T.C.
PAMUKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

ELEMAN SAYISI KISITLI PORTFÖY OPTİMİZASYONU PROBLEMİ İÇİN YAPAY ARI KOLONİSİ ALGORİTMASI TEMELLİ BİR ÇÖZÜM YAKLAŞIMI

YÜKSEK LİSANS TEZİ

ÖKKEŞ ERTENLİCE

DENİZLİ, ARALIK - 2018

T.C.
PAMUKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

ELEMAN SAYISI KISITLI PORTFÖY OPTİMİZASYONU PROBLEMİ İÇİN YAPAY ARI KOLONİSİ ALGORİTMASI TEMELLİ BİR ÇÖZÜM YAKLAŞIMI

YÜKSEK LİSANS TEZİ

ÖKKEŞ ERTENLİCE

DENİZLİ, ARALIK - 2018

KABUL VE ONAY SAYFASI

Ökkeş Ertenlice tarafından hazırlanan "**Eleman sayısı kısıtlı portföy optimizasyonu problemi için yapay arı kolonisi temelli bir çözüm yaklaşımı**" adlı tez çalışmasının savunma sınavı 28.12.2018 tarihinde yapılmış olup aşağıda verilen jüri tarafından oy birliği / oy çokluğu ile Pamukkale Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalı Yüksek Lisans Tezi olarak kabul edilmiştir.

Jüri Üyeleri	İmza
Danışman Doç. Dr. Can Berk KALAYCI Pamukkale Üniversitesi	.....
Üye Doç. Dr. Olcay POLAT Pamukkale Üniversitesi	.....
Üye Dr. Öğretim Üyesi Yusuf ŞAHİN Burdur Mehmet Akif Ersoy Üniversitesi	.....

Pamukkale Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu`nun 09/01/2019 tarih ve 02/13 sayılı kararıyla onaylanmıştır.

.....

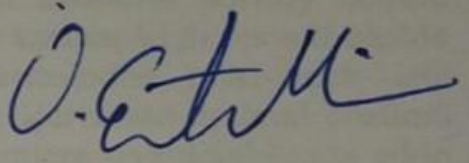
Prof. Dr. Uğur YÜCEL

Fen Bilimleri Enstitüsü Müdürü

Bu tez çalışması TUBİTAK tarafından 214M224 nolu proje ile desteklenmiştir.

Bu tezin tasarımı, hazırlanması, yürütülmesi, araştırmalarının yapılması ve bulgularının analizlerinde bilimsel etiğe ve akademik kurallara özenle riayet edildiğini; bu çalışmanın doğrudan birincil ürünü olmayan bulguların, verilerin ve materyallerin bilimsel etiğe uygun olarak kaynak gösterildiğini ve alıntı yapılan çalışmalara atfedildiğine beyan ederim.

ÖĶKEŞ ERTENLİCE



ÖZET

ELEMAN SAYISI KISITLI PORTFÖY OPTİMİZASYONU PROBLEMİ İÇİN YAPAY ARI KOLONİSİ ALGORİTMASI TEMELLİ BİR ÇÖZÜM YAKLAŞIMI

YÜKSEK LİSANS TEZİ

ÖKKEŞ ERTENLİCE

PAMUKKALE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

(TEZ DANIŞMANI:DOÇ. DR. CAN BERK KALAYCI)

DENİZLİ, ARALIK - 2018

Literatürde en çok çalışılan portföy optimizasyonu varyantlarından birisi klasik ortalama-varyans modeline kısıtlar eklenerek konveks kuadratik programlı problemden karma tamsayılı kudratik probleme dönüşmüş NP-Zor sınıfta olan kısıtlı portföy opptimizasyonu problemidir. Eklenen kısıtların portföy boyutu üzerinde direkt bir etkisi vardır dolayısıyla hesaplama karmaşıklığı önemli ölçüde arttırmaktadır. Artan hesaplama karmaşıklığının üstesinden gelebilmek için araştırmacılar, kesin çözüm tekniklerinin makul süre içerisinde optimal çözümü bulmakta yetersiz kalacağı ve büyük boyutlu problemlere uygulandığında etkin olamayacağı için metasezgiseller gibi etkin çözüm algoritmalarına odaklanmışlardır. Bu çalışmada, kısıtlı portföy optimizasyonu probleminin çözümü için yapay arı kolonisi temelli çözümleri uygulanabilir olmaya zorlayan ve uygulanabilir olmayan çözümlere tölerans sağlayan bir algoritma sunulmuştur. Elde edilen sonuçlar uygulanabilir çözümlere tölerans sağlayan prosedürün hem standart yapay arı kolonisine hem de litertürdeki diğer tekniklere karşı etkinliğini göstermektedir.

ANAHTAR KELİMELER: PORTFÖY OPTİMİZASYONU, METASEZGİSELLER, YAPAY ARI KOLONİSİ

ABSTRACT

AN ARTIFICIAL BEE COLONY ALGORITHM BASED SOLUTION APPROACH FOR CARDINALITY CONSTRAINED PORTFOLIO OPTIMIZATION PROBLEM

MSC THESIS

ÖKKEŞ ERTENLİCE

PAMUKKALE UNIVERSITY INSTITUTE OF SCIENCE

INDUSTRİAL ENGİNEERİNG

(SUPERVISOR:ASSOC. PROF. DR. CAN BERK KALAYCI)

DENİZLİ, DECEMBER 2018

One of the most studied variant of portfolio optimization problems is with cardinality constraints that transform classical mean-variance model from a convex quadratic programming problem into a mixed integer quadratic programming problem which brings the problem to the class of NP-Complete problems. Therefore, the computational complexity is significantly increased since cardinality constraints have a direct influence on the portfolio size. In order to overcome arising computational difficulties, for solving this problem, researchers have focused on investigating efficient solution algorithms such as metaheuristic algorithms since exact techniques may be inadequate to find an optimal solution in a reasonable time and are computationally ineffective when applied to large-scale problems. In this thesis, my purpose is to present an efficient solution approach based on an artificial bee colony algorithm with feasibility enforcement and infeasibility toleration procedures for solving cardinality constrained portfolio optimization problem. Computational solutions show the effectiveness of infeasibility toleration procedure against both standard artificial bee colony algorithm and other techniques in the literature.

KEYWORDS:PORTFOLIO OPTIMIZATION, METAHEURISTICS, ARTFICIAL BEE COLONY

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ

Sayfa

TABLO LİSTESİ

Sayfa

SEMBOL LİSTESİ

PO	**:**	Portföy Optimizasyonu
YAK	**:**	Yapay Arı Kolonisi
LK	**:**	Limit Kısıtı
ESK	**:**	Eleman Sayısı Kısıtı

ÖNSÖZ

Bu çalışmada, endüstri mühendisliğinin nispeten yeni ve popüler bir problemi olan eleman sayısı kısıtlı portföy optimizasyonunu etkin bir şekilde çözmek için bir algoritma geliştirilmiştir. Geliştirilen algoritma ve diğer araştırmacıların geliştirdikleri algoritmaları kıyaslaması için BİST veri seti literatüre kazandırılmıştır. Hayatım boyunca her zaman arkamda olan ve bana desteklerini eksik etmeyen biricik aileme en içten sevgilerimi sunarım. Ayrıca yüksek lisans eğitimim boyunca maddi ve manevi desteği ile her zaman yanımda olan sayın hocam Doç. Dr. Can Berk KALAYCI'ya minnetlerimi sunarım.

Yüksek lisans tezinde, 214M224 numaralı proje ile burs desteği veren Türkiye Bilimsel ve Teknolojik Araştırma Kurumu'na teşekkürlerimi sunarım.

1. GİRİŞ

Günümüz rekabet koşulları altında, hem bireysel hem de kurumsal yatırımcı açısından ekonomik faaliyetlerdeki karar verme süreçleri geçmişte olduğundan daha fazla önem kazanmıştır. Bu kararlardan bir tanesi de portföy seçimi ve yönetimidir. Optimal yatırım portföylerinin seçimi niceliksel finans alanında önemli bir problemdir ve bilimsel alanda yoğun ilgi görmektedir. Amerikan ekonomist Harry Markowitz'in 1952 yılında önermiş olduğu modern portföy teorisi, menkul kıymetler arasındaki ilişkiyi matematiksel olarak ifade ederek kıyaslanamayan ve çakışan iki farklı kriterin eş zamanlı olarak optimizasyonu üzerinedir. Bu kriterler; portföyün getirisi ve finansal kayıplarla ilgili risktir. Getiri, belirli bir dönem içinde yapılan bir yatırıma karşılık elde edilen geliri ifade etmektedir. Risk ise, genel anlamda gelecekte beklenmeyen sonuçlarla karşılaşma olasılığı olarak tanımlanmakta, beklenen durumlardan sapmayı ifade etmektedir. Bu nedenle, en iyi portföyü elde etmek için finansal karar verici, önceliklerine, tercihlerine, yargılarına veya sezgilerine göre bu kriterler arasında ödün vermek zorundadır. Portföy yöneticileri, yatırımcıları için risk ile getiri arasındaki en iyi dengeyi veren daha güvenilir stratejiler geliştirmeye çalışmaktadırlar.

Markowitz (1952a), portföy teorisi için ortalama- varyans (O-V) modeli ile devrimsel bir yaklaşım getirmiş ve modern portföy çağını başlatmıştır. Varyansın bir risk ölçütü olarak kullanılması bu devrimin altında yatan en önemli fikir olmuştur. Bu sayede niceliksel finansa giden yoldaki ilk kıvılcım yakılmıştır. Harry Markowitz 1991 yılında finans teorilerindeki öncü çalışmaları sayesinde Nobel ekonomi ödülüne layık görülmüş ve Portföy Optimizasyonu (PO) problemi akademik dünyanın da ilgisini çekmeye başlamıştır.

PO problemi ikinci dereceden bir amaç fonksiyonuna ve lineer kısıtlara sahiptir. NP-zor olduğu kanıtlanan bu problem için literatürde kısıtlı sayıda kesin çözüm üreten algoritma önerilmiştir. Araştırmacılar bu problemin çözümü için büyük çoğunlukla sezgisel yöntemlere başvurmuşlardır. Bu çalışmada, PO probleminin çözümü için Yapay Arı Kolonisi (YAK) temelli bir algoritma tasarlanmıştır. Geliştirilen algoritma hem standart YAK algoritması ile hem de literatürde PO

probleminin çözmek için geliştirilen diğer algoritmalar ile kıyaslanmıştır. Geliştirilen algoritma, bütün veri setleri ve bütün hata ölçütleri için, standart YAK algoritmasına üstünlük sağlamıştır. Ayrıca geliştirilen algoritma, literatürdeki diğer çözüm teknikleri ile rekabet edebilecek performansa sahiptir.

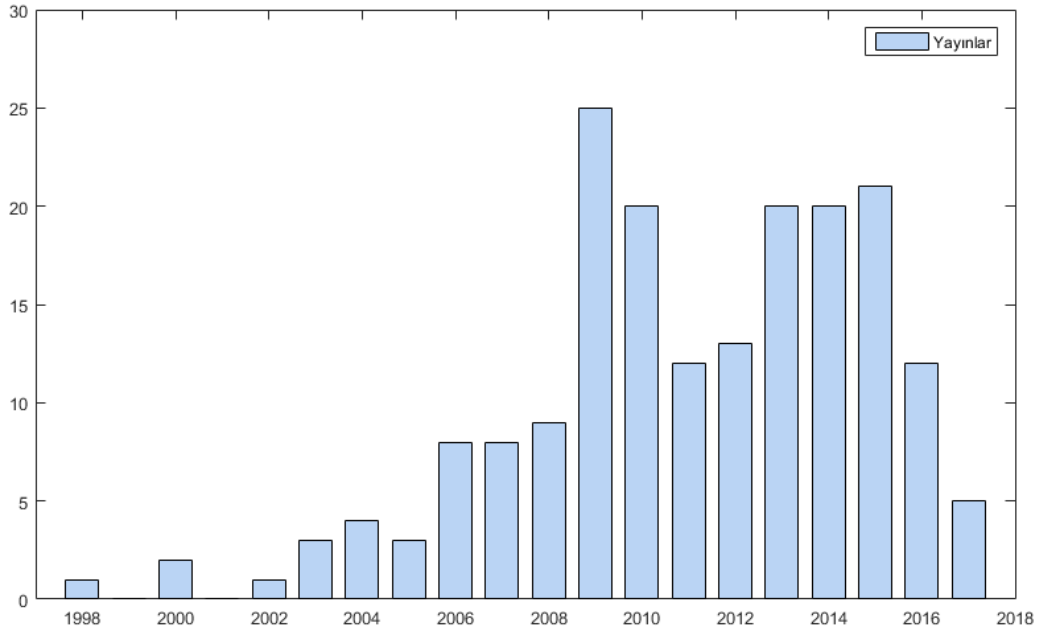
Bu çalışmanın ikinci bölümünde PO literatürüne genel bir bakış verilmiş ve bu probleme uygulanan sürü temelli algoritmalar açıklanmıştır. Üçüncü bölümde problemin matematiksel modeli, deneylerin yapıldığı veri setleri ve hata ölçütleri verilmiş, hem standart yapay arı kolonisi algoritması hem de geliştirilen algoritma anlatılmıştır. Dördüncü bölümde algoritmaların kodlanma ortamları ve parametre analizleri ve geliştirilen algoritmanın hem standart yapay arı kolonisi algoritması ile hem de literatürdeki diğer teknikler ile karşılaştırmalı sonuçları verilmiştir. Beşinci bölümde ise algoritmanın literatüre katkısı ve gelecek çalışmalardan bahsedilmiştir.

2. LİTERATÜR ÖZETİ

Markowitz (1952) modern portföy teorisinin kapısını açtıktan sonra, insanlar yatırımlarını geleneksel yöntemler yerine niceliksel yöntemleri tercih ederek yapmışlardır.

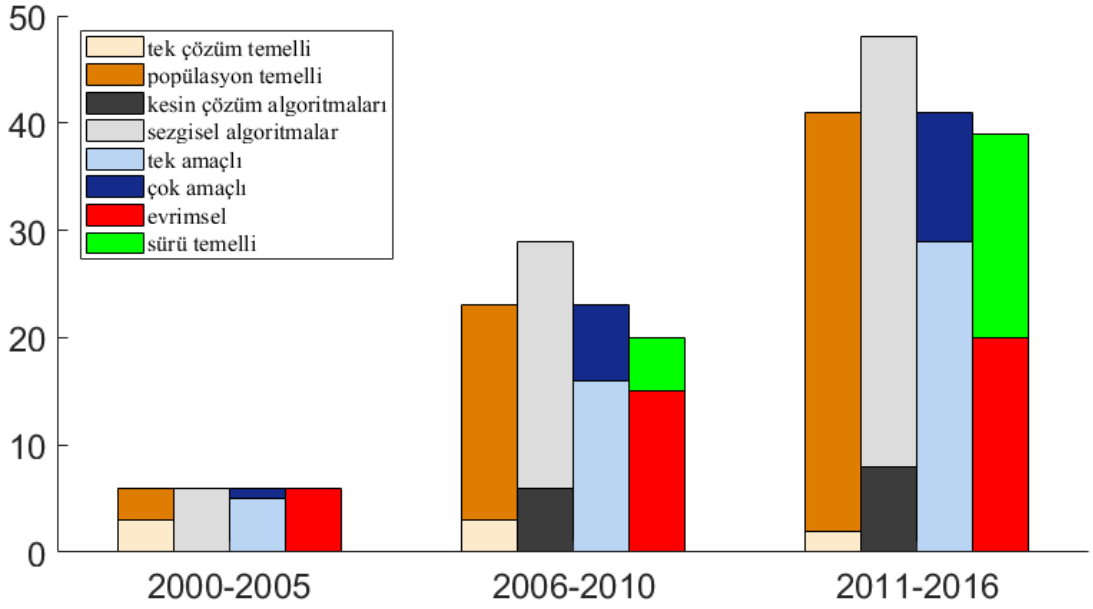
PO problemi akademik platformda da giderek artan bir ilgi görmektedir (Şekil 2.1). Araştırmacıların Markowitz (1952a) tarafından önerilen O-V modelini etkin bir şekilde çözmek için algoritma geliştirme çabaları, literatürde dev bir arşiv oluşturmuştur. O-V modeli ikinci dereceden bir yapıdadır ve dolayısıyla kesin çözüm veren algoritmalar O-V modelli PO problemini çözebilir niteliktedir. Ancak, ek kısıtlar eklendiğinde ya da problem boyutu arttığında, kesin çözüm algoritmaları PO problemini çözmekte yetersiz kalmaktadırlar.

Pek çok araştırmacı, NP-zor olduğu kanıtlanan (Moral-Escudero ve diğ. 2006) PO problemini çözmek için yakınsama tekniklerini kullanmışlardır.



Şekil 2.1: Yıllara göre ortalama-varyans modelli PO problemini çözmek için yapılan yayınların dağılımları

Şekil 2.2'de PO problemi için önerilen çözüm yaklaşımlarının yıllara göre değişimi gösterilmiştir. Yıllar ilerledikçe populasyon temelli yaklaşımların, tek çözüm temelli yaklaşımlara üstünlük sağladıüı açıkça görülmektedir. Ayrıca sürü temelli algoritmaların da kullanımı yıllarla beraber artmıştır.



Şekil 2.2: PO problemi çin önerilen çözüm yaklaşımlarının yıllara göre gelişimi

Bölüm 2.1'de sürü temelli algoritmaların sınıflandırılmalrı verilmiş ve PO problemindeki uygulamaları özetlenmmiştir.

2.1 PO Modelleri

Bu bölümde Sürü Zekası (SZ) literatüründeki PO modellerinden ve bu modelleri daha gerçekçi yapmak için eklenen kısıtlardan kısaca bahsedilmiştir. Tablo 2.1 PO modellerinin bir özetini sunmaktadır.

