve optimum amaç fonksiyon değeri 387.10 olarak hesaplanmıştır.



Şekil 4.6 GSP'nin optimal çözümü

5.3.2 Elektomanyetizma Algoritması Parametre Seçimi

Meta-sezgisel algoritmaların performansları yapılarında bulundurdukları parametrelerle doğrudan ilişkilidir. EM algoritması için en önemli üç parametre; Partikül sayısı, maksimum iterasyon sayısı (MAXITER) ve yerel arama sayısıdır (LSITER). Bu üç parametrenin gereğinden fazla seçilmiş olması algoritmanın tamamlanma süresini ciddi ölçüde arttıracaktır. Bu sebeple deneysel olarak da olsa farklı problem büyüklükleri için uygun parametre büyüklükleri ve kombinasyonları belirlenmiştir. Bu parametre kombinasyonları Çizelge 4.8'da detaylı olarak verilmiştir.

Partikül sayısının küçük tutulması fizibil uzay üzerindeki parçacıkların birkaç iterasyon sonrasında birbirlerine iyice yaklaşarak yapışmalarına dolayısı ile global optimuma ulaşmadan yerel minimuma takılı kalarak algoritmanın sonlanmasına sebebiyet verebilir. Bu sebeple daha esnek ve hareketli bir arama yapısını sağlayabilmek adına parçacık sayısını uygun bir sayıda seçmek gerekmektedir. Yapılan testlerde n<10 için m'yi 3 almak $10 < n < 20$ arasında m'yi 4 almak uygun olmaktadır. EM algoritması için maxiter sayısının n*25 olması önerilmiştir [11]. Yapılan testlerde işlem süresini biraz uzun tutsa da GSP problemleri için de MAXITER için n*15 büyüklüğü uygun görüşmüştür.

Çizelge 4.8 GSP için parametre seçimleri

Problem No	Düğüm Sayısı	Parçacık Sayısı	MAXITER	LSITER
1	8	3	120	30
2	9	3	135	30
3	10	3	150	30
4	11	4	165	30
5	12	4	180	40
6	13	4	195	40
7	14	4	210	40
8	16	4	240	40
9	18	4	270	50
10	20	4	300	50
11	22	5	330	50
12	24	5	360	50
13	26	5	390	50
14	28	5	420	50
15	30	5	450	50

5.4 Uygulama Sonuçları

GSP tarzı optimizasyon problemlerini çözmek üzere önerilen Em sezgiselinin performansını test etmek üzere farklı hacimlerdeki rotalar hem GAMS(2013) yazılımı, hem TB algoritması hem de EM algoritması ile çözülmüş ve sonuçlar karşılaştırılmıştır. TB ve EM sezgisel algoritmalarının yazılımında Excel 2012 platformu üzerinde VBA editörü kullanılmıştır.

Hem Excel formüllerinin hem de VBA editörünün birlikte kullanılabiliyor olması bu çalışmaya çok büyük ölçüde esneklik ve ergonomi kazandırmıştır.

Tasarlanan sezgisel algoritmanın çalışma performansı üzerinde çalışılan bilgisayarın donanım seviyesi ile de yakından ilişkilidir. Bu tez çalışmasında Windows 7 işletim sistemi, Office 2012 üzerinde çalışılmıştır. İşletim sistemi olarak i5-2400 CPU @ 3.10 GHz ve 4.00 GB Ram kapasitesine sahip 64 bitlik bir platform kullanılmıştır.

TB sezgiselinin parametreleri olarak soğuma çizelgesi ve iterasyon sayısı parametreleri seçilmiştir. Soğuma çizelgesi olarak geometrik soğuma seçilmiş ve $\alpha=0.9$ seçilmiştir. İterasyon sayısı olarak ise iterasyon sayısı=n*50 seçilmiştir.

Yapılan denemelerde görülmüştür ki EM sezgiselinin TB'ye göre çalışma performansı başarılıdır. Çözümü bulma başarıları açısından düşük hacimli problemlerde bu iki sezgisel başa baş bir başarı grafiği sergilerken sıralama içerisindeki şehir sayısı yükseldikçe EM daha başarılı bir görüntü çizmeye başlamıştır (Çizelge 4.9).