Tablo 2.1: PO Modelleri

Model	Öneren	Yapı	Yıl
Ortalama Varyans(O-V)	Markowitz (1952b)	İkinci dereceden	1952
Çarpıklıklı Varyans (ÇV)	Samuelson (1958)	İkinci dereceden	1958
Yarı Varyans (Y-V)	Markowitz (1959)	İkinci dereceden	1959
Ortalama Mutlak Sapma (OMS)	Konno ve Yamazaki (1991b)	Lineer	1991
Riskteki Değer (RD)	Jorion (1997)	Lineer	1997
Minimaks (MM)	Young (1998)	Lineer	1998
Koşullu Riskteki Değer (KRD)	Rockafellar ve Uryasev (2000)	Lineer	2000

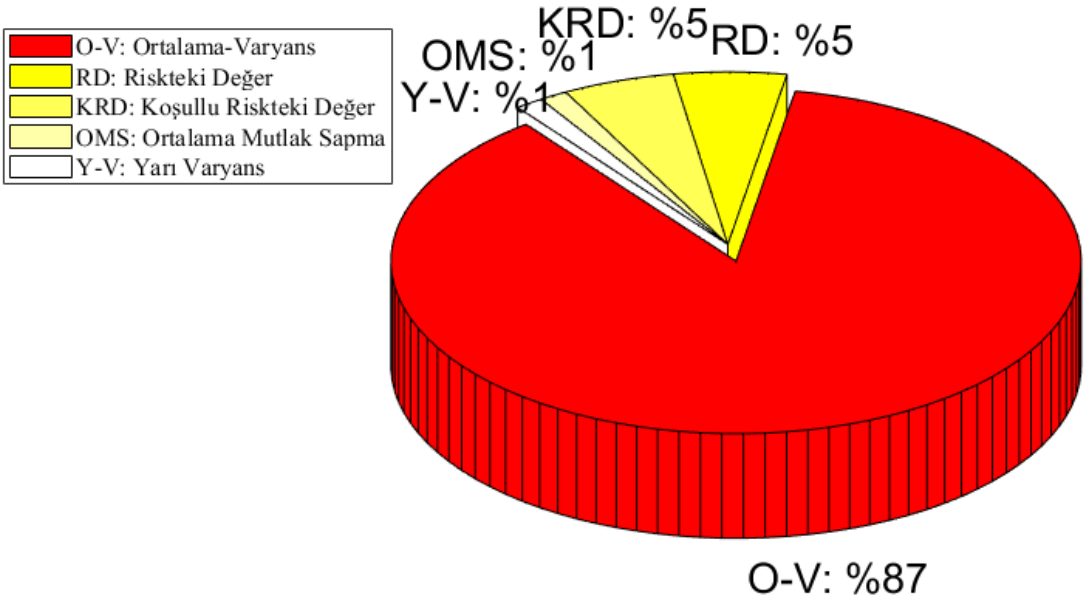
O-V modelinde portföy riski hisse senetlerinin varyansı ile ölçülür. Genelde, hisseler arasındaki kovaryans ve beklenen getiri, geçmiş veriler kullanılarak tahmin edilir. Ancak O-V yaklaşımı, asimetrik beklenen getiri dağılımı verilen portföyün yetersiz tahminine yol açabilir, çünkü M-V modeli, beklenen getirilerin simetrik normal dağılıma sahip olduğunu varsayar. Dolayısıyla Markowitz, asimetrik beklenen getirilere daha uygun olan Y-V modelini de önermiştir. Beklenen getirilerin karakterisliklerini daha başarılı yakalamak için önerilen bir diğer modelde ise O-V modelinde çarpıklık göz önünde bulundurulmuştur.

O-V modeli ikinci dereceden bir amaç fonksiyonuna ve lineer kısıtlara sahiptir. Markowitz modern portföy teorisinin babası olarak görülse de, O-V modeli yapısı nedeniyle eleştrilmeiş ve RD, KRD, OMS ve MM gibi alternatif risk ölçütleri önerilmiştir. İkinci dereceden yapıya sahip olan O-V modelinin artan hesaplama karmaşıklığı sorununu aşmak için, Konno ve Yamazaki (1991b) alternatif bir risk ölçütü kullanarak OMS modelini önermiştir. Bu modelde risk hisslerin getiri oranlarının mutlak sapmaları kullanılarak ölçülür. Bu nedenle kovaryans matrisi hesaplamak gerekli değildir, böylece daha az hesaplama maliyetine sahip ve yeni veri eklendiğinde güncellemesi çok kolaydır. Dahası, modelin risk fonksiyonu parametrik lineer programlamaya dönştürülebilir ve böylece problemi uygulamak daha kolay olur. Fakat Simaan (1997) küçük boyutlu verilerde hem O-V hem de OMS modellerinde yüksek risk toleransı olan yatırımcılar için tahmin hatasının daha yüksek olmasına rağmen, O-V modelin küçük boyutlu verilerde ve düşük risk toleransı olan yatırımcılar için daha düşük tahmin riski sağladığını belirtmiştir.

Diğer bir risk ölçütü olan MM, oyun teorisine dayanmaktadır. O-V portföy seçim kurallarının kullandığı ikinci dereceden bir yardımcı fonksiyonun mantıksal problemlerinden kaçınmak için bir risk ölçütü olarak varyanstan ziyade minimum getiriyi kullanır. MM modelde risk, bütün periyotlar boyunca minimum portföy getirisi kulanılarak ölçülür. Böylece, O-V ve OMS modellerinin aksine, MM modelinde risk ölçütü asimetriktir.

KRD, aynı zamanda Ortalama Fazla Kayıp, Ortalama Kestirme veya RD uzantısı olarak da bilinir, lineer bir model elde etmek için bir dizi senaryo ile denklem yoğunluğu fonksiyonunun yaklaşık olarak tahmin edilmesidir. Hedef zamandaki tahmin edilen kazanç ve kayıp dağılımı miktarını tanımlayan RD, bir portföyün potansiyel olarak zarar görebileceği en kötü kaybı (en düşük getiri) ölçer. Bununla birlikte, VaR'nin alt sınıfsallık ve dışbükeylik eksikliği gibi istenmeyen matematiksel özellikleri vardır (Rockafellar ve Uryasev 2000). KRD, geri dönüş ve risk analizlerinde de kullanılabilecek asimetrik bir risk ölçümünü temsil eden RD'ye dayalı yüksek kayıp riskini azaltmayı amaçlamaktadır.

Şekil 2.3 SZ literatüründeki PO modellerinin dağılmını göstermektedir. Açıkca görünmektedir ki O-V modeli diğer modellere büyük bir baskınık sağlamıştır.



Şekil 2.3: SZ literatüründeki PO modellerinin dağılımı

2.1.1 Gerçekçi Portföy Yönetimi için Ek Kısıtlar

Temel PO modellerinde özel kısıtların eksikliği, yatırımcıların gerçek hayattaki kısıtlara karşılık vermesinde sorunlar yaşatmaktadır. Kısıtsız model, yatırımın %100 oranında yapılması ve kısa satışlara izin verilmemesi dışında hiç bir kısıta sahip değildir. Bu kısıtlara ek limit kısıtı (LK)(Speranza 1996), eleman sayısı kısıtı (ESK)(Bienstock 1996), işlem maliyetleri (İM)(Magill ve Constantinides 1976) ve işlem lotları (İL)(Konno ve Yamazaki 1991a) gibi kısıtlar daha gerçekçi bir portföy yönetimi için gereklidir.

Portföydeki küçük oranlar genellikle performans üzerinde çok az etkiye sahiptir ve zayıf likiditeye sahiptir ve aracılık ücretleri veya izleme maliyetlerinin eklenmesi maliyeti olabilir. Bu nedenle, uygulamada, müşterilerin satın alma, satma veya revize etme durumlarında, minimum satın alma veya minimum işlem seviyesinden daha fazla yatırım yapmaması durumunda, bir pozisyonun tutulmasını engelleyen bir alt sınır istenebilir. Esneklik için bir üst sınır da kullanılabilir. Bu nedenle, LK matematiksel formülasyonlarda bir kısıtlama olarak eklenir.

Bir yatırımcı veya fon yöneticisinin, endeksteki tüm menkul kıymetleri satın alması mümkün değildir. Bu nedenle, portföyün daha kolay yönetimi için tüm menkul kıymetler kümesinin sadece bir alt kümesini satın alamktadırlar. Bu koşullar altında, bir portföyde tutulacak menkul kıymetlerin sayısını sınırlamak için de ESK'ye ihtiyaç duyulmaktadır.

Herhangi bir finansal piyasada yatırımcılar, alımlar, satışlar, komisyonların neden olduğu menkul kıymetlerin revize edilmesi, likidite maliyetleri, vergi veya fon kredileri için işlem masraflarını ödemekle yükümlüdür. İşlem maliyetleri, hisse senedinin toplam fiyatı ile orantılı olarak işlem gören tutarına bağlı olarak sabit veya değişken olabilir. İşlem maliyetlerinin yokluğunda, modeller gerçekçi olmaktan uzaktır çünkü işlem maliyetlerinin varlığı sadece beklenen sonucu etkilemekle kalmaz, aynı zamanda yatırımcıların davranışlarını da önemli ölçüde etkiler. Bu nedenle, işlem maliyetleriyle tüketilen para, matematiksel formülasyonlarda dikkate alınmalıdır (Ertenlice ve Kalayci 2018).

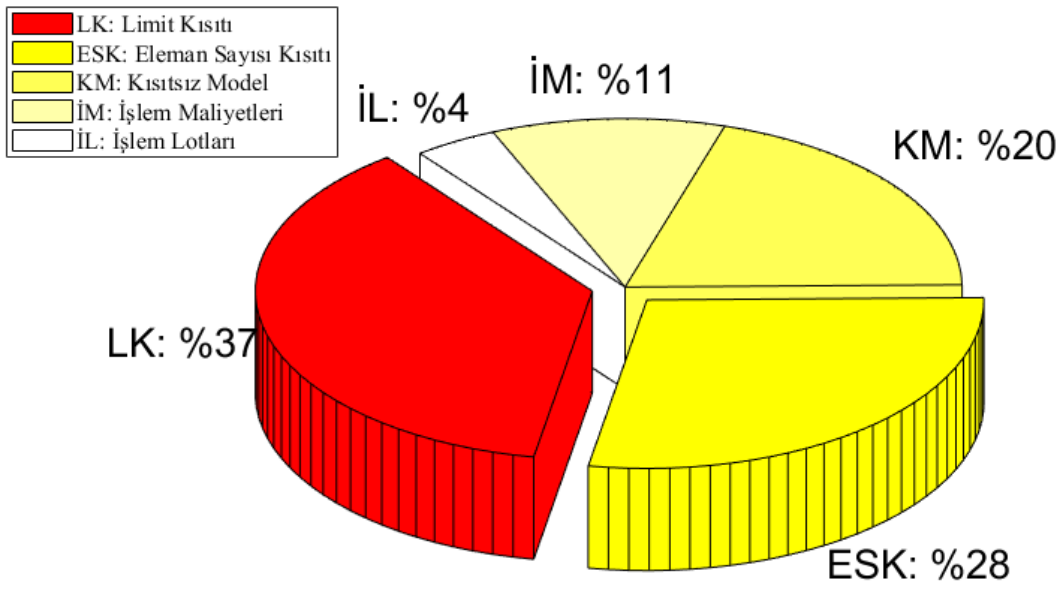
Gerçek dünyada, her bir menkul kıymet için portföyün yatırım oranı, asgari işlem lotuna sahip olduğu için ondalıklı sayılar ile temsil edilemez. Bu nedenle, sayıları en küçük / en yüksek birimin katlarına yuvarlayarak minimum / maksimum işlem birimlerini dikkate almak gerekir.

Tablo 2.2 PO problemini daha gerçekçi yapmak için kullanılan kısıtların bir özetini sunmaktadır.

Tablo 2.2: Gerçekçi portföy yönetimi için kısıtlar

Kısıtsız model (Markowitz 1952b):	- Yatırımın %100 oranında yapılmasını sağlar - Kısa satışlara izin vermez
Limit Kısıtı (LK) (Speranza 1996):	- Minimum ve maksimum yatırımı kısıtlar
Eleman Sayısı Kısıtı (ESK) (Bienstock 1996):	- Portföydeki hisse sayısını kısıtlar
İşlem Maliyetleri (İM) (Magill ve Constantinides 1976):	- Toplam getiriden işlem maliyetlerini düşer
İşlem Lotları (İL) (Konno ve Yamazaki 1991b):	- Minimum ve maksimum işlem lotlarını en yakın tam sayıya yuvarlar

Şekil 2.4'de kısıtların SZ literatüründeki dağılımı verilmiştir. Grafikte de görüldüğü üzere LK ve ESK'nın diğer kısıtlara göre bir üstünlüğü vardır.

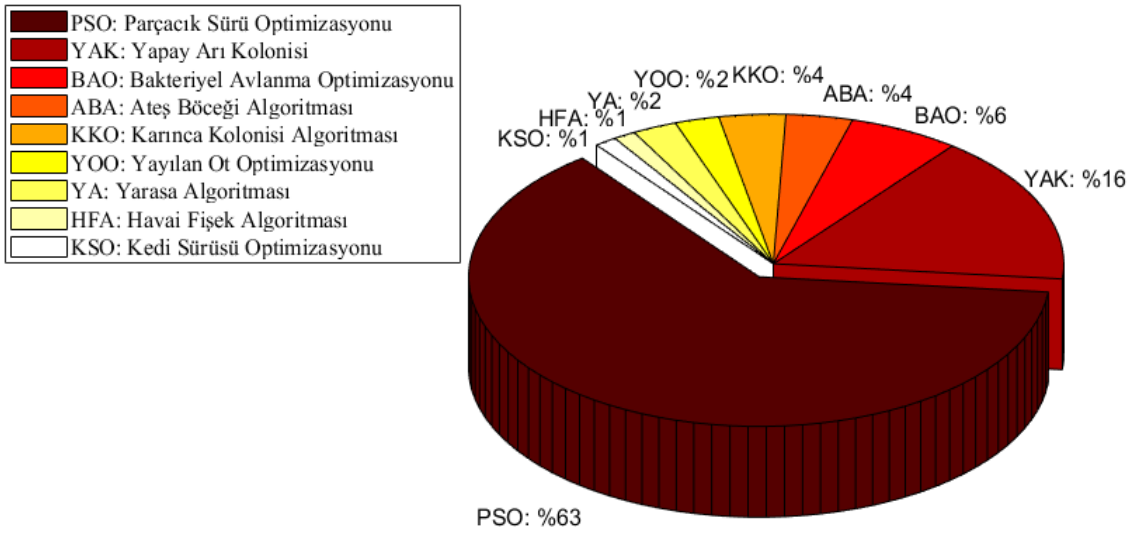


Şekil 2.4: SZ litertüründe PO için göz önünde bulundurulan kısıtlar

2.2 Sürü Temelli Algoritmalar

Sürü zekası (SZ) konsepti ilk defa (Beni ve Wang 1993) tarafından populasyondaki bireylerin kendi içlerindeki organizasyonu modelleyerek bilişimsel zekanın bir alt disiplini olarak bahsedilmiştir. Bireyler deneyimlerini paylaşabilirler, böylece bu etkileşimler populasyonun performansını arttırır.

Sürü temelli algoritmalar doğada sürü halinde yaşayan hayvanların doğal yaşamlarından esinlenilerek geliştirilmiş yakınsama teknikleridir. Kuş, balık, karınca, arı gibi sürü halinde yaşayan hayvanların doğal yaşamlarındaki davranışlarını (çoğunlukla avlanma) benzeterek, kombinasyonel problemler için hesaplamalı sistemler geliştirilmektedir. PO problemine uygulanan sürü temelli algoritmaların dağılımı Şekil 2.5'de gösterilmiştir.



Şekil 2.5: PO problemi için uygulanan sürü temelli algoritmaların dağılımı

Literatürde PO probleminin çözümü için uygulanan sürü temelli algoritmaların %63'ü Parçacık Sürü Optimizasyonu algoritamsı iken, Yapay Arı Kolonisi algoritması %16, Bakteriyel Avlanma Optimizasyonu algoritması % 6, Ateş Böceği Algoritması %4, Karınca Kolonisi Optimizasyonu algoritması %4, Yayılan Ot Optimizasyonu algoritması %2, Yarasa Algoritması %2, Havai Fişek Algoritması %1 ve Kedi Sürüsü Optimizasyonu algoritması ise %1 oranında uygulanmıştır.

Genelde metasezgisel algoritmaların en belirgin karakteristik özellikleri sömürme ve arama mekanızmalarıdır. Sömürme mekanızması arama sonucunda bulunan en iyi çözümün komşuluklarında dolaşarak daha iyi bir çözüm elde etmeyi amaçlarken, arama mekanızması bulunan çözümden daha farklı bir çözüme ulaşmayı ve lokal optimumdan kaçmayı amaçlamaktadır.

PO problemi için uygulanan sürü temelli algoritmalar, Tablo 2.3'de özetlenmiş ve esin kaynakları, orjinal uygulama alanlar, sömürme ve arama mekanızmalarına göre sınıflandırılmıştır.

Tablo 2.3: PO için uygulanan sürü temelli algoritmalar

Algoritma	Öneren	Yıl	Esin kaynağı	Orjinal uygulama alanı	Sömürma mekanızması	Arama mekanızması
PSO	Kennedy ve Eberhart (1995)	1995	Doğadakı balık ve kuş topluluklarının sürüsel davranışları	Devamlı alan	Az hız oranları	Çok hız oranları
KKO	Dorigo ve diğ. (1996a)	1996	Karıncaların avlanma davranışları	Kesikli alan	Sezgisel bilgiye göre çözümlerin yapısı	Feremon bilgisine göre olaslıkı seçim prosedürü
BAO	Passino (2002a)	2002	Bakteri sürüsü ve avlanma davanışları	Devamlı alan	Kemotaksis ve reprodüksiyon adımları	Eliminasyon-dispersiyon adımı
YAK	Karaboga (2005b)	2005	Arıların avlanma davranışları	Devamlı alan	İşçi ve gözcü arıların komşuluk arama mekanizması	Kaşif arının rasgele arama mekanizması
KSO	Chu ve diğ. (2006)	2006	Kedilerin arama ve izleme davranışları	Devamlı alan	İzleme modu	Arama modu
ABA	Yang (2010b)	2009	Tropik ateş böceklerinin palırtı desenleri ve davranışları	Devamlı alan	Çekiciliğe göre ateşböceği hareketi	En iyi ateşböceğinin rasgele hareketi
YOO	Karimkashi ve Kishk (2010)	2010	Doğada istilacı yabani otların kolonileşmesi	Devamlı alan	Küçük standart sapma	Büyük standart sapma
YA	Yang (2010a)	2010	Mikro yarasaların ekolokasyon davranışı	Devamlı alan	Ses ve nabız hızı tarafından kontrol edilen yoğun yerel arama	Frekans ayarlama
HFA	Tan ve Zhu (2010)	2010	Havai fişek patlaması gözlemleri	Devamlı alan	Küçük patlama genliği	Büyük patlama genliği

2.2.1 Parçacık Sürü Optimizasyonu

PSO tekniği, ilk defa (Kennedy ve Eberhart 1995) tarafından doğadaki kuş ve balık sürülerinin grup içindeki bilgi paylaşımı ve koordineli hareketlerinden esinlenilerek önerilen bir metasezgisel algoritmadır. Yiyecek ararken, parçacık olarak adlandırılan her bir üye hem kendi tecrübesinin hem de sürüdeki diğer parçacıkların tecrübesini kullanarak, pozisyonunu ve hızını ayarlar. PSO tekniğinde, bir parçacık çok boyutlu çözüm uzayında bir pozisyondan diğerine hareket ederek bir optimal çözüme yaklaşır. Çözüm uzayındaki bir pozisyon problemin bir çözümüne karşılık gelir. Genelde her bir parçacık performansını kendi hafızası ve grubun hafızasına bağlı olarak belirler. PSO tekniğinin temel adımları Tablo 2.4'de verilmiştir.

Tablo 2.4: PSO tekniğinin teme adımları

Adım	Açıklama
1	Başlangıç çözümü yarat
2	Bireylerin uygunluklarını hesapla
3	Her bir birey için bireysel en iyi pozisyonları ayarla
4	Populasyondaki en iyi birey için global en iyi pozisyonu ayarla
5	Her bir birey için hızı güncelle
6	Her bir birey için pozisyou güncelle
7	Eğer durdurma kriteri ile karşılaşılmadıysa, 2. Adıma git

PO problemi literatüründe en çok uygulanan sürü temelli yaklaşım PSO'dur. PSO tekniği PO problemine ilk defa Chen ve diğ. (2006) tarafından O-V modeli ve LK, İM ek kısıtları kullanılarak uygulanmış ve performansı Çin Borsasında onaylanmıştır. Cura (2009a) PSO tekniğini Genetik Algoritma (GA), Benzetilmiş Tavlama (BT) ve Tabu Araması (TA) teknikleri ile kıyaslamış ve hiç bir tekniğin üstünlük sağlayamadığını belirtmiştir. Niu ve diğ. (2009) doğadaki ortak yaşam ekosistem konseptinden etkilenerek bir simbiyotik çok-sürülü PSO tekniği önermiştir. Yazarlar çok sürülü bir populasyon şeması kullanmışlar ve her bir alt sürü diğer alt sürülerin deneyimlerinden etkilenmektedir. Chen ve Zhang (2010) gelişmiş bir hız hesaplama prosedürü ve ceza fonksiyonu kullanarak bir PSO tekniği geliştirmişlerdir. Zhu ve diğ. (2010) PSO ve KKO tekniklerini kıyaslamış ve KKO tekniğinin küçük ve büyük ölçekli portföylerde üstünlük kurduğunu fakat orta ölçekli portföylerde ise PSO tekniğinin üstün olduğunu belirtmiştir. Zhu ve diğ. (2011) PSO tekniğini GA ile kıyaslamış ve PSO'nun performansını arttırmak için hibrid tekniklerin kullanılmasını önermiştir. Sun ve diğ. (2011) kesikli PSO (KPSO) tekniğini geliştirmiş ve standart

PSO, GA ve LOQO ve CPLEX gibi çözücülerle kıyaslamıştır. KPSO tekniği yakınsama oranları ve hesaplama süreleri bazında üstünlük sağlamıştır. Golmakani ve Fazel (2011) de PSO algoritmasını GA ve gözlemlenen en iyi sonuçlarla kıyaslamıştır. Deng ve diğ. (2012a) literatürdeki diğer PSO algoritmalarına üstünlük sağlayan bir PSO algoritması geliştirimiştir. Corazza ve diğ. (2013) algoritmanın performansını arttırmak için ceza fonksiyonu eklemiştir. Dominant-olmayan sırlamalı çok amaçlı PSO (DSÇAPSO) tekniği Mishra ve diğ. (2014b) tarafından önerilmiş, LK ve ESK portföy optimizasyonu problemini çözmek için uygulanmıştır. Önerilen algoritmanın performansı GA, TA, BT ve PSO gibi tek amaçlı evrimsel algoritmalarla kıyaslanmış ve DSÇAPSO tekniği daha üstün bir performans sergilemiştir. Yin ve diğ. (2015) heterojen çoklu populasyon PSO (HÇPPSO) tekiniğini önermiş ve LK ve ESK ek kısıtları ile O-V modele uygulamıştır. Diğer klasik PSO teknikleri ile kıyaslandığında, HÇPPSO özellikle yüksek boyutlu problemlerde daha etkindir.

PSO tekniğinde, bir parçacık bir pozisyondan, kendi pozisyonuna ve populasyondaki en iyi parçacığın pozisyonuna bağlı olarak, diğer bir pozisyona hareket eder. Araştırmacıların pek çoğu Denklem 2.1'de verilen orjinal förmüle göre parçacıkların hızlarını günceller:

$$\overrightarrow{v_{yeni}} = w\overrightarrow{v_{eski}} + c_1 \times \vec{U} \otimes \left(\overrightarrow{p_b} - \overrightarrow{x_g}\right) + c_2 \times \vec{U} \otimes \left(\overrightarrow{p_g} - \overrightarrow{x_g}\right) \quad (2.1)$$

burada w global ve lokal tecrübelerin dengesini kotrol eden atalet ağırlığı, $\overrightarrow{v_{eski}}$ parçacık vektörünün bir önceki hızını, $\vec{U}$ [0,1] aralığında rasgele üretilen düzenli dağılıma sahip sayılar vektörünü temsil etmektedirler. $\overrightarrow{x_g}$ parçacığın güncel pozisyonunu, p_b ve p_g sırası ile bireysel ve global en iyi pozisyonları temsil etmektedirler. Bek çok araştırmacı hızı güncellemek için Denklem 2.1'i kullansa da, bazı araştırmacılar farklı metodlara başvurmuşlardır. Xu ve Chen (2006) hız limitli PSO (HLPSO) algoritmasını önermişler ve HLPSO algoritmasının daha az iterasyonda optimal çözüme ulaşarak, standart PSO'ya kıyasla daha iyi bir yakınsama etkinliğinin ve kesinliğinin olduğunu belirtmişlerdir. Koshino ve diğ. (2007) Daralma Faktor Yaklaşımı isimle yeni bir parametreyi hız güncelleme denklemine eklemişlerdir. Yaakob ve Watada (2010) parçacık hızını kontrol altına almak için Hız Kontrollü PSO tekniğini önermişlerdir.

Tablo 2.5'de PO problemini çözmek için geliştirilen bazı PSO tekinikleri, parçacık sayısı, parametre değerleri ve literatüre yaptığı katkı göz önünde bulundurularak özetlenmiştir.

Tablo 2.5: PO problemi için geliştirilmiş bazı PSO algoritmaları

Yıl	Yayın	Parçacık sayısı	Parametreler	Literatüre katkısı
2006	Xu ve Chen (2006)	20	$c_1 = c_2 = 1.494$ $w = 0.9$ $v_{max} = 0.4$ $v_{min} = -0.4$	Arama işlemi sırasında hız sınırlayıcı bir düzeni benimsenmiştir
2008	Liu ve Wang (2008)	20	belirtilmemiş	Dominant olmayan çözümleri depolamak için bir elitist arşivleme programı benimsenmiştir
2009	Wang ve diğ. (2009)	20	$c_1 = c_2 = 1.4962$ $w = 0.72984$	Sürünün çeşitliliği artan parçacıklar arası benzersizlikler tarafından dolaylı olarak korunur ve hız güncelleme maliyeti düşer. Böylece, algoritma daha hızlı yakınsar
2009	Niu ve diğ. (2009)	20	$c_1 = c_2 = 1.367$ $w = 0.9$ $w_b = 0.9, w_{min} = 0.4$	Çoklu sürü şeması
2010	Yaakob ve Watada (2010)	20	$c_1 = c_2 = 2$ $w_{begin} = 0.9$ $w_{end} = 0.4$ $v_{max} = 0.4$ $v_{min} = -0.4$	Populasyonun genetik karakterisliğini çeşitlendiren bir mutasyon operatörü eklenmiştir.
2011	Mozafari ve diğ. (2011)	50	$c_1 = c_2 = 2$ $w_{begin} = 1.5$ $w_{end} = 0.2\ or\ 0.1$	BT temelli bir lokal arama prosedürü ile lokal optimalden kaçma ve yakınsama kesinliği arttırılmıştır
2012	Deng ve diğ. (2012b)	100	$c_1^{begin} = c_1^{begin} = 2.5$ $c_1^{end} = c_1^{end} = 0.5$ $w_{begin} = 0.9$ $w_{end} = 0.4$	Başlangıç adımlarında arama mekanızması daha etkinken son adımlarda ise sömürme mekanızması daha etkindir
2013	Corazza ve diğ. (2013)	200	$w_{begin} = 0.9$ $w_{end} = 0.4$ $c_1 = c_2$ $=$ rastgele~[0,1.85]	Kısıtlayıcı olmayan kesin ceza yöntemi kısıtları sağlamak için kullanılmıştır
2015	Yin ve diğ. (2015)	100	belirtilmemiş	Belirli göç kurallarına göre arama ve sömürüyü koordine etmek için heterojen bir çoklu nüfus stratejisi kullanılmaktadır

$\boldsymbol{c_1}$: sosyal öğrenme faktörü; $\boldsymbol{c_2}$: bilişsel öğrenme faktörü; $\boldsymbol{w}$: atalet ağırlığı; $\boldsymbol{v_{max}}$: parçacık hızı için bir üst limit; $\boldsymbol{v_{min}}$: parçacık hızı için bir alt limit $\boldsymbol{w_{begin}}$: atalet momenti için başlangıç değeri; $\boldsymbol{w_{end}}$: atalet momenti için bitiş değeri; $\boldsymbol{c^{begin}}$: sosyal yada bilişsel öğrenme faktörü için başlangıç değeri; $\boldsymbol{c^{end}}$: sosyal yada bilişsel öğrenme faktörü için bitiş değeri.

2.2.2 Yapay Arı Kolonisi

Yapay arı kolonisi (YAK) algoritması Karaboga (2005a) tarafından arı sürülerinin kovana besin getirme için yaptıkaları hareketlerden esinlenilerek önerilmiştir. Arılar doğada koloniler halinde yaşayan sosyal calılardır. Kolonide üç tip arı çeşidi vardır: işçi arılar, gözcü arılar ve kaşif arılar. İşçi arılar besin kaynaklarını sömürür ve nektar miktarlarını bir dans aracılığı ile koloniyle paylaşır. Gözcü arılar bu dansı izler ve nektar miktarlarına göre hangi yöndeki besin kaynaklarının sömürülmesine karar verir. Bilinen bütün besin kaynaklarındaki nektarlar bittiği zaman, kaşif arılar yeni besin kaynaklari ararlar. Kaşif arıların işi rasgele yeni besin kaynakları keşfetmektir. Besin kaynağının pozisyonu YAK algoritmasında bir çözüme ve nektar miktarı ise o çözümün kalitesine karşılık gelir. YAK algoritmasının temel adımları Tablo 2.6'da verilmiştir.

Tablo 2.6: YAK algoritmasının temel adımları

Adım	Açıklama
1	Başlangıç populasyonu yarat
2	Populasyonun uygunluğunu hesapla
3	Işçi arı aşaması
4	Gözcü arı aşaması
5	Kaşif arı aşaması
6	En iyi çözümü kaydet
7	Eğer durdurma kriteri ile karşılaşılmamışsa, 3. Adıma git

YAK algoritması PO problemine ilk defa Hong-Mei ve diğ. (2010) tarafından uygulanmış ve GA ile karşılaştırılmıştır. Test sonuçları YAK algoritmasının yakınsama hızı ve çözüm kalitesi olarak daha iyi bir performans sergilediğini göstermiştir. Chen ve diğ. (2012) YAK algoritmasını PO problemine uygulamış ve literatürdeki değişken komşuluk arama (DKA), BT ve TA algoritmaları ile kıyaslamıştır. Yazarlar YAK algoritmasının PO probleminin çöziminde diğer sezgisel algoritmalardan daha iyi çözümler elde ettiğini belirtmiştir. Wang ve diğ. (2012) de YAK algoritmasını GA, TA, BT (Chang ve diğ. 2000) ve PSO (Cura 2009a) teknikleriyle kıyaslamıı ve YAK algoritmasını diğer tekniklerden daha iyi çözümler elde ettiğini belirtmişlerdir. Chen ve diğ. (2013) kaşif arıların standart bir başlangıç populasyonundan üretilmesinin güçlü bir çeşitlilik sağlasa da, çözüm kalitesini düşürdüğünü belirtmiştir. Dolayısıyla, yazarlar sömürme ve aram mekanızmaları arasında bir denge sağlayan gelişmiş bir YAK algoritması önermişlerdir.

Karşılaştırmalı sonuçlar göstermiştir ki, geliştirilen YAK algoritması çeşitlilik, yakınsama ve etkinlik açısından DKA, BT, TA ve standart YAK algoritmalarına üstünlük sağlamıştır. Bacanin ve diğ. (2014) kısıtlı PO problemine etkin bir kısıt elleçleme metodlu YAK algoritması uygulamışlardır. Yazarlar YAK algoritmasını GA ve ABA teknikleri ile kıyaslamış ve YAK algoritmasının PO probleminin etkin bir şekilde çözme potansiyeline dikkat çekmişlerdir. Suthiwong ve Sodanil (2016) YAK algoritmasının işçi arı aşamsındaki sömürme mekanizmasının performansını geliştirmek için, PSO algoritmasın aramaya rehberlik etmek için en iyi çözümün bilgisinin paylaşılması avantajından esinlenerek bir YAK algoritması önermiştir. Ge (2015) başka bir umut verici starndart YAK algoritmasınında üstün bir YAK algoritması önermiştir. Kumar ve Mishra (2017) ESK ve yatırım limiti kıstılı PO problemi için kovaryans güdümlü bir YAK algoritması önermiş ve OR-library (Beasley 1990) veri setinde test etmişlerdir. Kalayci ve diğ. (2017) YAK temelli yeni bir metodoloji önermişler ve algoritmanın üstünlüğünü litertürdeki diğer OR-library veri seti kullanan yayınlarla kıyaslayarak onaylamışlardır. Önerilen algoritmada LK ve ESK kısıtlarını elleçlemek için uygunluğa zorlayan ve uygunsuzluğa izin veren bir prosedür geliştirmiştir. Tablo 2.7'de PO problemine uygulanan bazı YAK algoritmaları özetlenmiştir.

Tablo 2.7: PO problemini çözmek için geliştirilen bazı YAK algoritmaları

Yıl	Yayın	Besin kaynağı sayısı	Parametreler	Literature katkısı
2010	Hong-Mei ve diğ. (2010)	125	$EB = {}^{SN}/_{2}$ $OB = {}^{SN}/_{2}$ $SB = 1$	YAK algoritması PO problemine ilk kez uygulandı
2012-2013	Chen ve diğ. (2012) Chen ve diğ. (2013)	$N/4$	$EB = SN$ $SB = SN$ $limit = 50$	Gerçek ve tam sayılı hibit bir kodlama kullanıldı
2015	Ge (2015)	20	$limit = 100$	PSO'dan esinlenilerek en iyi çözümün tecrübesi sömürme mekanızmasının performansını arrtırmak için kullanıldı
2017	Kumar ve Mishra (2017)	$10N$	$EB = {}^{SN}/_{2}$ $SB = {}^{SN}/_{2}$ $limit = 10N$	Kovaryans presibi YAK algoritması ile birleştirilerek daha hızlı ve kesin bir yakınsama sağlandı
2017	Kalayci ve diğ. (2017)	N	$EB = SN$ $OB = {}^{SN}/_{2}$ $limit = SN \times N$	Kısıtları elleçlemek için uygunluğa zorlayan ve uygunsuzluğa izin veren yen bir mekanizma geliştirildi

EB: işçi arı sayısı, *OB*: gözcü arı sayısı, *SB*: kaşif arı sayısı, *SN*: besin kaynağı sayısı, *N*: problem boyutu; *limit*: kaşif arıların yeni besi kaynakları keşfetmesi için salınma zamanına karar veren parametre

2.2.3 Diğer Sürü Temelli Algoritmalar

Araştırmacılar sadece balık, kuş ve arı sürülerinden değil, karıncalardan, bakterilerden, yarasalardan ve ateş böceklerinden de etkilenmişlerdir. Bu bölümde karınca kolonisi optimizasyonu (KKO), bakteriyel avlanma optimizasyonu (BAO) ve ateş böceği algoritması (ABA) gibi PO problemine uygulanan sürü temelli algortimalar tartışılmıştır.

Karıncaların avlanma stratejilerinden esinlenilen, karınca sistemi ilk defa (Dorigo ve diğ. 1996b) tarafından bahsedilmiştir. Karıncalar, doğada yiyecek ararken, balıkların, kuşların ve arıların aksine dolaylı olarak iletişim kurarlar. Karıncalar yolda ilerlerken arkalarında feromon adı verilen bir kimyasal madde bırakırlar. Karıncalar yollardaki feromonu algılayabilirler ve feromon miktarının yoğun olduğu yoldan gitme eğilimindedirler. Karıncalar aynı yolda yürüdükçe yoldaki feromon miktarı artacağından, o yol diğer karıncalara daha cazip gelecektir. Eğer yolda bir engel var ise karıncalar engelin etrafından rasgele bir yol seçerek geçeceklerdir, fakat er yada geç karıncalar feromon yoğunluğuna göre en kısa yolu bulacaklardır (Bissiri ve diğ. 2002). KKO algoritması ilk defa gezgin satıcı probleminin çözümü için önerilmiştir (Dorigo ve Gambardella 1997). Bu yüzden, KKO algoritmasının yapısı temel olarak kesikle problemler için uygundur. PO probleminin yapısından dolayı, bu problemi çözmek için KKO algoritması sürekli yapıya adapte edilmelidir. Sürekli alanda, feromon matrisinin gösterimi, reel sayılar kullanıldığı için sonsuz olasılıktan dolayı, imkansızdır. (Socha ve Dorigo 2008) $KKO_{\mathbb{R}}$ olarak adlandırılan, KKO algoritmasının sürekli versiyonunu önermişlerdir. (Khalidji ve diğ. 2009) dinamik ağırlıklandırılmış $DAKKO_{\mathbb{R}}$ isimli iki amaçlı sürekli bir KKO algoritması önermişler ve domine edilmemiş sıralamalı genetik algoritma ile kıyaslamışlardır. (Deng ve Lin 2010) $KKO_{\mathbb{R}}$ algoritmasını PSO ile kıyaslamışlar ve sonuçların $KKO_{\mathbb{R}}$ algoritmasının özellikle düşük risk seviyelerinde daha iyi olduğunu belirtmişlerdir. (Zhu ve diğ. 2010) $KKO_{\mathbb{R}}$ ve PSO algoritmalarını kıyaslayan bir çalışma yayınlamışlardır. Yazarlar, küçük ve büyük boyutlu portföylerde $KKO_{\mathbb{R}}$ algoritmasının daha iyi sonuçlar veriğini belirtirken, orta boyutlu portföylerde ise PSO algoritmasının daha iyi performans sergilediğini belirtmişlerdir.

Bakteriyel avlanma optimizasyonu (BAO) algoritması bakterilerin avlanmak ve değişik ortamlarda hayatta kalmak için sergilediği organize hareketlerinden esinlenilerek önerilmiştir (Passino 2002b). Bir BAO algoritması problemdeki çözümleri temsil eden birkaç bakteri ve üç proses: kemotaksi, üreme ve eleme-dispersiyon içerir. (Kao ve Cheng 2013) bir BAO algoritması önermişler ve GA, TA, BT (Chang ve diğ. 2000) ve PSO (Cura 2009a) algoritmaları ile kıyaslamışlardır. Yazarlar BAO algoritmasının etkin sınırı bulmak için daha iyi bir seçim olduğunu raporlamışlardır. (Tan ve diğ. 2013), global arama ve yerel aramada doğru dengeyi korumak için tüm kemotaktik harekete doğrusal olarak azalan bir mekanizma kullanan değiştirilmiş bir BAO algoritması önerdi. Yazarlar, deneysel sonuçların yapılan değişikliğin BAO algoritmasını yüksek kaliteli çözümler elde etme açısından daha üstün yaptığını onalyladığını belitrmişlerdir. BAO'nun performansını daha da iyileştirmek için, (Niu ve diğ. 2012), BAO'nun GA, PSO ve standart BAO'dan daha üstün, lineer azalan kemotaksiye sahip bir BAO algoritmasını önermişlerdir.