Çizelge 4.9 Sonuçlar Tablosu

		EM		TB		GAMS
Problem	Çiftlik Sayısı	Amaç Fonksiyonu	İşlem Süresi Zaman (sn)	Amaç Fonksiyonu	İşlem Süresi Zaman (sn)	Amaç Fonksiyonu
1	8	361,82	0,60	361,82	2,93	361,82
2	9	387,10	1,23	387,10	2,35	387,10
3	10	393,79	5,44	393,79	6,41	393,79
4	11	407,05	4,30	411,46	5,06	407,05
5	12	420,93	7,91	426,02	10,09	420,93
6	13	455,45	9,33	458,58	13,30	455,45
7	14	479,37	8,12	482,59	12,68	473,41
8	15	489,47	9,86	490,13	12,23	486,12
9	16	496,06	14,22	494,35	18,52	491,14
10	17	510,89	12,69	511,52	13,62	502,56
11	18	519,09	15,30	524,45	17,37	514,58
12	19	562,32	14,90	557,39	19,82	556,69
13	20	597,45	19,62	598,25	22,48	588,36
14	21	606,83	17,50	604,39	17,92	601,87
15	22	628,10	22,14	628,84	23,33	623,56

Ağı oluşturan çiftlik sayısı arttıkça optimal sonuç üzerinde meydana gelen hata yüzdeleri Denklem 4.8'de görüldüğü gibi hesaplanır.

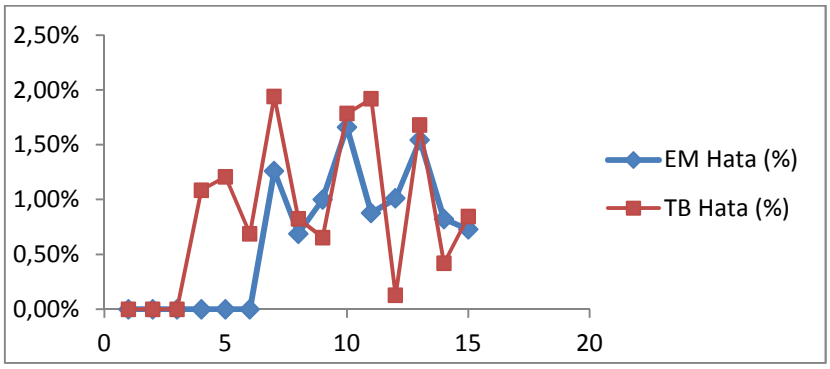
$$\%Fark = \frac{f(x^{Gams}) - f(x^{sezgisel})}{f(x^{Gams})} \quad (4.8)$$

Farklar hesaplandıktan sonra oluşan hata değerleri Çizelge 4.10'da gösterilmiştir.

Çizelge 4.10 Hatalar Tablosu

Problem	EM Hata (%)	TB Hata (%)
1	0,00%	0,00%
2	0,00%	0,00%
3	0,00%	0,00%
4	0,00%	1,08%
5	0,00%	1,21%
6	0,00%	0,69%
7	1,26%	1,94%
8	0,69%	0,83%
9	1,00%	0,65%
10	1,66%	1,78%
11	0,88%	1,92%
12	1,01%	0,13%
13	1,54%	1,68%
14	0,82%	0,42%
15	0,73%	0,85%

Şekil 4.7'de görülüğü üzere şehir sayısı arttıkça EM sezgiselinin hata yüzdesi TB sezgiseline göre daha düşük seyretmektedir. Zaten hata yüzde payları küçük olduğu için EM sezgiselinin çözüm kalitesinin yüksek olduğu söylenebilir.



Şekil 4.7 Hatalar Grafiği

BÖLÜM 5

SONUÇ VE ÖNERİLER

Bu çalışmada ilk olarak 2003 yılında önerilen ve global optimizasyon problemlerinde kullanılmak üzere tasarlanmış olan elektromanyetizma sezgisel algoritmasının gezgin satıcı problemleri gibi kesikli ve kısıtlı problemler üzerindeki çözüm yetenekleri incelenmiştir. Esasen, elektromanyetizma sezgisel algoritması Coulomb Kanunu'nun itme ve çekme sistemini kullanarak optimal sonuca yaklaşmayı hedefleyen ve literatüre nispeten yeni girmiş bir meta-sezgisel algoritmadır. Yöneylem problemlerini çözebilmek için ilk olarak rassal anahtar yöntemi ile klasik elektromanyetizma algoritması birleştirilerek kesikli ve kısıtlı problemleri çözmeye elverişli bir arama mekanizması oluşturulmuştur.

Oluşturulan yeni mekanizmanın çözüm yeteneklerini test etmek adına kapasite kısıtını göz önünde bulundurmadan farklı büyüklüklerdeki gezgins atıcı problemleri üzerinde deneyler yapılmış ve optimalliğe yaklaşma becerisi ile çözüm süreleri incelenmiştir. Elektromanyetizma sezgisel algoritmasının çözüm yetenekleri GAMS(2013) ve tavlama benzetimi sezgisel algoritması ile karşılaştırılmıştır.

Yapılan incelemeler için, GAMS programı kullanılarak farklı büyüklükte doğrusal modeller oluşturulmuş ve bu modellerde kullanılan verilen tavlama benzetimi ile elektromanyetizma sezgisellerine uygulanarak sonuçlar ve işlem süreleri karşılaştırılmıştır.

Gezgin satıcı problemlerini matematiksel programlama teknikleri kullanarak çözmeyi zor kılan en büyük engel sub-tour problemidir. Yani oluşan çözüm içerisinde tek bir tur yerine bir çok alt turların oluşması durumudur. Küçük ölçekli problemler (n<20) için bu problemin önüne geçmek için alt tur eliminasyon kısıtı eklenerek bu sorunun önüne geçilebilmektedir. Fakat rota içinde bulunan düğüm noktaları arttıkça alt tur sayıları çok fazla arttığı için bir süreden sonra gezgin satıcı problemi gibi problemlerin matematiksel modelleme ile çözülmesi uygun olmamaktadır. Bu durumda devreye giren sezgisel ve meta-sezgisel algoritmalar ile olumlu sonuçlar alınmıştır. Elektromanyetizma algoritmasının gezgin satıcı problemleri için farklı problem büyüklüklerinde son derece olumlu sonuçlar verdiği gözlemlenmiştir. Fakat algoritmanın çalışma hızı özellikle Tavlama benzetimi algoritmasına nazaran yavaş kalmaktadır. Bundaki en önemli gerekçelerden biri yerel arama prosedürünün yapısında bulunan ikili karşılaştırma sürecinin çok uzun sürmesidir. Bunun haricinde algoritma kompleks bir yapıya sahip olup yapısında büyük bir kod yığınına sahiptir. Algoritmanın çalışma hızını arttırmak için Yerel arama prosedüründe kullanılan aç gözlü arama algoritması (Greddy Search) yerine daha hızlı çalışan arama algoritmaları (The cross search algorithm- 1990) kullanılabilir. Çünkü arama esnasında yapılan ikili karşılaştırmalar problemin hacmi yükseldikçe daha fazla vakit almaktadır. Aç gözlü arama algoritmasının zaman karmaşıklığı $O(b^m)$'dir. Elektromanyetizma algoritmasının işlem süresinin uzun sürmesinin bir diğer sebebi ise tamamıyla rassal başlangıç çözümleri ile işe başlamasıdır. Bunun önüne geçmek için algoritmaya bir başlangıç çözümü oluşturma prosedürü eklenerek hibrit bir yapı oluşturulabilir. Bu çalışma zaman probleminin önüne ciddi ölçüde geçebilir.

Excel VBA ile kodlanan algoritmanın içerisinde bulunan grafik ve görsellik amaçlı yazılmış kodlar sebebiyle işlem süresi yükselmiştir MATLAB veya C# gibi editörlerle yazılan bir kodun çalışma süresi daha kısa olacaktır.

Diğer tüm meta-sezgisel algoritmalarda olduğu gibi elektromanyetizma algoritması içinde algoritma parametreleri çalışma performansı açısından büyük önem taşımaktadır. Bu problemin önüne geçmek için amaca özelleştirilmiş algoritmalar üzerinde deneyler yaparak en iyi parametre kombinasyonları tespit edilerek algoritmanın yapısına entegre edilebilir.

KAYNAKLAR

[1] Lawler, E.L., Lenstra, J.K., Rinnooy Kan,A.H.G. ve Shmoys D.B., (1985), The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization, Wiley & Sons, New York

[2] Lin, S. ve Kernighan, B. W., (1973), "An effective heuristic algorithm for the traveling-salesman problem", Oper. Res., 21:498-516.

[3] Geng, X., Chenb, Z.,Yanga, W., Shia, D. ve Zhaoa K.,(2011),"Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search", Applied Soft Computing, 11(4):3680-3689.

[4] Jun-man, K. ve Yi, Z., (2012),"Application of an Improved Ant Colony Optimization on Generalized Traveling Salesman Problem", Energy Procedia,17(Part A):319-325.

[5] Gendreau, M., Laporte, G. ve Semet, F. (1998). A tabu search heuristic for the undirected selective travelling salesman problem. European Journal of Operational Research ,106: 539-545.

[6] Pedro, O. ve Saldanha, R. (2013). A Tabu Search Approach for the Prize Collecting Traveling Salesman Problem. Electronic Notes in Discrete Mathematics, 41: 261-268.

[7] Nagata, Y. ve Soler, D. (2012). A new genetic algorithm for the asymmetric traveling salesman problem. Expert Systems with Applications. 39: 8947–8953.