Ateş böceklerinin ışıldama desenlerinden ve hareketlerindend esinlenilen ateş böceği algoritması da (ABA) (Yang 2009) PO problemini çözmek için kullanılmıştır. (Bacanin ve Tuba 2014) YAK algoritasındaki $limit$ parametresini gereksiz aşırı arama prosesini önlemek için ABA algoritmasına adapte etmiştir. Yazarlar geliştirilen algoritmayı GA, TA, BT (Chang ve diğ. 2000) ve PSO (Cura 2009a) algoritmaları ile kıyaslamış ve ABA algoritmasının daha iyi performans sergilediğini belirtmişlerdir. (Tuba ve Bacanin 2014a) son iterasyonlarda daha yoğun sömürme mekanızmalı bir ABA algortiması geliştrimiş ve GA, TA, BT (Chang ve diğ. 2000) algoritmaları ile kıyaslayarak, geliştirlen algoritmanın standart ABA algoritmasından tüm durumlarda daha iyi olduğunu ve diğer algoritmalardan çoğu durumda daha iyi olduğunu belirtmişlerdir.

3. YÖNTEM

Bu bölümde portöy optimizasyonu probleminin matematiksel modeli, algoritmanın test edilmesi için kullanılan veri setleri ve hata ölçütleri, stadart yapay arı kolonisi ve geliştirlmiş yapay arı kolonisi algoritmaları verilmiştir.

3.1 PO Probleminin Matematiksel Modeli

PO problemi ikinci dereceden bir amaç fonksiyonuna ve lineer kısıtlara sahip iki çakışan amaçlı bir problemdir. Bu iki çakışan amaç; risk minimizasyonu ve kar maksimizasyonu aynı anda başaralamayacağı için pareto optimal olarak çözülür. Pareto optimal çözümler kümesi PO probleminde etkin sınırı belirler. Eğer bir risk değerinde bir portföyden daha fazla getiri sağlayabilecek bir portföy yok ise o portföy etkin sınırdadır. (Markowitz 1952a) tarafından önerilen Ortalama-Varyans modeli aşağıda verilmiştir:

Parametreler:

N Uygun varlıkların sayısı

μ_i i. varlığın beklenen getirisi

σ_{ij} i. ve j. varlıkların arasındaki kovaryans değeri

R^* İstenen düzeydeki beklenen getiri

Karar Değişkenleri:

w_i i. varlığın oranı

Amaç Fonksiyonu:
$$min \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} \tag{3.1}$$

Kısıtlar:
$$\sum_{i=1}^{N} w_i \mu_i = R^* \tag{3.2}$$

$$\sum_{i=1}^{N} w_i = 1 \tag{3.3}$$

$$0 \leq w_i \leq 1, \qquad i = 1, \dots, N \tag{3.4}$$

Denklem (3.1), portföydeki toplam riski minimize ederken, (3.2) ise portföyün R^* beklenen getirisini sağlamasını garanti eder. Denklem (3.3) portföyde kullanılan varlıkların oranları toplamının 1 olmasını sağlar. Bu formülasyon ile herhangi bir veri seti için optimal çözümün hesaplanması pratikte mümkün olabilmektedir. Bu modeli çözerek, R^*hedeflenen getirisinin değişen değerlerine göre kesiksiz artan bir eğri ile etkin sınır bulunabilir. Etkin sınırı bir λ ağırlıklandırma parametresi ile takip etmek mümkündür (Chang ve diğ. 2000). Denklem (3.5)' deki λ parametresi 0 olduğunda, riske bakılmaksızın beklenen getiri maksimum olmakta ve optimal çözüm en yüksek getiri veren bir adet varlıktan oluşmaktadır. λ parametresi 1 olduğunda ise, beklenen getiriye bakılmaksızın risk minimum olmakta ve optimal çözüm birden fazla varlıktan oluşabilmektedir. λ parametresinin $0 < \lambda < 1$ değerleri için, $\lambda = 0$ ve $\lambda = 1$ sınır değerleri arasında getiri ile risk arasında ödünleşerek etkin sınır üzerindeki noktaları oluşturur.

Amaç Fonksiyonu:
$$min \quad \lambda \left[\sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} \right] - (1-\lambda) \left[\sum_{i=1}^{N} w_i \mu_i \right] \tag{3.5}$$

Kısıtlar:
$$\sum_{i=1}^{N} w_i = 1 \tag{3.6}$$

$$0 \leq w_i \leq 1 \qquad i = 1, \dots, N \tag{3.7}$$

Denklem (3.5) - (3.7) ile verilen formülasyona ESK ve LK kısıtları eklendiğinde eleman sayısı kısıtlı portföy optimizasyonu problemi elde edilmektedir. Eklenen kısıtlar ve karar değişkeni aşağıdaki karma tamsayılı doğrusal olmayan programlama (KTDOP) modeline ilişkin formülasyonla gösterilmiştir:

Modele eklenen parametreler:

K Portföydeki istenen varlık sayısı

ε_i i. varlığın portföydeki minimum ağırlığı

δ_i i. varlığın portföydeki maksimum ağırlığı

Modele eklenen karar değişkeni:

$$z_i = \begin{cases} 1 & \text{Eğer } i. \text{ varlık portföyde ise} \\ 0 & \text{Aksi halde} \end{cases}$$

$$\underline{\text{Amaç Fonksiyonu:}} \quad min \quad \lambda \left[\sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} \right] - (1-\lambda) \left[\sum_{i=1}^{N} w_i \mu_i \right] \tag{3.8}$$

$$\underline{\text{Kısıtlar:}} \quad \sum_{i=1}^{N} w_i = 1 \tag{3.9}$$

$$\sum_{i=1}^{N} z_i = K \tag{3.10}$$

$$\varepsilon_i z_i \leq w_i \leq \delta_i z_i \qquad i = 1, \dots, N \tag{3.11}$$

$$z_i \in (0,1) \qquad i = 1, \dots, N \tag{3.12}$$

$$0 \leq w_i \leq 1, \qquad i = 1, \dots, N \tag{3.13}$$

$$0 \leq \varepsilon_i \leq \delta_i \leq 1, \qquad i = 1, \dots, N \tag{3.14}$$

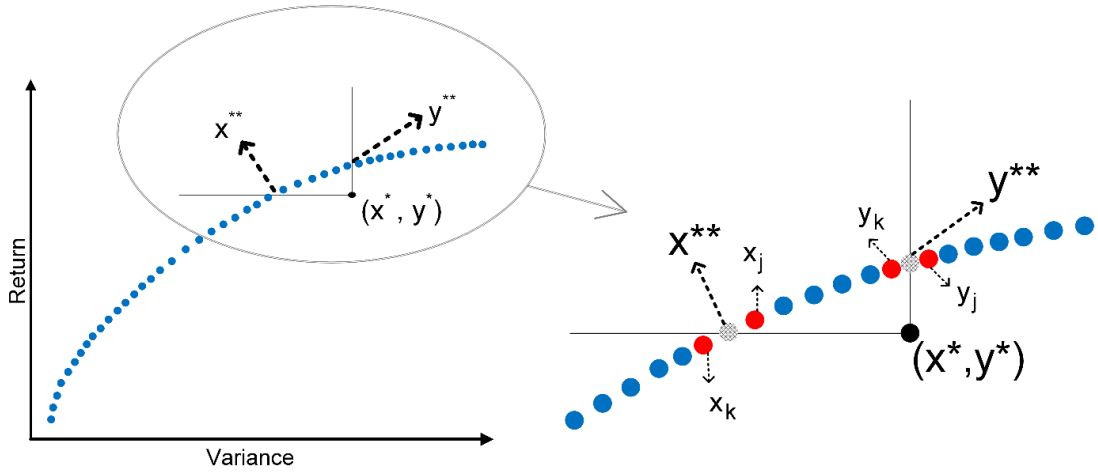
Denklem (3.8) ve (3.9) kısıtsız modelden eklenmiş olup, denklem (3.10), portföyde toplam K sayıda varlığın bulunmasını garanti ederken, denklem (3.11) ise portföye alınan varlığın ağırlığının belirlenen minimum ve maksimum değerler arasında olmasını sağlar. Denklem (3.12) de karar değişkenin 0 veya 1 olmasını sağlayan bütünlük kısıtıdır. Denklem (3.13) ile ifade edilen kısıt bir varlığın ağırlığının 0 ile 1 arasında olmasını sağlarken, denklem (3. 14) bir varlığın alabileceği minimum ve maksimum ağırlık kısıtlarını sağlamaktadır.

3.2 Veri Setleri ve Hata Ölçütleri

Literatürde yaygın olarak karşılaştırma amaçlı kullanılan veri setleri OR-Library (http://people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html) web sitesinden indirilmiştir. Bu veri setleri Mart 1992 ile Eylül 1997 tarihleri arasındaki sırasıyla, 31, 85, 89, 98 ve 225 adet hisse bulunduran Hang Seng, DAX 100, FTSE 100, S&P 100 ve Nikkei 225 günlük kapanış fiyatları baz alınarak oluşturulmuştur. OR-Library veri setine ek olarak Kalayci ve diğ. (2017) tarafından literatüre kazandırılan BİST30 ve BİST100 veri setleri de geliştirilen algoritmanın performansını ölçmek için kullanılmıştır.

Literatürde algoritmaların oluşturduğu kısıtlı etkin sınır ile standart kısıtsız etkin sınır arasındaki hataları hesaplamak için yaygın olarak kullanılan performans ölçütleri bu tezde de kullanılmıştır. Bu ölçütler: Yüzde Hatasının Ortalaması (YHO), Yüzde Hatasının Minimumu (YHMİN), Yüzde Hatasının Maksimumu (YHMAKS), Yüzde Hatasının Medyanı (YHMED), Getiri Hatasının Varyansı 1 (GHV-I), Ortalama Getiri Hatasını 1 (OGH-1), Ortalama Öklid Uzaklığı (OÖU), Getiri Hatasının Varyansı 2 (GHV-II) ve Ortalama Getiri Hatasını 2 (OGH-II).

(x_i, y_i), $(i = 1, \dots, 2000)$), standart etkin sınır üzerindeki risk ve getiri değerlerini temsil etmekte ve (x^*, y^*), $(j = 1, \dots, E)$algoritmalar tarafından üretilen etkin sınır üzerindeki risk ve getiri değerlerini temsil etmektedir (Şekil 3.6). Basit interpolasyon kullanılarak, geliştirilen algoritma tarafından elde edilen noktaların, standart etkin sınıra göre olan hataları hesaplanabilir (Chang ve diğ. 2000).



Şekil 3.6:Temsili bir hata hesaplama

$$\varphi_j = 100\left|{(y^* - y^{**})}/{y^{**}}\right| \quad \text{/*getirin için yüzde hata*/} \quad (3.15)$$

$$\omega_j = 100\left|{(x^* - x^{**})}/{x^{**}}\right| \quad \text{/*varyans için yüzde hata*/} \quad (3.16)$$

$$x^{**} = x_k + (x_j - x_k)\left[\frac{(y^* - y_k)}{(y_j - y_k)}\right] \quad \text{/*}(x^*, y^*)\text{ noktasının standart etkin sınır üzerine yatay izdüşümü*/} \quad (3.17)$$

$$y^{**} = y_k + (y_j - y_k)\left[\frac{(x^* - x_k)}{(x_j - x_k)}\right] \quad \text{/*}(x^*, y^*)\text{ noktasının standart etkin sınır üzerine dikey izdüşümü*/} \quad (3.18)$$

$$y_j = \min[y_i | y_i \geq y^*] \quad (3.19)$$

$$y_k = \max[y_i | y_i \leq y^*] \quad (3.20)$$

$$x_j = \min[x_i | x_i \geq x^*] \quad (3.21)$$

$$x_k = \max[x_i | x_i \leq x^*] \quad (3.22)$$

Yukarıda verilen formülasyonları kullanarak elde edilen performans ölçütleri aşağıda verilmiştir:

YHO, YHMED, YHMİN, YHMAKS (Chang ve diğ. 2000):

$$\text{YHO}=\frac{\sum_{j=1}^{E} \min(\varphi_j,\psi_j)}{E} \quad (3.23)$$

$$\psi_j = 100 \left| \frac{(\sqrt{x^{**}}-\sqrt{x^*})}{x^{**}} \right|$$

$$\text{YHMED} = \min\{medyan\{\varphi_1, \varphi_2, \dots, \varphi_E\} \ , \ medyan\{\psi_1, \psi_2, \dots, \psi_E\}\} \quad (3.24)$$

$$\text{YHMİN} = \min\{\min\{\varphi_1, \varphi_2, \dots, \varphi_E\} \ , \ \min\{\psi_1, \psi_2, \dots, \psi_E\}\} \quad (3.25)$$

$$\text{YHMAKS} = maks\{maks\{\varphi_1, \varphi_2, \dots, \varphi_E\} \ , \ maks\{\psi_1, \psi_2, \dots, \psi_E\}\} \quad (3.26)$$

GHV-I OGH-I (Fernandez ve Gomez 2007):

$$\text{GHV-I}=\frac{\sum_{j=1}^{E} \varphi_j}{E} \quad (3.27)$$

$$\text{OGH-I}= \frac{\sum_{j=1}^{E} w_j}{E} \quad (3.28)$$

(x^{***}, y^{***}), standart etkin sınır üzerindeki noktalardan (x^*, y^*) noktasına en yakın noktadır.

OÖU, GHV-II, OGH-II (Cura 2009a):

$$\text{OÖU}= \frac{\sum_{j=1}^{E} \sqrt{(x^{***}-x^*)+(y^{***}-y^*)}}{E} \quad (3.29)$$

$$\text{GHV-II}= \frac{\sum_{j=1}^{E} \frac{100|(x^{***}-x^*)|}{x^*}}{E} \quad (3.30)$$

$$\text{OGH-II}= \frac{\sum_{j=1}^{E} \frac{100|(y^{***}-y^*)|}{y^*}}{E} \quad (3.31)$$

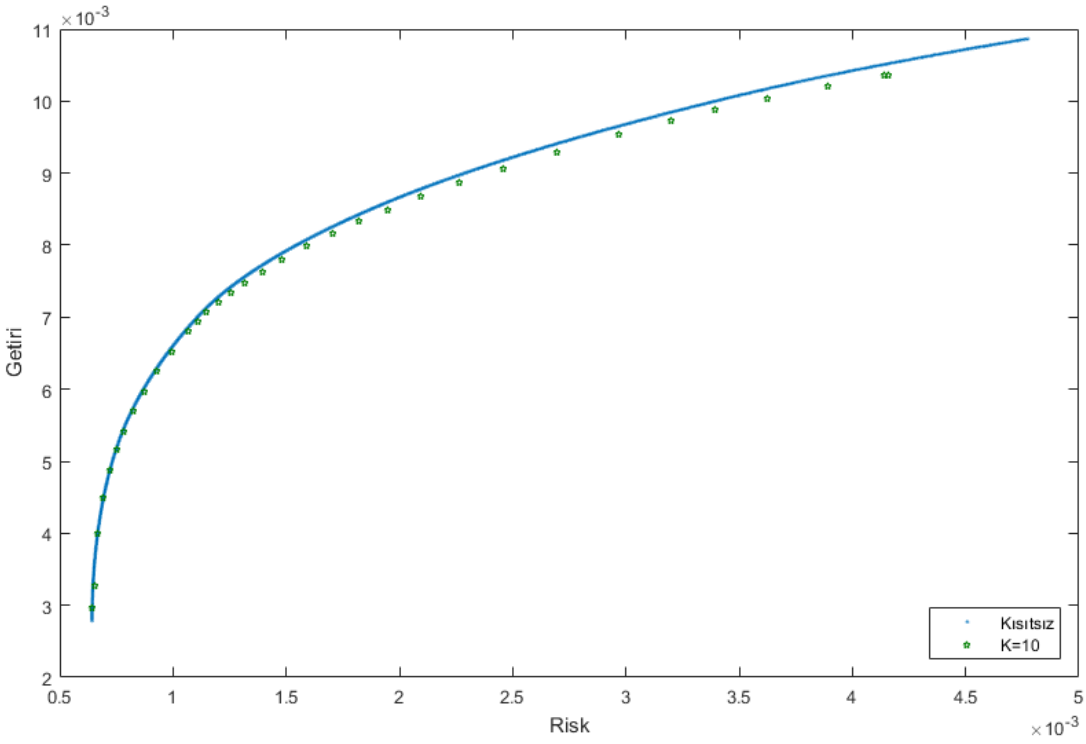
Eleman sayısı kısıtlı portföy optimizasyonu problemi için Denklem (3.8) - (3.14)'de verilen matematiksel model, yüksek seviye bir modelleme dili olan, GAMS (General Algebraic Modeling System) yardımı ile kodlanmıştır. GAMS programında, KTDOP modelinin çözümünde kullanılan COUNNE ve BONMIN çözücüleri ile çözüm aranmıştır. Ayrıca eleman sayısı kısıtlı portföy optimizasyonu problemi için

Denklem (3.8) - (3.14)'de verilen matematiksel model, doğası gereği aynı zamanda kuadratik programlama kullanılarak da çözümlenebilmektedir. Bu kapsamda doğrusal programlama modellerinde olduğu kadar kuadratik programlama modellerinin çözümünde de yaygın olarak kullanılan CPLEX çözücüsü ile de çözüm aranmıştır. BONMIN çözücüsü ile problem ayrıca kuadratik programlama modeli olarak da çözülmüştür.

Kodlanan modelin doğrulanması ve geçerliliğinin sınanması için küçük çaplı bir ver seti olan Hang Seng indeksine ait hisseler kullanılmıştır. Modelde portföy içerisindeki varlık sayısı kısıtı $K = 10$ olarak alınmıştır. Portföydeki varlıkların minimum ağırlığı ise $\varepsilon = 0.01$ olarak kabul edilmiştir. Kullanılan 3 farklı çözücü ile küçük çaplı veri seti için elde edilen sonuçlar OÖU, GHV-II (%), OGH-II (%) ve çözüm zamanlarına göre Tablo 3.8'de yer almaktadır. Bu çözücüler arasında COUNNE çözücüsü, Hang Seng veri seti için önceden belirlenmiş bir zaman limitinde (2 saat) çözüm üretememiştir. Diğer çözücüler arasında en iyi çözümü sunan çözücü CPLEX çözücüsüdür. CPLEX çözücüsünde KTDOP problem tipi mevcut değildir. Hang Seng için literatürde yer alan kısıtsız ve CPLEX ile elde edilen kısıtlı etkin sınırlar Şekil 3.7'da verilmiştir.