[8] Shi, X.H., Liang, Y.C., Lee, H.P., Lu, C. ve Wang, Q.X. (2007). Particle swarm optimization-based algorithms for TSP and generalized TSP, Information Processing Letters, 103: 169–176.

[9] Chen, S.-H. ve Chien, C.Y., (2011) Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques. Expert Systems with Applications, 38: 14439–14450

[10] Birbil, S., and Fang, S.-C. (2003). An electromagnetism-like mechanism for global optimization. Journal of Global Optimization, 25: 263–282.

[11] Birbil, S., Fang, S.-C., and Sheu, R.-L. (2004). On the convergence of a population-based global optimization algorithm. Journal of Global Optimization, 30: 301–318.

[12] Taheri, S.H., Ghazvini, H., Saberi-Nadjafi, J. ve Biazar, J., (2007). A hybrid of the restarted Arnoldi and electromagnetism meta-heuristic methods for calculating eigenvalues and eigenvectors of a non-symmetric matrix. Applied Mathematics and Computation. 191: 79–88

[13] Wu, P., Yang K.-J., ve Huang B.-Y. (2007). A revised EM-like mechanism for solving the vehicle routing problems. In Second international conference on innovative computing, information and control, ICICIC '07 (p. 181).

[14] Yurtkuran, A., ve Emel, E. (2010). A new hybrid electromagnetism-like algorithm for capacitated vehicle routing problems. Expert Systems with Applications, 37: 3427–3433.

[15] Jolai, F., 19.Tavakkoli-Moghaddam, R., Golmohammadi, A. ve Javadi, B., (2012), An Electromagnetism-like algorithm for cell formation and layout problem, Expert Systems with Applications, 39: 2172–2182.

[16] Filipovic. V., Kartelj, A. ve Matic, D. (2013), An electromagnetism metaheuristic for solving the Maximum Betweenness Problem, Applied Soft Computing, 13: 1303–1313

[17] Maenhout, B., ve Vanhoucke, M. (2007). An electromagnetic meta-heuristic for the nurse scheduling problem. Journal of Heuristics, 13: 359–385.

[18] Debels, D., ve Vanhoucke, M. (2006). The electromagnetism meta-heuristic applied to the resource-constrained project scheduling problem. Lecture Notes in Computer Science, 3871: 259–270.

[19] Chang, P.-C, Chen, S.-H. ve Fan, C.-Y. (2009), A hybrid electromagnetism-like algorithm for single machine scheduling problem. Expert Systems with Applications,36: 1259–1267.

[20] Sels, V., ve Vanhoucke, M. (2010). A hybrid Electromagnetism-like Mechanism/tabu search procedure for the single machine scheduling problem with a maximum lateness objective. Computers and Industrial Engineering 67: 44–55.

[21] Chen, S.-H., Chang, P.-C., Chan, C.-L., ve Mani, V. (2007). A hybrid electromagnetism like algorithm for the single machine scheduling problem. Lecture notes in Computer Sciences, 4682: 543–552.

[22] Chao, C.-W, Liao, C.-J. (2012), A discrete electromagnetism-like mechanism for single machine total weighted tardiness problem with sequence-dependent setup times. Applied Soft Computing. 12: 3079–3087.

[23] Javadian, N, Alikhani, M.G, Tavakkoli-Moghaddam, R., (2008), A Discrete Binary Version of the Electromagnetism-Like Heuristic for Solving Traveling Salesman Problem, Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence Lecture Notes in Computer Science Volume 5227: 123–130

[24] Ateş, E., (2012), Karınca Kolonisi Optimizasyonu Algoritmaları İle Gezgin Satıcı Probleminin Çözümü Ve 3 Boyutlu Benzetimi, Basılmamış Lisans Tezi, Ege Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, İzmir.

[25] Nabiyev, V. V., (2007), Yapay Zeka - İnsan Bilgisayar Etkileşimi, Seçkin Yayıncılık, Ankara.

[26] Helsgaun, K., (2006),"An effective implementation of k-opt moves for the Lin-Kernighan TSP heuristic", Writings on Computer Science, Roskilde University.

[27] Mattsson, P., (2010), The Asymmetric Traveling Salesman Problem, Uppsala Universitet.

[28] Aras, N., Boyacı, B., Koşucuoğlu, D. ve Aksen, D., (2007), Karlı Gezgin Satıcı Problemi için Sezgisel Yöntemler, Yöneylem Araştırması / Endüstri Mühendisliği 27. Ulusal Kongresi, İzmir.