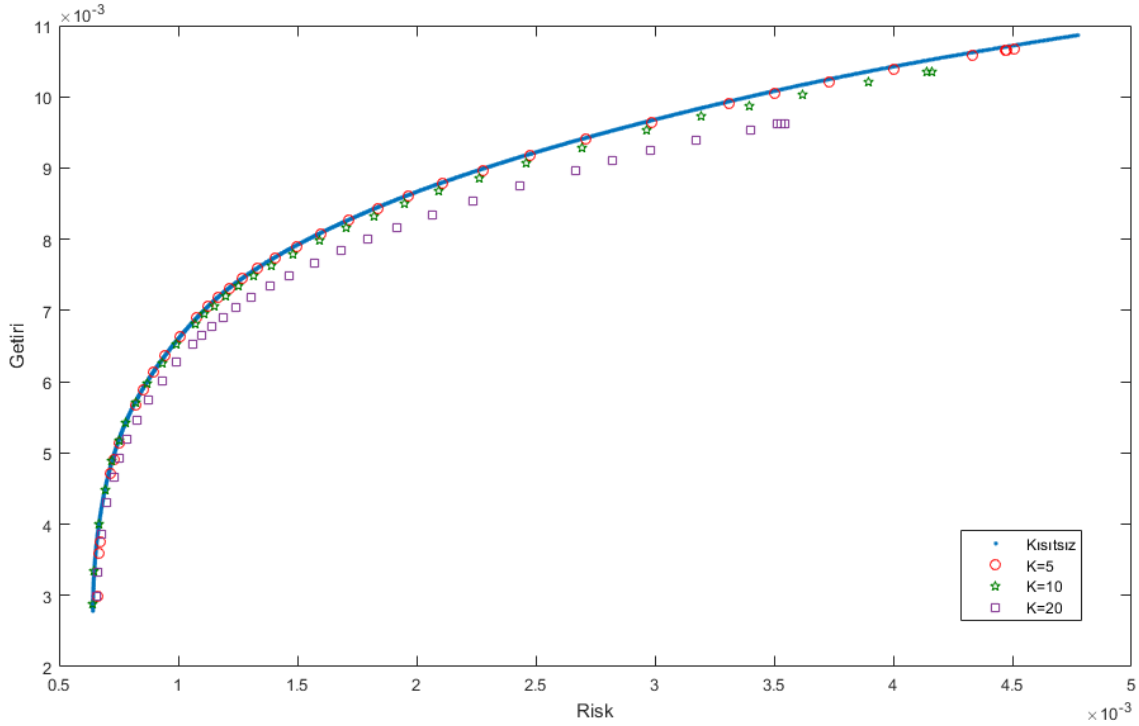
Tablo 3.8:Farklı çözücüler için Hang Seng veri seti ile GAMS çözümleri

	MINLP				MIQCP			
Hatalar / Çözücü	*OÖU*	*GHV-II (%)*	*OGH-II (%)*	*Zaman (s)*	*OÖU*	*GHV-II (%)*	*OGH-II (%)*	*Zaman (s)*
COUNNE	-	-	-	-	-	-	-	-
BONMIN	*0.0001*	*1.6463*	*0.6019*	*18.832*	*0.0001*	*1.6463*	*0.6020*	*18.930*
CPLEX	*Mevcut değil*				*0.0001*	*1.6275*	*0.6034*	*8.746*



Şekil 3.7:Hang Seng veri setinde K=10 değeri için CPLEX ile bulunan etkin sınır

Şekil 3.8'de görüldüğü gibi kısıtsız modele, ESK (K) eklendiğinde, kısıtlı etkin sınır üzerindeki noktalar kısıtsız etkin sınırdan uzaklaşmıştır. Bunun sebebi modele portföyde K adet varlık bulundurması zorunluluğu getirilmesidir. Model kısıtsız olduğunda, portföy bir risk değerindeki K'dan az sayıda varlık içerirken, kısıtlandırıldığı zaman varlık sayısını K'ya tamamlamak için getirisi daha düşük varlıklara yatırım yapmak zorunda kalabilmektedir. Bu da o risk değeri için etkin sınırdaki noktayı daha az getiri sağlayacak şekilde aşağıya çekmektedir. Eğer kısıttaki K değeri artırılırsa, model portföye daha az getirisi olan varlıklardan daha fazla koyacağı için kısıtlı etkin sınır ve kısıtsız etkin sınırdaki noktalar arasındaki mesafe artacaktır. Şekil 3.8'de Hang Seng veri seti için K değerinin sırası ile 5, 10 ve 20 olduğu durumlar için CPLEX ile bulunan kısıtlı sınırlar ve literatürde yer alan kısıtsız etkin sınır yer almaktadır. Model, düşük risk seviyeleri için beklenilen getiriyi rahatlıkla karşılayabiliyor iken, risk seviyesi arttığında beklenilen getirinin karşılanamaması nedeniyle noktalar arasındaki mesafe artmaktadır. Tablo 3.9'da farklı K değerleri için CPLEX ile bulunan sonuçların belirlenen performans ölçütlerine göre analizi de bu durumu özetlemektedir.



Şekil 3.8: Hang Seng veri setinde farklı K değerleri için CPLEX ile bulunan etkin sınırlar

Tablo 3.9: Hang Seng veri setinde farklı K değerleri için CPLEX ile bulunan sonuçların performans analizi

		Performans Ölçütü		
		OÖU	*GHV-II (%)*	*OGH-II (%)*
	5	0.0000	0.5365	0.1665
K	10	0.0001	1.6282	0.6037
	20	0.0002	6.9439	1.7929

Küçük çaplı veri setlerinin optimal olarak çözümünde genel olarak başarı sağlayan test edilen çözücüler, DAX 100 gibi karmaşık veri setlerinin çözümünde ise başarı sağlayamamıştır. Bu sonuçlar göstermektedir ki kesin çözüm üreten bu araçlar, ele alınan probleme ilişkin veri setleri büyüdüğünde, karmaşıklıkları artığında ve/veya yeni kısıtlar probleme eklendiğinde yetersiz kalabilmektedir. Bu nedenle, kısa sürelerde yaklaşık çözüm üretebilen, büyük çaplı karmaşık veri setlerinin çözümünde kullanılabilecek sezgisel yaklaşımlı algoritmaların geliştirilmesine ihtiyaç duyulmaktadır.

3.3 Yapay Arı Kolonisi

Yapay arı kolonisi (YAK) algoritması, Karaboga (2005a) tarafından bal arılarının doğada yiyecek ararken yaptıkları hareketlerden esinlenilerek önerilmiştir. Bu algoritmada üç tip arı: işçi arı, gözcü arı ve kaşif arı koordineli olarak çalışmaktadırlar. İşçi arılar besin kaynaklarındaki nektarı sömürürler ve besin kaynağının pozisyonunu ve nektar miktarını dans yoluyla gözcü arırlarla paylaşırlar. Daha sonra gözcü arılar hangi besin kaynaklarına gidilmesi gerektiğini aldıkları bilgilere göre belirlerler. Eğer bilinen bütün besin kaynaklarındaki nektarlar bitmiş ise kaşif arılar devreye girerek rasgele yeni besin kaynakları arayışına geçerler.

Karaboga ve Akay (2011)'e göre genel YAK algoritması PO probleminin çözümü için uygundur. Standart YAK algoritması Şekil 3.9'de verilmiştir. YAK algoritması düzenli rasgele bir dağılıma sahip bir populasyonun Denklem 3.32 yardımı ile yaratılmasıyla başlar (Şekil 3.9, adım 9).

$$x_{ij} = x_j^{min} + r(x_j^{max} - x_j^{min}) \qquad i = 1,2,\dots,SN \text{ and } j = 1,2,\dots,D, r \in [-1,1] \quad (3.32)$$

burada SN populasyondaki çözüm sayısını D ise problem boyutunu (hisse sayısını) ve r ise düzgün dağılıma sahip rasgele bir sayıyı temsil etmektedirler.

İşçi arı aşamasında (Şekil 3.9, adım 12-22), modifikasyon oranına (mr) göre olasılıklı olarak güncel çözümlerden yeni çözümler üretilir, burada mr [0,1] aralığında populasyondaki çözümlerin ne kadarlık bir kısmının değiştirleceğini belirleyen bir kontrol parametresidir. Büyük bir modifikasyon oranı populasyonda önemli değişikliklere sebep olurken, küçük bir modifikasyon oranı ise küçük değişikliklere sebep olur. Denklem (3.33) kullanılarak bellekte bulunan önceki bir çözümden yeni bir çözüm elde edilir.

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \qquad k = 1,2,\dots,SN \text{ and } k \neq i, \phi_{ij} \in [-1,1] \quad (3.33)$$

burada k indeks i'den farklı olarak rasgele seçilen bir indekstir ve ϕ_{ij} düzgün dağılıma sahip rasgele bir sayıdır. Böylece, seçilen arının mevcut konumuna ve popülasyondan rastgele seçilen başka bir arının pozisyonuna göre yeni bir pozisyonun hesaplanmasıyla çözüm elde edilir.

Gözcü arı aşamasında (Şekil 3.9, adım 23-32), iyi bir çözüm olasılıklı olarak seçilerek, gözcü arılar tarafından sömürülür. Nektar bilgileri değerlendirilir ve bu nektar miktarı ile ilişkili olasılık değeri P_{x_i} ile doğru orantılı olarak bir gıda kaynağı olasılık olarak seçilir:

$$P_{x_i} = {f_{x_i}}\Big/{\sum_{i=1}^{SN} f_{x_i}} \tag{3.34}$$

burada f_{x_i} çözüm x_i'nin uygunluk değeridir. İşçi ve gözcü arı aşamalarını her ikisinde de eğer sömürülen çözümden daha iyi bir çözü elde edilmez ise ilişkili limit sayacı arttırılır, aksi halde bulunan iyi çözüm populasyondaki eski çözümle yerdeğiştirilir. İşçi ve gözcü arı aşamaları arılar sömürme prosesini bitirinceye kadar devam eder.

Kaşif arı aşamasında (Şekil 3.9, adım 33-42), diğer bir kontrol parametresi olan kaşif arı üretim periyodu (spp) kullanılarak arama prosesi geciktirilmiştir. Her bir spp periyodunda, eğer belirli bir çözüm için hesaplanan limit sayacı önceden belirlenen limit değerini ($ilimit$) aşmış ise, bir kaşif arı keşfedilmemiş çözüm uzayında yeni bir çözüm aramak için salınır. Algoritmanın kaşif arı üretme prosesi populasyona uygun olmayan bir çözümün girmesine izin vererek bir çeşitlilik mekanızması sağlar. Bu üç aşama aşılması gereken iterasyon sayısı (T) aşılıncaya kadar devam etmektedir.

Algoritma: Yapay Arı Kolonisi-I
x_s: populasyondaki s. çözüm, $s = 1, \dots, SN$
v_s: s. çözüm kullanılarak üretilen yeni çözüm
T: maksimum iterasyon sayıs
$limit_s$: s. çözüm için limit sayacı, $s = 1, \dots, ps$
$ilimit$: kaşif arıların devreye girmesi için aşılması gereken limit değri
r_s: [0,1] aralığında düzenli dağılıma sahip rasgele üretilmiş sayı
başla
  Denklem (3.32) kullanarak rasgele bir başlangıç populasyonu yarat
  iterasyon = 1
  tekrar et
    //işçi arı aşaması
    $s = 1$
    tekrar et
      eğer $r_s < mr$ **ise**
      Çözüm x_s'ten Denklem (3.33) kullanılarak yeni bir çözüm v_s üret
        eğer $f_{v_s} < f_{x_s}$ **ise** $limit_s = 0$ **ve** $x_s \leftarrow v_s$
        aksi halde $limit_s = limit_s + 1$
        bitir
      $s = s + 1$
      bitir
    $s < SN$ **oluncaya kadar**
    //gözcü arı aşaması
    $l = 1$
    tekrar et
      Denklem (3.34) kullanılarak populasyondan bir çözüm seç
      Seçilen çözüm x_l'den Denklem (3.33) kullanılarak yeni bir çözüm v_s üret
      eğer $f_{v_l} < f_{x_l}$ **ise** $limit_l = 0$ **ve** $x_l \leftarrow v_l$
      aksi halde $limit_l = limit_l + 1$
      bitir
      $l = l + 1$
    $l < SN$ **oluncaya kadar**
    //kaşif arı aşaması
    eğer mod(*iteration, spp)* = 0 **ise**
    $m = 1$
    tekrar et
      eğer $limit_m > ilimit$ **ise**
      x_m çözümünün yerine Denklem (3.32) kullanarak yeni bir çözüm üret
      bitir
      $m = m + 1$
    $m < SN$ **oluncaya kadar**
    bitir
    Iteration = iteration + 1
  $iterasyon = T$ **oluncaya kadar**
bitir

Şekil 3.9: Yapay arı kolonisi algoritması

3.3.1 Kısıt İşleme için Tamir Mekanizması

Bir çözümün uygulanabilir olduğundan emin olmak için problemdeki kısıtlar (Denklem 3.9 - 3.14) sağlanmalıdır. Litertürde, (Chang ve diğ. 2000) tarafından önerilen bir tamir prosedürü (Şekil 3.10) kısıtları sağlamak için sıkça kullanılmıştır (Anagnostopoulos ve Mamanis 2011a, b; Chang ve diğ. 2000; Chang ve diğ. 2009; Khalidji ve diğ. 2009; Lwin ve diğ. 2014; Mashayekhi ve Omrani 2016; Mishra ve diğ. 2014a; Moral-Escudero ve diğ. 2006; Pai ve Michel 2009; Woodside-Oriakhi ve diğ. 2011). Böylece çözümler belirlenen limitler içinde kalması için zorlanır.

Prosedür: Tamir prosedürü-I
Girdi: N, K, S, W
Çıktı: W^r
N: İndeksteki toplam varlık sayısı
K: Portföyde bulunmasına izin verilen toplam varlık sayısı
S: İndeksteki varlıklar kümesi
W: Tamire giren portföyün ağırlıkları kümesi
w_i: Tamire giren i. varlığın ağırlığı $i \in S$
w_i^r: Tamirden çıkan i. varlığın ağırlığı $i \in S$
W^r: Tamire giren portföyün ağırlıkları kümesi
z_i: i. varlığın portföyde yer alıp almama durumunu gösteren ikili değişken, $i \in S$
başla
tekrar et
  eğer $\sum_{i=1}^{N} z_i > K$ **ise**
    Portföyde bulunan en küçük w_i değerine sahip varlık için $z_i = 0,\ w_i = 0$ yap, $i \in S$
  eğer $\sum_{i=1}^{N} z_i < K$ **ise**
    Portföyde bulunmayan rasgele bir z_i değeri için $z_i = 1$ yap, $i \in S$
$\sum_{i=1}^{N} z_i = K$ **oluncaya kadar**
$CS = \sum_{i=1}^{N} w_i \quad i \in S,$ */* portföydeki varlıkların ağırlıkları toplamı */*
$FP = 1 - \varepsilon * K$ */* portföydeki varlıklara atanabilecek maksimum ağırlık */*
$w_i^r = \varepsilon * z_i + w_i * z_i * FP/CS \quad \forall i \in W, \forall i \in W^r$
bitir

Şekil 3.10: Tamir prosedürü-I

3.3.2 Değerlendirme Mekanizması

Bir çözüm risk ve getiri değerleri açısından ilişkili uygunluk değeriyle birlikte değerlendirilmelidir. Şekil 3.11 Chang ve diğ. (2000) tarafından önerilen bir değerlendirme mekanizmasını göstermektedir. Temel olarak bu prosedür, toplam risk, toplam getiri ve uygunlukları, portföydeki hisslerin ağırlıklarını, getirilerini ve varyans-kovaryans verilerini kullanarak hesaplar. Bu prosedürün uygulanması, Şekil

3.10'de verilen tamir prosedürü kullanılarak uygulanabilirliklerinden emin olunmuş ise, oldukça basittir.

1: **Prosedür:** Değerlendirme prosedürü-I
2: **Girdi:** $W^r, N, VC_{ij}, R_i, \lambda$
3: **Çıktı:** R^*, V^*, f
4: W^r: Varlıkların ağırlıkları kümesi
5: N: İndeksteki toplam varlık sayısı
6: f: Portföyün uygunluk değeri
7: R^*: Portföyün toplam getirisi
8: V^*: Portföyün varyansı (Riski)
9: VC_{ij}: i. ve j. varlıklar arasındaki varyans-kovaryans değeri $i, j \in S$
10: w_i^r: i. Varlığın ağırlığı $w_i^r \in W^r$
11: R_i: i. Varlığın ortalama getirisi $i \in S$
12: **Başla**
13: $R^* = \sum_{i=1}^{N} w_i^r R_i$ /* *portföyün toplam getirisi* */
14: $V^* = \sum_{i=1}^{N} \sum_{j=1}^{N} w_i^r w_j^r VC_{ij}$ /* *portföyün toplam risk değeri* */
15: $f = \lambda V^* - (1-\lambda) R^*$ /* λ *ile hesaplanan uygunluk değeri* */
16: **Bitir**

Şekil 3.11: Değerlendirme prosedürü–I

3.4 Geliştirilmiş YAK algoritması

Bu tezde, PO problemini çözmek için iki farklı YAK algoritması geliştirilmiştir. Bölüm 3.3'de verilen ilk YAK algoritması, Bölüm 3.3.1'de anlatılan popüler tamir mekanizmasına dayanmakta iken, ikinci YAK algoritmasında, (Deb 2000a) ve (Karaboga ve Akay 2011) temel alınlarak, uygulanabilir olmayan çözümlere izin veren kısmi bir tamir prosedürü önerilmiştir. Her iki algoritma da Şekil 3.9'de verilen YAK algoritması adımlarını kullanmaktadırlar.

Şekil 3.10'de verilen tamir prosedürünün dez avantajı algoritmanın çözüm uzayından serbestçe dolaşmasını engellemektir, bu da zayıf bir yakınsmaya sebep oluyor. Bu nedenle, Denklem (3.9) ve (3.10) da verilen kısıtların sağlanmasını garanti ederken, Denklem (3.11)'de verilen kısıtı yumuşak bir kısıt olarak kabul ederek kısmı serbest arama yapılmasına imkan veren alternatif bir tamir prosedürü geliştirilmiştir (Şekil 3.12). Böylece, Şekil 3.10'deki adım 21'de verilen popüler denklemin çıkarılmasıyla yeni bir mekanizma elde edilir.

Eğer Tamir prosedürü-I'in yerine Tamir prosdürü-II kullanılır ise, çözümlerin uygulanabilirliği kesin olarak sağlanmadığından Değerlendirme prosedürü-I direkt olarak uygulanamaz. Bu nedenle, alternatif bir değerlendirme prosedürü geliştirilmiştir.

Prosedür: Tamir prosedürü-II
Girdi: N, K, S, W
Çıktı: W^r
N: İndeksteki toplam varlık sayısı
K: Portföyde bulunmasına izin verilen toplam varlık sayısı
S: İndeksteki varlıklar kümesi
W: Tamire giren portföyün ağırlıkları kümesi
w_i: Tamire giren i. varlığın ağırlığı $i \in S$
w_i^r: Tamirden çıkan i. varlığın ağırlığı $i \in S$
W^r: Tamire giren portföyün ağırlıkları kümesi
z_i: i. varlığın portföyde yer alıp almama durumunu gösteren ikili değişken, $i \in S$
başla
tekrar et
    eğer $\sum_{i=1}^{N} z_i > K$ **ise**
        Portföyde bulunan en küçük w_i değerine sahip varlık için $z_i = 0,\ w_i = 0$ yap, $i \in S$
    eğer $\sum_{i=1}^{N} z_i < K$ **ise**
        Portföyde bulunmayan rasgele bir z_i değeri için $z_i = 1$ yap, $i \in S$
$\sum_{i=1}^{N} z_i = K$ **oluncaya kadar**
$CS = \sum_{i=1}^{N} w_i \quad i \in S,$ */* portföydeki varlıkların ağırlıkları toplamı */*
$w_i^r = w_i * z_i / CS \quad \forall i \in W, \forall i \in W^r$
bitir

Şekil 3.12: Tamir prosedürü-II

Prosedür: Değerlendirme prosedürü-II
Girdi: n, k, ε, w
Çıktı: w
N: İndeksteki toplam varlık sayısı
K: Portföyde bulunmasına izin verilen toplam varlık sayısı
S: İndeksteki varlıklar kümesi
ε: bir varlığın alabileceği minimum ağırlık
W: Tamire giren portföydeki toplam varlık sayısı
w_i: Tamire giren i. varlığın ağırlığı $i \in S$
vio: çözümün ne kadar uygun olmadığını gösteren değişken
u: çözümün uygulanabilir olup olmadığını gösteren ikili değişken
başla
$vio = |W - K|$
tekrar et
    eğer $w_i = 0$ **ise devam et**
    değilse eğer $w_i < \varepsilon$
        $vio = vio + 1$
$i = n$ **olana kadar**
eğer $vio = 0$ **ise** $u = 0$ */*uygulanabilir (feasible) bir çözüm değil*/*
değilse $u = 1$
$R^* = \sum_{i=1}^{N} w_i^r R_i$ */* toplam portföy getirisi */*
$V^* = \sum_{i=1}^{N} \sum_{j=1}^{N} w_i^r w_j^r VC_{ij}$ */* toplam portföy riski */*
$f = \lambda V^* - (1 - \lambda) R^*$ */* $uygunluk\ değeri$*/*
bitir

Şekil 3.13: Değerlendirme prosedürü-II

PO problemi kısıtlarının sağlanmasındaki ana odak, uygulanabilirliğe zorlayarak, uygulanabilir olmayan çözümlere tolerans sağlamaktır. Deb (2000a) tarafındand geliştirilen yöntem temel alınarak, uygulanabilir ve kısıt ihlali az olan çözümlere uygulanabilir olmayan çözümlerden daha fazla öncelik veren bir seçim prosedürü kullanılmıştır (Şekil 3.14). Bu prosedür uygulanabilir çözümlere en yüksek öncelik, az ihlalli çözümlere orta öncelik ve çok ihlalli çözümlere ise az öncelik vermektedir. Böylece, küçük kıstı ihlalleri her zaman büyük kısıt ihlallerine tercih edilmektedir.

Procedure: Seçim prosedürü
Girdi: $PC, POPC, limit_s, feasibility, violation$
Çıktı: $POPC, limit,$
$POPC$: populasyondaki kıyaslanan çözüm
PC: eski çözümden üretilen yeni çözüm
$uygulanabilirlilik$: çözümün uygulanabilir olup olmadığını gösteren ikili değişken
$ihlal$: çözümün alt sınır (ε_i) ve üst sınır (δ_i) ihlal değeri
$f(x)$: x çözümünün uygunluk değeri
$limit$: çözümün limit sayacı
başla
    eğer $uygulanabilirlilik(PC)$=1 && $uygulanabilirlilik(POPC)$=1 **ise**
        eğer $f(PC) < f(POPC)$ **ise**
            $POPC = PC$
            $limit := 0$
        aksi halde
            $limit = limit + 1$
    aksi halde eğer $uygulanabilirlilik(PC)$=1 && $uygulanabilirlilik(POPC)$=0 **ise**
        $POPC = PC$
        $limit := 0$
    aksi halde eğer $uygulanabilirlilik(PC)$=0 && $uygulanabilirlilik(POPC)$=1 **ise**
        $limit = limit + 1$
    aksi halde eğer $uygulanabilirlilik(PC)$=0 && $uygulanabilirlilik(POPC)$=0 **ise**
        eğer $ihlal(PC) < ihlal(POPC)$ **ise**
            $POPC = PC$
            $limit := 0$
        aksi halde
            $limit = limit + 1$
bitir

Şekil 3.14: Seçim prosedürü

Şekil 3.15'de verilen olasılık hesaplama prosedürü kısıtları ihlal eden çözümlerin seçimini sınırlamaktadır (Karaboga ve Akay 2011). Deb (2000a) tarafından önerilen kısıt elleçleme metodunun algoritmada tek başına kullanılması, uygulanabilir çözümler uygulanamayan çözümlere her zaman tercih edileceği için çeşitlilik eksikliğine sebep olacaktır (Karaboga ve Akay 2011). İşçi yada gözcü arılar tarafından populasyondaki bir çözümden üretilen yeni çözüm, sade ve sadece eski çözüm uygulanabilir olmadığında yada daha yüksek bir ihlale sahip olduğunda populasyona girebilir. Eğer eski çözüm uygulanabilir ise uygulanabilir olmayan bir çözüm asla populasyona giremez. Bu sebepten dolayı, çeşitlilik sınırlanır ve arama mekanızması yeterince etkin olmayabilir. Arama mekanizmasını güçlendirmek ve çeşitlendirmeyi sağlamak için, YAK algoritmasının güçlü bir mekanizması olan kaşif arılar, popülasyona katılmak için uygun olmayan çözümlere izin vererek uygulanabilir olmayan çözüm uzayında geçici bir aramaya izin verir. Böylece daha iyi bir yakınsama elde edilir.

Prosedür: Olasılık hesaplama prosedürü
Girdi: $PC, POPC, limit_s, uygulanabilirlilik, ihlal$
Çıktı: $POPC$, $limit$,
POP_i: populasyondaki i. kromozom
$olasılık$: rulet tekerinde seçilme olasılığı
$uygulanabilirlilik$: çözümün uygulanabilir olup olmadığını gösteren ikili değişken
$ihlal$: çözümün alt sınır (ε_i) ve üst sınır (δ_i) ihlal değeri
başla
$i := 1$
  tekrar et
    eğer $uygulanabilirlilik(POP_i) = 1$ **ise**
      eğer $f(POP_i) < 0$ **ise**
      $K_i = 1 + |f(POP_i)|$
      aksi halde
      $K_i = 1/(1 + f(POP_i))$
    i++
  $i = N$ **oluncaya kadar**
  $i := 1$
  tekrar et
    eğer $uygulanabilirlilik(POP_i) = 1$ **ise**
      $olasılık(POP_i) = 0.5 + 0.5\left(K_i \Big/ \sum_i^N K_i\right)$
      aksi halde
      $olasılık(POP_i) = 0.5\left(1 - ihlal(POP_i) \Big/ \sum_i^N ihlal(POP_i)\right)$
    i++
  $i = N$ **oluncaya kadar**
bitir

Şekil 3.15: Olasılık hesaplama prosedürü

Bölüm 3.3'de verilen standart YAK algoritmasının kısıt işleme mekanizması değiştirilip, yukarıda anlatılan prosedürler kullanıldığında, uygulanabilir çözümlere izin veren yeni bir YAK-II (Şekil 3.16) algoritması elde ederiz. Her iki algoritmada kullanılan prosedürler Tablo 3.10'de verilmiştir.

Tablo 3.10: YAK-I ve YAK-II algoritmalarında kullanılan prosedürler

YAK-I	YAK-II
Tamir prosedürü-I	Tamir prosedürü-II
Değerlendirme prosedürü-I	Değerlendirme prosedürü-II
Standart rulet tekeri seçim prosedürü	Seçim prosedürü (Deb 2000b) ve olasılık hesaplama prosedürü (Karaboga ve Akay 2011)

Algoritma: YAK-II
Girdi: Veri (R_i, VC_{ij}) ve parametreler $(\varepsilon, K, E, ps, klimit, IT)$
Çıktı: H
n: İndeksteki toplam varlık sayısı
ps: Popülasyon büyüklüğü
POP: popülasyon
E: tanımlanan λ sayısı
T: maksimum iterasyon sayısı
H: son popülasyon
EB: işçi arı safhasından üretilen ağırlıklar kümesi
OB: gözcü arı safhasından üretilen ağırlıklar kümesi
f: portföyün uygunluk değeri
$limit_s$: popülasyondaki çözümlerin limit değerleri, $s = 1, \dots, ps$
r: düzgün bir rassal sayı, $r \in [-1,1]$
rk: düzgün rassal bir tamsayı, $rk \in [1, ps]$
$klimit$: kaşif arıların devreye girmesi için aşılması gereken limit değeri
başla
$H = \emptyset$
$e = 1$
tekrar Et
  $\lambda = (e-1)/(E-1)$
  $W_i = \varepsilon + r(1-\varepsilon), \quad i = 1, \dots, ps \quad \forall W_i \in POP$
  $W_i^{rep} \leftarrow Tamir(N, K, S, W_i, z_i) \;\; i = 1, \dots, ps \quad \forall W_i \in POP$
  $EB \leftarrow Uygunluk\ hesapla(EB, N, VC_{ij}, R_i, \lambda)$
  $POP = \{W_1^{rep}, \dots, W_i^{rep} \dots, W_{ps}^{rep}\}$
  /*işçi arı aşaması*/
  $s = 1$
  tekrar Et
    $k = rk, rk \in [1, ps], rk \neq s$
    $EB_i \leftarrow POP_{si} + rPOP_{si} - POP_{ki} \qquad /* \; i = 1, \dots, N, s = 1, \dots, ps \; */$
    $EB \leftarrow TamirII(N, K, S, EB)$
    $EB \leftarrow Değerlendirme\ prosed$ürü $- II(n, k, \varepsilon, EB)$
    $EB \leftarrow Seçim\ prosed$ürü$(EB, POP_S)$ /*seçim prosedünde iki çözüm karşılaştırılıp daha iyi çözüm EB çözümüyle değiştiriliyor ve gelişme olup olmadığı ikili değişkene atılıyor*/
    $POP_s \leftarrow EB$
    eğer $gelişme = 1$ **ise**
    $limit_s = 0$
    aksi halde $limit_s = limit_s + 1$
    $s = s + 1$
  $s = ps$ **oluncaya kadar**
  /*gözcü arı aşaması*/
  $s = 1$
  tekrar Et
    $l \leftarrow rulet\ tekeri(N, POP)$
    $OB = POP_l$
    $k = rk, rk \in [1, ps], rk \neq l$
    $OB_i = POP_{li} + r * (POP_{li} - POP_{ki}), \quad i = 1, \dots, N$
    $OB \leftarrow TamirII(N, K, S, OB)$
    $OB \leftarrow Değerlendirme\ prosed$ürü $- II(n, k, \varepsilon, OB)$
    $OB \leftarrow Seçim\ prosed$ürü$(OB, POP_S)$ /*seçim prosedüründe iki çözüm karşılaştırılıp daha iyi çözüm OB çözümüyle değiştiriliyor ve gelişme olup olmadığı ikili değişkene atılıyor*/
    $POP_s \leftarrow EB$
    eğer $gelişme = 1$ **ise**
    $limit_s = 0$
    aksi halde $limit_s = limit_s + 1$
    $s = s + 1$
  $s = ps/2$ **oluncaya kadar**
  /*kaşif arı aşaması*/
  $s = 1$
  tekrar Et
    eğer $limit_s > klimit$ **ise**
      $W_i = \varepsilon + r(1-\varepsilon), \quad i = 1, \dots, ps \quad \forall W_i \in POP$
      $SB \leftarrow TamirII(N, K, S, W_i)$
      $SB \leftarrow Değerlendirme\ prosed$ürü $- II(n, k, \varepsilon, SB)$
      $POP_s \leftarrow SB$
      $limit_s = 0$
  $s = ps$ **oluncaya kadar**
  /*popülasyonun en iyi çözümü pareto optimal çözümler kümesine atılır*/
  $H_e = POP_{en\_iyi}$
$e = E$ **oluncaya kadar**
bitir

Şekil 3.16: YAK-II algoritması

4. BULGULAR

Bu bölümde algoritmaların kodlandığı programlar, parametere ayarları ve hesaplamalı sonuçları verilmiştir.

4.1 Kodlama

Önerilen her iki algoritma da, MATLAB R2016b kullanılarak modellenmiş ve test edilmiş ve daha sonra hız amacıyla Microsoft Visual C ++ 2015 ortamında yeniden kodlanmıştır. Deneyler, Intel Xeon E5-2650 2.0 GHz bir işlemciye sahip 32 GB RAM bulunduran bir bilgisayarda yapılmıştır.

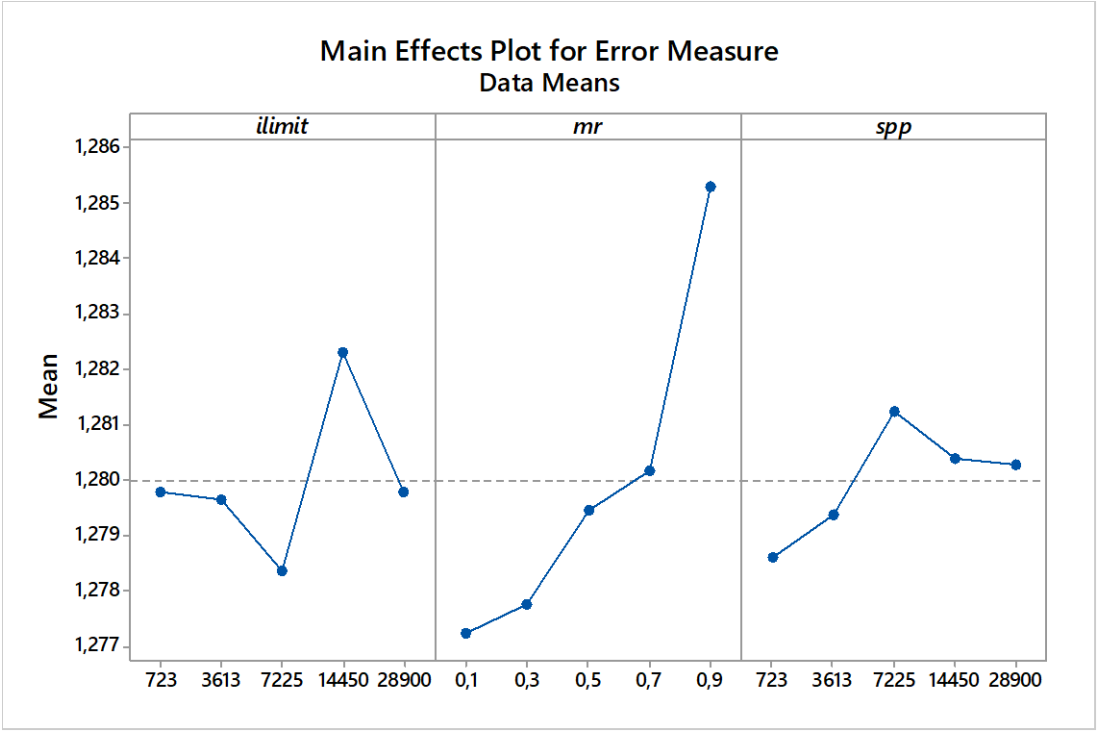
4.2 Parametre Ayarlama

Önerilen algoritmanın performansını arttırmak için parametre değerleri Tablo 4.11'de verilen faktör seviyeleri kullanılarak test edilmiştir. Tam faktoriyel deneysel tasarım kullanılarak, 85 hisseli DAX 100 veri setinde parametreler test edilmiştir. Minitab kullanılarak elde edilen ana etki grafikleri, hata ölçütü için (GHV-II) Şekil 4.17'da ve maksimum iterasyon sayısı için Şekil 4.18'de verilmiştir.

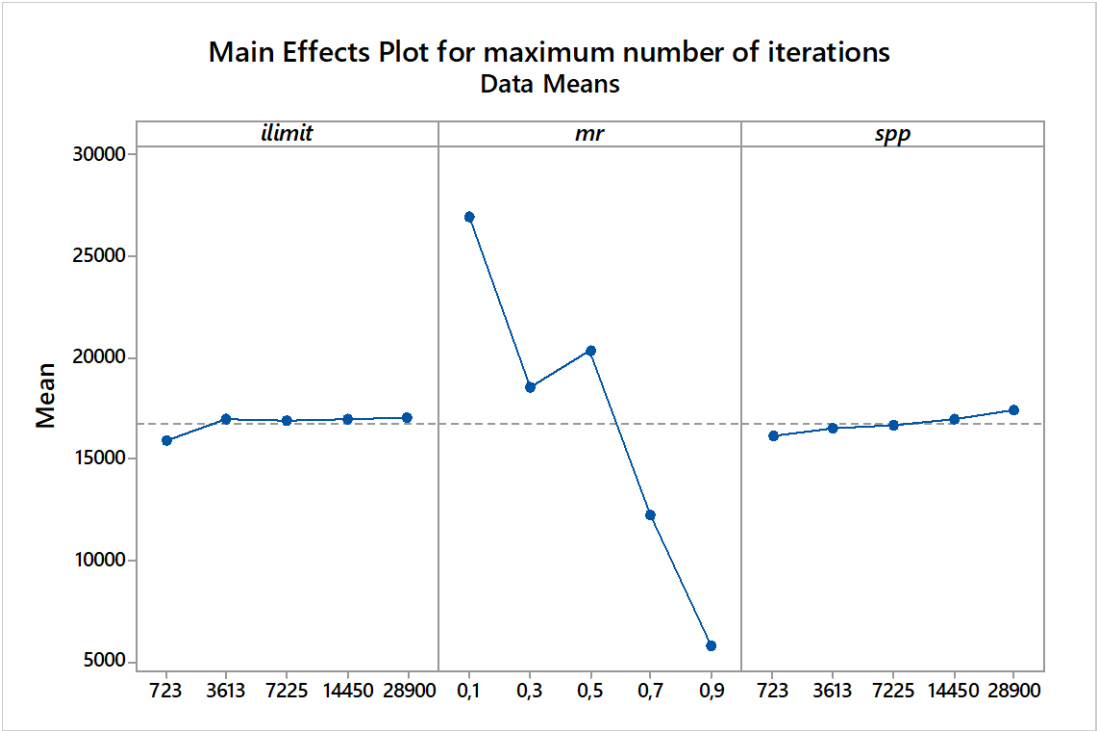
Tablo 4.11: Deneysel tasarımda kullanılan faktör seviyeleri

Seviye	$\boldsymbol{spp}$	$\boldsymbol{ilimit}$	$\boldsymbol{mr}$
1	$0.1\times \boldsymbol{SN}\times \boldsymbol{N}$	$0.1\times \boldsymbol{SN}\times \boldsymbol{N}$	0.1
2	$0.5\times \boldsymbol{SN}\times \boldsymbol{N}$	$0.5\times \boldsymbol{SN}\times \boldsymbol{N}$	0.3
3	$1\times \boldsymbol{SN}\times \boldsymbol{N}$	$1\times \boldsymbol{SN}\times \boldsymbol{N}$	0.5
4	$\boldsymbol{2}\times \boldsymbol{SN}\times \boldsymbol{N}$	$\boldsymbol{2}\times \boldsymbol{SN}\times \boldsymbol{N}$	0.7
5	$\boldsymbol{4}\times \boldsymbol{SN}\times \boldsymbol{N}$	$\boldsymbol{4}\times \boldsymbol{SN}\times \boldsymbol{N}$	0.9

spp:kaşif arı üretme periyodu, $ilimit$:kaşif arıların salınması için aşılması gereken itearsyon sayısı, mr:portföydeki hisselerin ne kadarının modifiye edileceğini belirleyen oran, $N = 85$, $SN = N$



Şekil 4.17: Hata ölçütü için ana etki grafikleri



Şekil 4.18: Maksimum iterasyon için ana etki grafikleri

Şekil 4.17 göstermektedir ki *mr* ve *spp* parametrelerinin küçük değerleri için GHV-II hatası düşmektedir. Ancak Şekil 4.18'de görüldüğü üzere *ilimit* ve *spp*

değerlerinin düşük değerleri için maksimum iterasyon sayısı düşerken mr parametresi için bunun tam tersidir. Sonuç olarak yapılan deneylerde YAK-I ve YAK-II algoritmalarında parametreler; $spp = 0.1 \times PS \times N$, $ilimit = 1 \times PS \times N$ ve $mr = 0.1$ olarak sabitlenmiştir.