[29] Koç, Ö.N., (2012). Zaman Pencereli Gezgin Satıcı Problemi İçin Yeni Karar Modelleri, Başkent Üniversitesi, Fen Bilimleri Enstitüsü, Basılmamış Yüksek Lisans Tezi, Ankara.

[30] Zhang, B. ve Peng, J., (2014), "Uncertain Traveling Salesman Problem". Erişim linki: http://orsc.edu.cn/online/110731.pdf, 03 Şubat 2014

[31] Yadlapalli, S., Rathinam, S. ve Darbha, S. , (2012), "3-Approximation Algorithm for a Two Depot, Heterogeneous Traveling Salesman Problem", Optimization Letters, 6(1): 141-152.

[32] Haskell, B.W., Toriello, A., Poremba, M. ve Epstein, D.J., (2013), "A Dynamic Traveling Salesman Problem with Stochastic Arc Costs", Department of Industrial and Systems Engineering University of Southern California Los Angeles, California.

[33] Kara, İ. ve Demir, E., (2006), "Genelleştirilmiş Gezgin Satıcı Poblemi İçin Yeni Tamsayılı Karar Modelleri", Yöneylem Araştırması / Endüstri Mühendisliği 26. Ulusal Kongresi, Kocaeli Üniversitesi, 3-5 Temmuz, Kocaeli.

[34] Helvig, C.S., Robins, G. ve Zelikovsky, A., (2003), "The Moving-Target Traveling Salesman Problem" , Journal of Algorithms, 49:153-174.

[35] Kara, İ., Derya, T., Demir, E. ve Bektaş, T., (2005), "Genelleştirilmiş Gezgin Satıcı Probleminin Tamsayılı Doğrusal Karar Modeli", Yöneylem Araştırması / Endüstri Mühendisliği 25. Ulusal Kongresi, Koç Üniversitesi, 4-6 Temmuz, İstanbul.

[36] Matai, R., Singh, S.P. ve Mittal, M.L., (2010), "Traveling Salesman Problem: An Overview of Applications, Formulations, and Solution Approaches" in Traveling Salesman Problem, Theory and Applications Donald Davendra (Ed.), 1-24, InTech, Croatia.

[37] Dantzig, G.B., Fulkerson D. R. ve Johnson, S. M., (1954), "Solution of a large-scale traveling-salesman problem", Oper. Res., 2:393-410.

[38] Karp, R. M., (1972), "Reducibility among combinatorial problems", Complexity of Computer Computations", Plenum Press, New York, 85-103.

[39] Diaby, M., (2007), "The Traveling Salesman Problem: A Linear Programming Formulation", WSEAS Transactions on Mathematics, 6(6):745–754.

[40] Malandraki, C. ve Dial, RB.,(1996), "A Restricted Dynamic Programming Heuristic Algorithm for Time Dependent Traveling Salesman Problem", European Journal of Operations Research, 90(1):45–55.

[41] Padherg, M. ve Rinaldi, R., (1987), "Optimization of a 532-City Symmetric Travelling Salesman Problem by Branch and Cut", Operations Research Letters, 6(1): 1-7.

[42] Padherg, M. ve Rinaldi, G., (1988), "Branch-and-Cut Approach to a Variant of the Traveling Salesman Problem", Journal of Guidance, Control, and Dynamics, 11(5): 436–440.

[43] Lin, S., Kernighan, B., (1973), "An Effective Heuristic Algorithm for the Traveling-Salesman Problem", Operations Research, 21(2):498-516.

[44] Punnen, A., Margot, F. ve Kabadi, S., (2003), "TSP Heuristics: Domination Analysis and Complexity", Algorithmica, 35:111–127.

[45] Martin, O., Otto, S. ve Felten, E., (1991), "Large-step markov chains for the traveling salesman problem", Complex Systems, 5(3):299-326.

[46] Kirkpatrik, S., (1984), “Optimization by Simulated Annealing: Quantitative Studies”, Journal of Statistical Physics, 34: 975-986.

[47] Wu, JB., Xiong, SW. ve Xu, N.,(2007), “Simulated Annealing Algorithm Based on Controllable Temperature for Solving TSP”, Application Research of Computers, 24(5): 66–89.

[48] Dam, M. Ve Zachariasen, M., (1996), “Tabu Search on the geometric travelling salesman problem. Meta-heuristics: Theory and Applications”, Kluwer Academic Publishers, Boston, 571-587.

[49] Wu, Y. ve Zhou, X., (2008), “Meliorative Tabu Search Algorithm for TSP Problem”, Journal of Computer Engineering and Applications, 44(1): 57–59.

[50] Hurkens, C.A.ve Woeginger, G.J., (2004), “On the Nearest Neighbor Rule for the Traveling Salesman Problem”, Operations Research Letters, 32(1): 1-4.

[51] Dry, M., Preiss, K. ve Wagemans, J.,(2012), “Clustering, Randomness, and Regularity: Spatial Distributions and Human Performance on the Traveling Salesperson Problem an Minimum Spanning Tree Problem”, The Journal of Problem Solving, 4: 1, 2.

[52] Akyol, S., Alataş, B., (2012), “Güncel Sürü Zekası Optamizasyon Algoritmaları”, Nevşehir Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 1(1):36-50

[53] Fogel, M.B., (1986), “The Genetic Algorithm for TSP”. IEEE Transaction on systems, 16:1-13.

[54] Albayrak, M. ve Allahverdi, N., (2011), “Development a New Mutation Operator to Solve the Traveling Salesman Problem by Aid of Genetic Algorithms”. Expert Systems with Applications, 38:1313–1320.

[55] Dorigo, M. ve Gambardella, L. M., (1997), “Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem”, IEEE Transactions on Evolutionary Computation, 1:53–66.

[56] Xu C., Xu J., Chang H.,(2009), “Ant Colony Optimization Based on Estimation of Distribution For The Traveling Salesman Problem”, Fifth international conference on natural computation, Tianjin, China, 19–23.

[57] Wong, L., Low, Y.H.P., Chong, C.S., (2010), “ A Bee Colony Optimization with Local Search for Travelling Salesman”, International Journal on Artificial Intelligence Tools, 19(3): 305–334

[58] Karaboğa, D. ve Görkemli, B., (2011), "A combinatorial Artificial Bee Colony Algorithm for Traveling Salesman Problem", INISTA 2011 - 2011 International Symposium on Innovations in Intelligent Systems and Applications, İstanbul, Turkey, 50-53.

[59] Nakaguchi, T., Jin'no, K. ve Tanaka, M., (2000), "Hysteresis Neural Networks for Solving Travelling Salesman Problems", 2000 IEEE International Symposium on Circuits and Systems, Switzerland, 03:153-156.

[60] Li, M., Yi, Z. ve Zhu, M., (2009), "Solving TSP by using Lotka-Volterra Neural Networks", Neurocomputing, 72(16–18): 3873–3880.

[61] De Castro, L.N. ve Von Zuben, F.J., (2000), "The Clonal Selection Algorithm With Engineering Applications". In Proceedings of GECCO, 36-39.

[62] Baykasoglu, A., Saltabas, A., Tasan, A. S. ve Subulan, K., (2012), "Yapay Bagışıklık Sisteminin Çoklu Etmen Benzetim Ortaminda Realize Edilmesi ve GSP Uygulanması", Journal of the Faculty of Engineering ve Architecture of Gazi University. 27(4):901-909.

[63] Wang, K.P., Huang, L., Zhou, C.G., Pang ve W., (2003), "Particle swarm optimization for traveling salesman problem. In Machine Learning and Cybernetics", 2003 International Conference on Machine Learning and Cybernetics, China 3:1583-1585.

[64] Shah-Hosseini, H., (2007), "Problem Solving by Intelligent Water Drops", 2007 IEEE Congress on Evolutionary Computation (CEC 2007), Siingapore, 3226-3231

[65] Javadian, N., Alikhani, M.G. ve Tavakkoli-Moghaddam, R., (2008), "A Discrete Binary Version of the Electromagnetism-Like Heuristic for Solving Traveling Salesman Problem". Lecture Notes in Artificial Intelligence, 5227: 123-130.

[66] Özkan, B., Cevre, U. ve Uğur, A., (2008), "Melez bir eniyileme yöntemi ile rota planlama", Akademik Biliim 2008, Çanakkale.

[67] Gambardella, L. M. ve Dorigo, M., (1996), "Solving symmetric and asymmetric TSPs by ant colonies", Proc. of IEEE Int. Conf. on Evol. Computation, 622-627.

[68] Padberg, M. W. ve Rinaldi, G., (1991), "A branch-and-cut algorithm for the resolution of symmetric traveling salesman problems", SIAM Review, 33:60-100.

[69] Grötchel, M. ve Holland, O., (1991), "Solution of large scale symmetric travelling salesman", Math. Programming, 51:141-202.

[70] Applegate, D., Bixby, R., Chvàtal V. ve Cook, W., (1995), “Finding Cuts in the TSP (A Preliminary Report)”, DIMACS, Tech. Report 95-05.

[71] Bellmore, M. ve Malone, J.C., (1972), “Pathology of traveling-salesman subtour-elimination algorithms”, Oper. Res., 19:278-307.