Populasyon temelli algoritmalarda, populasyon büyüklüğü, YAK algoritmasında besin kaynağı sayısı ile temsil edilir ve maksimum iterasyon sayısı algoritmanın çalışma süresi üzerinde direkt bir etkiye sahiptir. PO literatüründe önerilen algoritmaların çoğunluğunda durdurma kriteri $1000 \times N$ olarark sabitlenmiştir (Chang ve diğ. 2000; Cura 2009b; Deng ve diğ. 2012b; Fernández ve Gómez 2007; Lwin ve Qu 2013). Adil bir kıyaslama yapabilmek için önerilen algoritmada, besin kaynağı sayısı problem boyutuna (N) ve maksimum değerlendirme sayısı ise $1000 \times N$ olarak sabitlenmiştir. Değerlendirme sayısı, algoritma çalışırken işçi, gözcü ve kaşif arılar tarafından üretilen yeni çözümlerin değerlendirme prosedürüne giriş sayıları hesaplanarak bulunur.

4.3 Hesaplamalı Sonuçlar

Bü bölümde önerilen algoritmanın hesaplamalı sonuçları tüm veri setleri için verilmiş ve algoritmanın performansı literatürde önerilen bazı algoritmalar ile karşılaştırılmıştır.

4.3.1 YAK-I ve YAK-II Algoritmalarının Karşılaştırmalı Sonuçları

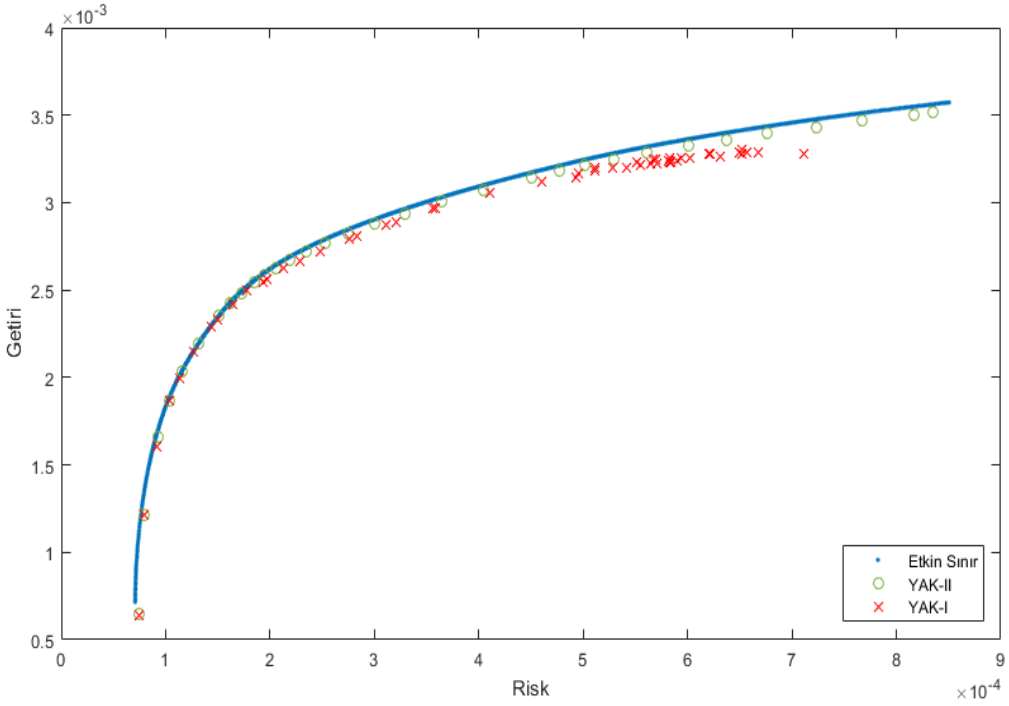
Bu bölümde YAK-I ve YAK-II algoritmalarının karşılaştırmalı sonuçları verilmiştir (Tablo 4.12). Hesaplamalı sonuçlar açıkça göstermektedir ki, bütünleştirilmiş uygulanabilir olmayan çözümlere izin veren ve uygulanabilir olmaya zorlayan tamir prosedürü kullanılan yapay arı kolonisi algoritması (YAK-II) belirgin bir şekilde sadece tamir prosedürü kullanılan YAK-I algoritmasına üstün gelmektedir. YAK-I ve YAK-II algoritmaları ile çizilen kısıtlı etkin sınırlar: Hang Seng Şekil 4.19'da, DAX 100 Şekil 4.20'de, FTSE 100 Şekil 4.21'de, S&P 100 Şekil 4.22'de, Nikkei Şekil 4.23'de, BİST30 Şekil 4.24'de ve BİST100 Şekil 4.25'de verilmiştir.

Tablo 4.12: YAK-I ve YAK-II algoritmalarının performans karşılaştırması

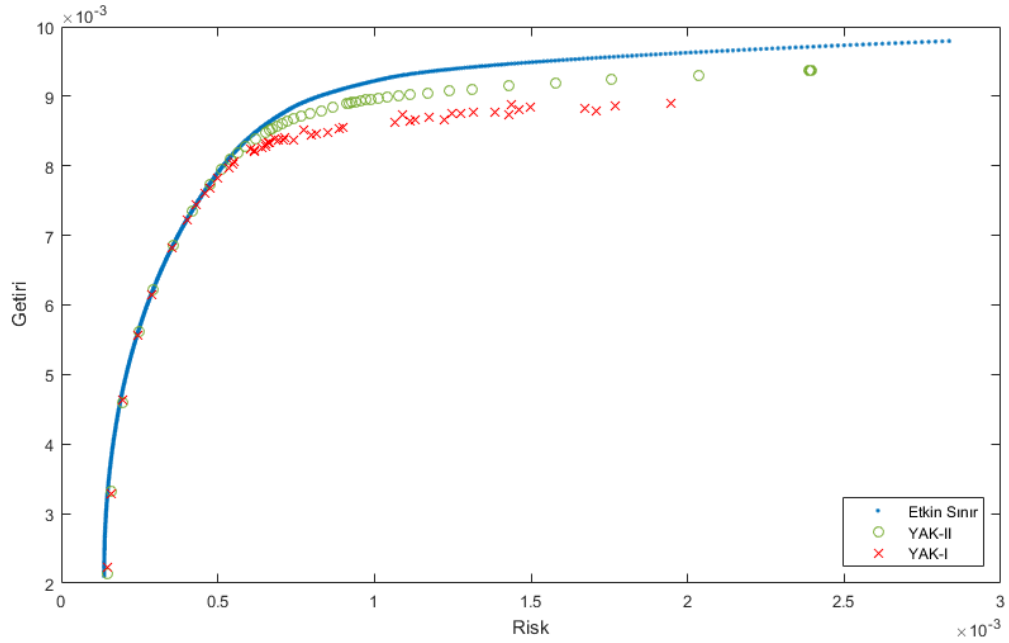
INDEKS N	Hang Seng			DAX 100			FTSE 100			S&P 100		
	31			85			89			98		
P. Ölçütü	YAK-I	YAK-II	Δ	YAK -I	YAK -II	Δ	YAK -I	YAK -II	Δ	YAK -I	YAK -II	Δ
YHO	2.4327	**1.0953**	54.9760	4.4799	**2.3295**	48.0011	1.4883	**0.8481**	43.0155	2.0948	**1.2930**	38.2757
YHMED	2.6909	**1.2181**	54.7326	4.4023	**2.5678**	41.6714	1.7776	**1.0841**	39.0133	2.0063	**1.1369**	43.3335
YHMİN	0.0329	**0.0000**	100.0000	0.0859	**0.0023**	97.3225	0.0310	**0.0047**	84.8387	0.0194	**0.0000**	100.0000
YHMAKS	4.4815	**1.5538**	65.3286	7.9642	**4.0275**	49.4299	3.2289	**2.0638**	36.0835	5.7988	**5.4422**	6.1495
GHV-I	8.9629	**4.1302**	53.9189	41.9042	**32.6157**	22.1660	7.3210	**4.5658**	37.6342	14.5024	**11.5133**	20.6111
OGH-I	2.7448	**1.1364**	58.5981	5.2692	**3.0743**	41.6553	2.1839	**1.4328**	34.3926	3.2236	**2.3309**	27.6926
OÖU	0.0002	**0.0001**	50.0000	0.0002	**0.0001**	50.0000	0.0001	**0.0000**	100.0000	**0.0001**	**0.0001**	0.0000
GHV-II	4.0351	**1.6432**	59.2773	15.9677	**6.7925**	57.4610	4.8366	**2.4397**	49.5575	5.4076	**2.5260**	53.2880
OGH-II	1.1899	**0.6047**	49.1806	1.6927	**1.2761**	24.6116	0.3926	**0.3255**	17.0912	1.2345	**0.8885**	28.0275

INDEKS N	NIKKE			XU030			XU100		
	225			30			99		
P. Ölçütü	YAK -I	YAK -II	Δ	YAK -I	YAK -II	Δ	YAK -I	YAK -II	Δ
YHO	1.2600	**0.5781**	54.1190	3.2030	**1.1256**	64.8579	2.3370	**1.0272**	56,0462
YHMED	1.3379	**0.5856**	56.2299	4.5870	**1.2549**	72.6422	2.4420	**1.1586**	52,5553
YHMİN	0.0114	**0.0000**	100.0000	0.0268	**0.0012**	95.5224	0.3779	**0.0132**	96,5070
YHMAKS	2.6911	**1.1606**	56.8727	5.9630	**2.1286**	64.3032	3.7930	**1.7384**	54,1682
GHV-I	9.0457	**5.5757**	38.3608	11.7200	**6.5390**	44,2065	12.22	**5.6169**	54,0352
OGH-I	2.6666	**1.6764**	37.1334	3.5100	**1.2537**	64,2821	3.325	**2.0126**	39,4707
OÖU	**0.0000**	**0.0000**	0.0000	**0.0000**	**0.0000**	0.0000	**0.0000**	**0.0000**	0.0000
GHV-II	1.9814	**0.8396**	57.6259	7.2240	**1.9324**	73,2503	6.6050	**2.4973**	62,1908
OGH-II	0.8514	**0.4127**	51.5269	1.1670	**0.7358**	36,9494	1.1200	**0.7306**	34,7679

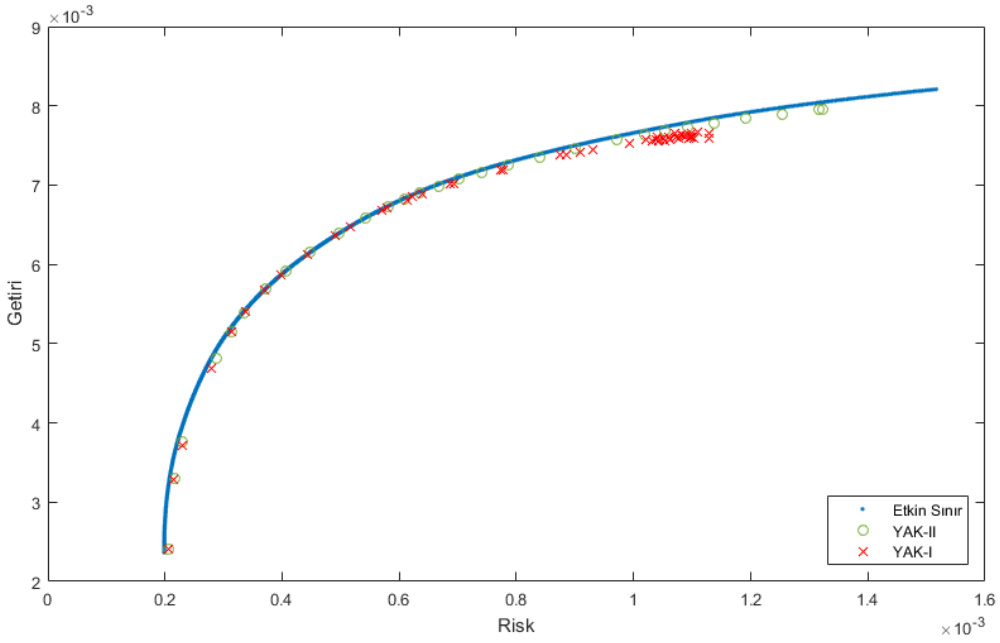
Δ performanstaki iyileştirmeyi temsil etmekte ve şu formüle göre hesaplanmaktadır: $\frac{(\text{YAK-II})-(\text{YAK}-\text{I})}{\text{YAK-I}}$



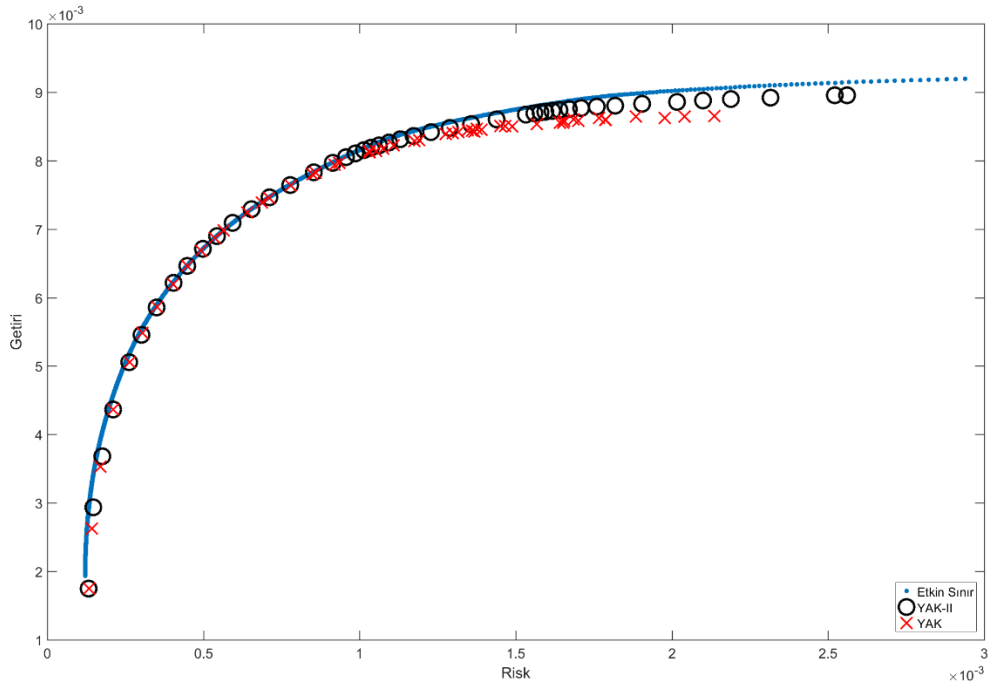
Şekil 4.19: Hang Seng veri seti için YAK-I ve YAK-II algoritmaları ile çizilen etkin sınırlar



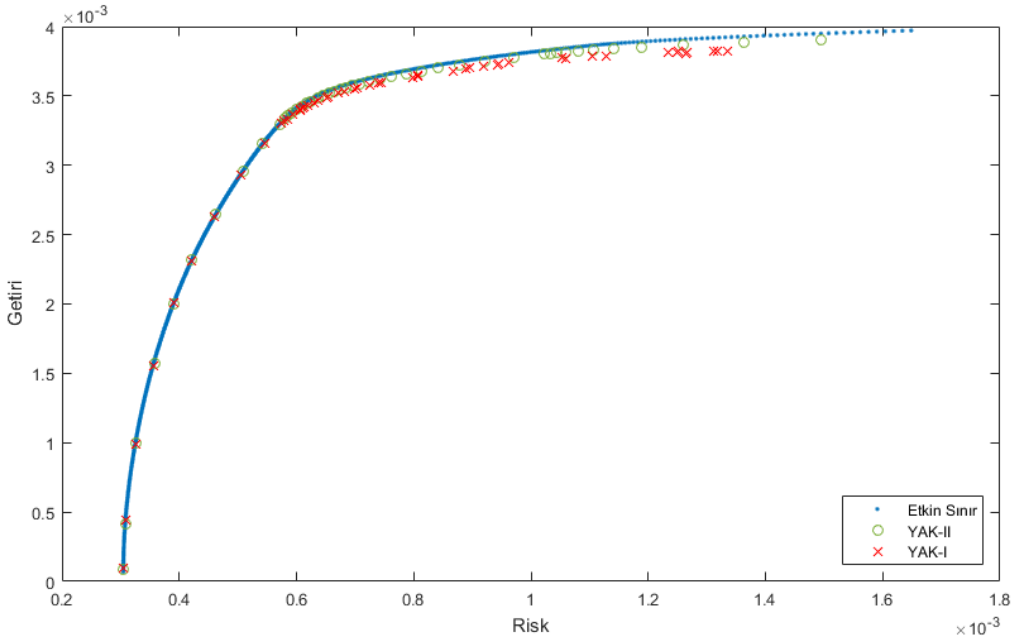
Şekil 4.20: DAX 100 veri seti için YAK-I ve YAK-II algoritmaları ile çizilen etkin sınırlar