[72] Rosenkrantz, D. E. Stearns, R. E. ve Lewis II, P. M., (1977), “An analysis of several heuristics for the traveling salesman problem”, SIAM J. Comput., 6:563-581.

[73] Laporte, G., (1992), “The traveling salesman problem: An overview of exact and approximate algorithms, Eur”, J. Oper. Res., 59:231-247.

[74] Melamed, I. I., Sergeev, S. I. ve Sigal, I., (1989), “The traveling salesman problem: Approximate algorithms“, Avtomat, Telemekh., 11:3-26.

[75] Gerşil, M. ve Palamutçuoğlu, T., (2013), “Ders Çizelgeleme Probleminin Melez Genetik Algoritmalar İle Performans Analizi”, Niğde Üniversitesi İİBF Dergisi, 6(1): 242-262.

[76] Yiğit, V. ve Türkbey, O., (2003), “Tesis Yerleşim Problemlerine Sezgisel Metotlarla Yaklaşım, Gazi Üniversitesi Mühendislik. Mimarlık Fakültesi Dergisi”, 18(4): 45-56.

[77] Lourenço, H.R., Martin, O.C. ve Stutzle, T., (2001), “A Beginner’s Introduction to Iterated Local Search”, 4th Metaheuristics International Conference (MIC’01), Porto, Portugal, 16-20.

[78] Juan, A., Lourenço, H.R., Mateo, M., Luo, R. ve Castella, Q., (2013), “Using Iterated Local Search For Solving the Flow-Shop Problem: Paralleization, Parametrization and Randomization Issues”, International Transactions in Operational Research, 21:103-126.

[79] Kirkpatrick, S., Gelatt, C. D. ve Vechi, M. P.,(1983), “Optimization by Simulated Annealing”, Science, 220(4598): 671-680.

[80] Kumbharana, N.S. ve Pandey, G.M., (2013), “A Comparative Study of ACO, GA and SA for Solving Travelling Salesman Problem. International Journal of Societal Applications of Computer Science”, 2(2):224-228.

[81] Söke, A. ve Bingül, Z., (2005), İ”ki Boyutlu Giyotinsiz Kesme Problemlerinin Benzetilmiş Tavlama Algoritması ile Çözümlerinin İncelenmesi”, Politeknik Dergisi, 8(1)25-35.

[82] Luke, S., (2013), Essentials of Metaheuristics, Second Edition, Lulu, USA.

[83] Tokgöz, A. ve Bulkan, S., (2013), "Weapon Target Assignment with Combinatorial Optimization Techniques". (IJARAI) International Journal of Advanced Research in Artificial Intelligence, 2(7):39-50

[84] Glover, F., (1986), "Future Paths For İnteger Programming and Links to Artificial İntelligence", Computer and Operation Research, 13:533-549.

[85] Hansen, P., (1986), "The Steepest Ascent Mildest Descent Heuristic for Combinatorial Programming", Proceedings of the Congress on Numerical Methods in Combinatorial Optimization, Capri, Italy.

[86] Cura, T., (2008), Modern Sezgisel Teknikler ve Uygulamaları, Papatya Yayıncılık, İstanbul.

[87] Karaboğa, D., (2011) Yapay Zekâ Optimizasyon Algoritmaları, Genişletilmiş II. Basım, Nobel Yayın Dağıtım, İstanbul.

[88] Güney, K. ve Akdağlı, A., (1999), "Tabu Araştırma Algoritması İle Lineer Anten Dizisinin Genlik Uyarım Katsayılarının Belirlenmesi", 8. Ulusal Elektrik Elektronik Bilgisayar Mühendisliği Kongresi, Gaziantep Üniversitesi, 456-459.

[89] Pollard, J.M., (1978), "Monte Carlo Methods for Index Computation (mod p)", Mathematics of Computation, 32(143):918–924.

[90] Van Oorschot, P. ve Wiener, M., (1996), "On Diffie-Hellman Key Agreement with Short Exponents", Advances in Cryptology, 332-343.

[91] Duta, L., Henrioud, J.M. ve Caciula, I., (2007), "A Real Time Solution to Control Disassembly Processes", Proceedings of the 4th IFAC Conference on Management and Control of Production and Logistics, Sibiu.

[92] Demirtaş, Y.E. ve Keskintürk, T., (2011), "Kanguru Algoritması ve Gezgin Satıcı Problemine Uygulanması", İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi, 10 (19): 51-63.

[93] Karaboğa, D., (2005), An İdea Based On Honey Bee Swarm For Numerical Optimization. Technical Report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department.

[94] McCaffrey, J., (2014), "Natural Algorithms: Use Bee Colony Algorithms to Solve Impossible Problems", MSDN Magazines, erişim linki: http://msdn.microsoft.com/en-us/magazine/gg983491.aspx , 10 Haziran 2014

[95] Goldberg, D.E., (1989), Genetic Algorithms in Search Optimization and Machine Learning, Addison Wesley Publishing Company, USA.

[96] Reeves, C.R., (1995), Modern heuristic techniques for combinatorial problems, McGraw-Hill Book Company Inc., Europe.

[97] Obitko, M., (1998), Genetic Algorithms, (Online), erişim linki: http://www.obitko.com/tutorials/genetic-algorithms/index.php , 16 Mart 2014

[98] Sarker, R. ve Newton, C., (2002), "A Genetic Algorithm for Solving Economic Lot Size Scheduling Problem", Computer & Industrial Engineering, 12(5):195-196.

[99] Cheng, R., Gen, M. ve Yasuhiro, T., (1999), "A Tutorial Survey of Job-Shop Scheduling Problems Using Genetic Algorithms, Part II: Hybrid Genetic Search Strategies", Computers and Industrial Engineering, 36: 343-364.

[100] Keskintürk, T. ve Şahin, S., (2009), "Doğrusal Olmayan Regresyon Analizinde Gerçek Değer Kodlamalı Genetik Algoritma", İTÜ Sosyal Bilimler Dergisi, 8:167-178.

[101] Keskintürk, T. ve Söyler, H., (2006), "Global Karınca Kolonisi Optimizasyonu", Gazi Üniversitesi Mimarlık ve Mühendislik Dergisi, 21:689-698.

[102] Söyler, H. ve Keskintürk, T., (2007), "Karınca Kolonisi Algoritması İle Gezen Satıcı Probleminin Çözümü", 8. Türkiye Ekonometri ve İstatistik Kongresi, Malatya, 1-11.

[103] Dorigo, M., (1992), Ottimizzazione, apprendimento automatico, ed algoritmi basati su metafora naturale (Optimization, Learning and Natural Algorithms), Ph.D.Thesis, Politecnico di Milano, Italy, in Italian. DT.01-POLIMI92.

[104] Brownlee, J., (2012), Clever Algorithms, Nature-Inspired Programming Recipes, Open Source Book.

[105] Schoen, F., (1989), A wide class of test functions for global optimization, Proceedings of the 3rd International Conference on Genetic Algorithms, 51–60.

[106] More, B. J. ve Wu, Z., (1995), "Global Smoothing and Continuation for Large-scale Molecular Optimization", Argonne National Laboratory, Illinois: Preprint MCS-P539-1095.

[107] Michalewicz, Z. (1994), Genetic Algorithms + Data Structures = Evolution Programs, Springer, Berlin.

[108] Hart, W. E., (1994), Adaptive Global Optimization with Local Search. PhD Thesis, University of California, San Diego.

[109] Glover, J. K. F. ve Laguna,M., (1995), "Genetic algorithms and tabu search: Hybrids for optimization", Computers and Operations Research 22: 111–134.

[110] Solis, F. J. ve Wets, R. J-B. (1981), “Minimization by random search techniques”, Mathematics of Operations Research 6: 19–30.

[111] Törn, A., Ali, M. M. ve Viitanen, S., (1999), “Stochastic global optimization: Problem classes and solution techniques”, Journal of Global Optimization 14: 437–447.

[112] Huyer,W. ve Neumaier, A. (1999), “Global optimization by multilevel coordinate search. Journal of Global Optimization”, 14: 331–355.

[113] Rastrigin, L. A., (1974), Systems of extremal control.

[114] Mühlenbein H., Schomisch D. ve Born J., (1991), “The Parallel Genetic Algorithm as Function Optimizer “, Parallel Computing, 17:619–632

[115] Wikipedia the Free Encyclopedia, Rastrigin function http://en.wikipedia.org/wiki/Rastrigin_function , 12 Ekim 2013

[116] Bean, J.C., (1994), “Genetic Algoritms and Random Keys for Sequencing and Optimization”, ORSA JOURNAL on Computing, 6(2): 154-160

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı :Burak TOPCU

Doğum Tarihi ve Yeri :16.01.1985

Yabancı Dili :Fransızca ve İngilizce

E-posta :buraktopcu@gmail.com

ÖĞRENİM DURUMU

Derece	**Alan**	**Okul/Üniversite**	**Mezuniyet Yılı**
Lisans	Endüstri Mühendisliği	Sakarya Üniversitesi	2010
Lise	Fen	Galatasaray Lisesi	2005