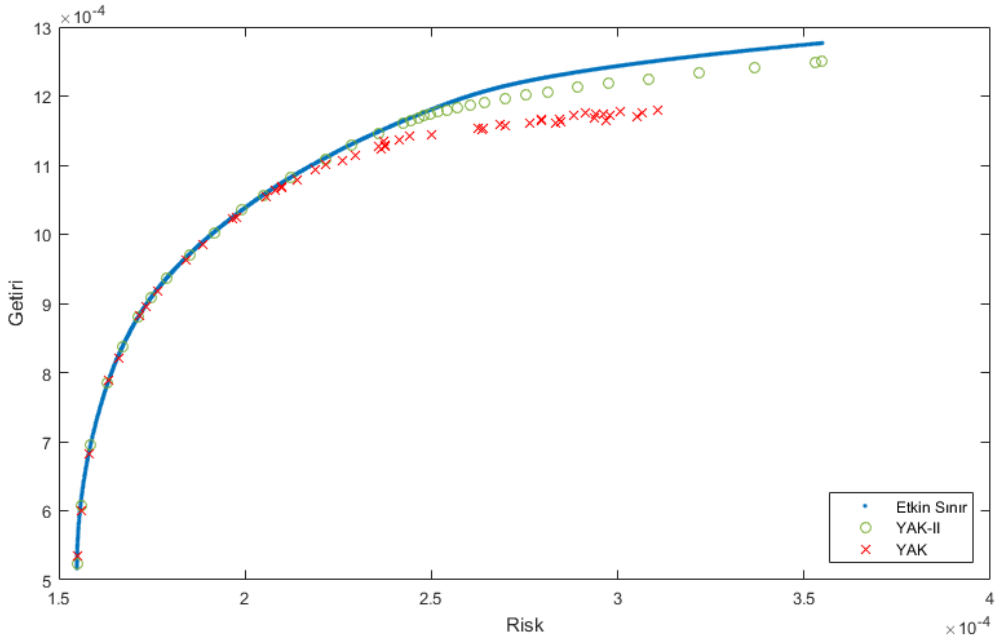
Şekil 4.21: FTSE 100 veri seti için YAK-I ve YAK-II algoritmaları ile çizilen etkin sınırlar



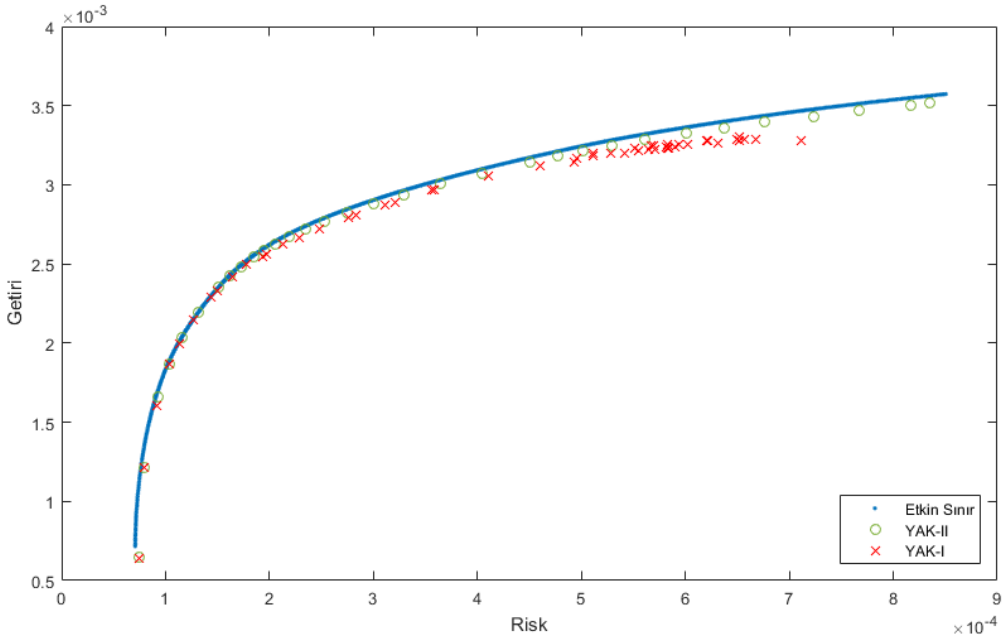
Şekil 4.22: S&P 100 veri seti için YAK-I ve YAK-II algoritmaları ile çizilen etkin sınırlar



Şekil 4.23: Nikkei veri seti için YAK-I ve YAK-II algoritmaları ile çizilen etkin sınırlar



Şekil 4.24: BİST 30 veri seti için YAK-I ve YAK-II algoritmaları ile çizilen etkin sınırlar



Şekil 4.25: BİST 100 veri seti için YAK-I ve YAK-II algoritmaları ile çizilen etkin sınırlar

4.3.2 YAK-II Algoritmasının Litertürdeki Diğer Tekniklerle Karşılaştırmalı Sonuçları

Literatürde fakrlı çalışmalarda farklı hata ölçütleri kullanıldığından dolayı, Bölüm 3.2'de sınıflandırılan hata ölçütleri kullanılarak YAK-II algoritmasının performansı diğer literatürdeki diğer algoritmalarla kıyaslanmıştır. Karşılaştırmalı sonuçlar ilgili çalışmalara göre ayrılmış ve Tablo 4.13-4.15'de verilmiştir. Tablo 4.13 YAK-II algoritmasının Chang ve diğ. (2000) (GA, TA ve BT algoritmaları), Deng ve diğ. (2012b) (PSO algoritması), Lwin ve Qu (2013) (PBILDE) ve Baykasoğlu ve diğ. (2015) (GRASP-QP) teknikleriyle karşılaştırmalı sonuçlarını sunmaktadır. Tablo 4.14 YAK-II algoritmasının Fernández ve Gómez (2007) (yapay sinir ağı (YSA)) ve Baykasoğlu ve diğ. (2015) ile ve Tablo 4.15 Cura (2009b) (PSO), Mozafari ve diğ. (2011) (PSO ve BT birleşimi bir yaklaşım), Sadigh ve diğ. (2012) (PSO ve Hopfield yapay sinir ağı birleşimi bir yaklaşım), Baykasoğlu ve diğ. (2015) ve Tuba ve Bacanin (2014b) (HFA ile hibritleştirilmiş bir YAK algoritması) ile kaşılaştırmalı sonuçlarını vermektedir.

Tablo 4.13: YHO, YHMED, YHMİN, YHMAKS hataları için performans karşılaştırması

İndeks	N	Performans ölçütü	GA (Chang ve diğ. 2000)	TS (Chang ve diğ. 2000)	SA (Chang ve diğ. 2000)	PSO (Deng ve diğ. 2012b)	PBILDE (Lwin ve Qu 2013)	GRASP-QP (Baykasoğlu ve diğ. 2016; Baykasoğlu ve diğ. 2015)	YAK-II
Hang Seng	31	YHO	1.0974	1.1217	1.0957	1.0953	1.1431	1.0965	**1.0953**
		YHMED	1.2181	1.2181	1.2181	-	1.2390	1.2155	1.2181
		YHMİN	-	-	-	-	-	**0.0000**	**0.0000**
		YHMAKS	-	-	-	-	-	**1.5538**	**1.5538**
		Süre(s)	172	74	79	-	-	62	56
DAX 100	85	YHO	2.5424	**3.3049**	2.9297	2.5417	2.4251	2.3126	2.3295
		YHMED	**2.5466**	2.6380	2.5661	-	2.5866	2.5630	2.5678
		YHMİN	-	-	-	-	-	0.0059	0.0023
		YHMAKS	-	-	-	-	-	4.0275	4.0275
		Süre(s)	544	199	210	-	-	139	135
FTSE 100	89	YHO	1.1076	1.6080	1.4623	1.0628	0.9706	0.8451	0.8481
		YHMED	1.0841	1.0841	1.0841	-	1.0840	1.0841	1.0841
		YHMİN	-	-	-	-	-	**0.0016**	0.0047
		YHMAKS	-	-	-	-	-	**2.0576**	2.0638
		Süre(s)	573	246	215	-	-	143	109
S&P 100	98	YHO	1.9328	3.3092	3.0696	1.6890	1.6386	1.2937	1.2930
		YHMED	1.2244	1.2882	1.1823	-	1.1692	1.1420	**1.1369**
		YHMİN	-	-	-	-	-	0.0009	**0.0000**
		YHMAKS	-	-	-	-	-	5.4551	5.4422
		Süre(s)	638	225	242	-	-	172	160
Nikkei	225	YHO	0.7961	0.8975	0.6066	0.6870	0.5972	0.5782	**0.5781**
		YHMED	0.6133	0.6093	0.6732	-	0.5896	0.5857	0.5856
		YHMİN	-	-	-	-	-	**0.0000**	**0.0000**
		YHMAKS	-	-	-	-	-	**1.1606**	**1.1606**
		Süre(s)	1964	545	553	-	-	438	410
Ortalama		YHO	1.4952	2.0483	1.8328	1.4152	1.3549	**1.2252**	1.2280
		YHMED	1.3373	1.3675	1.3448	-	1.3337	**1.3181**	1.3185
		YHMİN	-	-	-	-	-	0.0017	0.0014
		YHMAKS	-	-	-	-	-	2.8509	**2.8495**
		Süre(s)	778	257	260	-	-	191	174

Tablo 4.14: GHV-I, OGH-I hata performansı karşılaştırması

İndeks	N	Performans Ölçütü	YSA (Fernández ve Gómez 2007)	GRASP-QP (Baykasoğlu ve diğ. 2016; Baykasoğlu ve diğ. 2015)	YAK-II
Hang Seng	31	GHV-I	**4.1039**	4.1341	4.1302
		OGH-I	1.4530	1.1380	**1.1364**
		Süre(s)	390	62	56
DAX 100	85	GHV-I	**12.5914**	32.7815	32.6157
		OGH-I	**2.2060**	3.0504	3.0743
		Süre(s)	1069	139	135
FTSE 100	89	GHV-I	**4.4663**	4.5594	4.5658
		OGH-I	1.9636	1.5188	**1.4328**
		Süre(s)	1106	143	109
S&P 100	98	GHV-I	**8.3811**	12.0795	11.5133
		OGH-I	2.6816	**1.5599**	2.3389
		Süre(s)	1211	172	160
Nikkei	225	GHV-I	6.5924	**5.5753**	5.5757
		OGH-I	3.1050	**1.6595**	1.6764
		Süre(s)	2827	438	410
Ortalama		GHV-I	**7.2270**	11.8260	11.6801
		OGH-I	2.2818	**1.7853**	1.9358
		Süre(s)	1321	191	174

Tablo 4.15: OÖU, GHV-II, OGH-II hata performansı karşılaştırması

Indeks	N	Performans ölçütü	PSO (Cura 2009b)	PSO-SA (Mozafari ve diğ. 2011)	PSO-HNN (Sadigh ve diğ. 2012)	GRASP-QP (Baykasoğlu ve diğ. 2015)	ABC-FA (Tuba ve Bacanin 2014b)	YAK-II
Hang Seng	31	OÖU	0.0049	0.0001	0.0001	0.0001	0.0004	0.0001
		GHV-II	2.2421	1.6388	2.5908	1.6400	**1.3952**	1.6432
		OGH-II	0.7427	0.6059	0.7335	0.6060	**0.5289**	0.6047
		Süre(s)	34	-	-	62	**12**	56
DAX 100	85	OÖU	0.0090	0.0001	**0.0000**	0.0001	0.0009	0.0001
		GHV-II	6.8588	6.7806	**5.7585**	6.7593	7.2649	6.7925
		OGH-II	1.5885	1.2770	**0.1466**	1.2769	1.3523	1.2761
		Süre(s)	179	-	-	139	62	135
FTSE 100	89	OÖU	0.0022	**0.0000**	**0.0000**	**0.0000**	0.0003	**0.0000**
		GHV-II	3.0596	2.4701	5.4141	**2.4350**	2.6721	2.4397
		OGH-II	0.3640	0.3247	**0.3095**	0.3245	0.3187	0.3255
		Süre(s)	190	-	-	143	76	109
S&P 100	98	OÖU	0.0052	0.0001	**0.0000**	0.0001	0.0001	0.0001
		GHV-II	3.9136	2.6281	5.1456	2.5211	3.7598	2.5260
		OGH-II	1.4040	0.7846	**0.2925**	0.9063	0.9529	0.8885
		Süre(s)	214	-	-	172	125	160
Nikkei	225	OÖU	0.0019	**0.0000**	**0.0000**	**0.0000**	**0.0000**	0.0000
		GHV-II	2.4274	0.9583	4.7779	**0.8359**	1.6982	0.8396
		OGH-II	0.7997	1.7090	0.7040	0.4184	0.6719	**0.4127**
		Süre(s)	919	-	-	438	**329**	410
Ortalama		OÖU	0.0046	0.0001	**0.0000**	0.0001	0.00034	0.0001
		GHV-II	3.7003	2.8952	4.7374	**2.8383**	3.35804	2.8482
		OGH-II	0.9798	0.9402	**0.4372**	0.7064	0.76494	0.7015
		Süre(s)	307	-	-	191	120.8	174

Sonuç olarak, önerilen YAK-II algoritması literatürdeki diğer tekniklerle yarışabilecek kadar iyi bir performans sergilemiştir. Her ne kadar GRASP-QP tekniği YAK-II algoritması ile benzer performans göstermiş olsa da, dikkat edilmelidirki GRASP-QP yaklaşımı MATLAB programındaki kuadratik programlama aracına dayanan bir çalışma prensibi ile kodlanmıştır. Dolayısıyla diğer platformlarda kodlanması pek mümkün değildir. Öte yandan önerilen YAK algoritması değişik programlama dillerinde kodlanmak için yeterince esnektir. Dahası, YAK algoritması istenen kısıtlamaların da karşılanması nedeniyle portföy optimizasyonuna büyük ölçüde katkıda bulunabilecek Ortolama-Varyans PO problemini daha az hata ile çözebilen, kodlaması kolay bir algoritmadır. Az sayıda kontrol parametresine sahip olmak ve parametrelere daha az duyarlı olmak, önerilen algoritmanın bir başka olumlu özelliğidir. Bu yaklaşım, diğer portföy optimizasyon varyantlarının yanı sıra NP-Zor problemleri sınıfındaki benzer kombinatoryal optimizasyon problemlerini çözmek için kolaylıkla uyarlanabilir.

5. SONUÇ VE ÖNERİLER

Bu tezde kısıtlı portföy optimizasyonu probleminin çözümü için yapay arı kolonisi temelli iki yaklaşım açıklanmıştır. Çözümlerin uygulanabilir olmaya zorlanması, tamir mekanizması, çözümlerin her zaman uygulanabilir alanda kalmasını sağlamasına rağmen, algoritmanın çözüm uzayında serbestçe dolaşmasını engelleyerek daha zayıf bir yakınsamaya sebep olabilir. Hesaplamalı sonuçlar göstermiştir ki, çözümlerin istenilen sınırlar arasında kalması için sadece tamir mekanizmasının uygulanlması etkin sınıra yaklaşmak için etkin değildir. Öte yandan, algoritmanın geliştirilmesinde yapılan ön çalışmalar, bir onarım mekanizmasından yararlanılmadan, uygun çözümlerin süratli bir şekilde elde edilmesinin mümkün olmadığını gösterdi. Bu nedenle, bu tezde kısmi olarak uygulanabilir alanda kalmaya zorlarken uygulanabilir olmayan çözümlere tolerans getiren bir yapay arı kolonisi algoritması, kısıtlı portföy optimizasyonu problemini çözmek için önerilmiştir. Uygulanabilir olmayan çözümlere tolerans getiren tamir prosedürü, çözümlerin istenilen alt ve üst limit kısıtlarını ihlal etmesine izin verirken, portföyde bulunan hisse sayısının arzulanan değerde sabit kalmasını garanti etmektedir. Hesaplamalı sonuçlar, bu tekniğin, tek başına çözümleri bütün kısıtlar için uygulanabilir alanda kalmaya zorlayan teknikten daha iyi çözümler elde ettiğini gösterdi. Algoritmanın performansı literatürdeki diğer tekniklerle rekabet edebelecek kadar iyidir. Bu algoritmayı diğer portföy optimizasyonu varyasyonlarına uygulamak, gelecek çalışmalar için uygun olabilir. Probleme işlem maliyeti gibi diğer kısıtların eklenmesi durumunda algoritmanın performansının nasıl etkileneceği ilginç bir araştırma konusu olabilir.

6. KAYNAKLAR

Anagnostopoulos, Konstantinos P., Mamanis, Georgios, "Multiobjective evolutionary algorithms for complex portfolio optimization problems", *Computational Management Science*, 8 (3), 259-279, (2011a).

Anagnostopoulos, Konstantinos P., Mamanis, Georgios, "The mean–variance cardinality constrained portfolio optimization problem: An experimental evaluation of five multiobjective evolutionary algorithms", *Expert Systems with Applications*, 38 (11), 14208-14217, (2011b).

Bacanin, N., Tuba, M., "Firefly algorithm for cardinality constrained mean-variance portfolio optimization problem with entropy diversity constraint", *Scientific World Journal*, 2014 (2014).

Bacanin, Nebojsa, Tuba, Milan, Pelevic, Branislav, "Constrained portfolio selection using artificial bee colony (ABC) algorithm", *International Journal of Mathematical Models and Methods in Applied Sciences*, 8, 190-198, (2014).

Baykasoğlu, Adil, Avci, Mualla Gonca, Burcin Özsoydan, F., "Erratum to "A GRASP based solution approach to solve cardinality constrained portfolio optimization problems" [Comput. Indus. Eng. 90 (2015) 339–351]", *Computers & Industrial Engineering*, 96, 249-250, (2016).

Baykasoğlu, Adil, Yunusoglu, Mualla Gonca, Burcin Özsoydan, F., "A GRASP based solution approach to solve cardinality constrained portfolio optimization problems", *Computers & Industrial Engineering*, 90, 339-351, (2015).

Beasley, J. E., "OR-Library", people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html, Alındığı tarih: 2016.

Beni, Gerardo, Wang, Jing, "Swarm Intelligence in Cellular Robotic Systems", (eds:Dario, P., Sandini, G., Aebischer, P.), *Robots and Biological Systems: Towards a New Bionics?*, Berlin, Heidelberg: Springer Berlin Heidelberg, 703-712, (1993).

Bienstock, D., "Computational study of a family of mixed-integer quadratic programming problems", *Mathematical Programming*, 74 (2), 121-140, (1996).

Bissiri, Yassiah, Dunbar, W Scott, Hall, Allan, "Swarm–based truck-shovel dispatching system in open pit mine operations", Ph. D. thesis, University of British Columbia, (2002).

Chang, T. J., Meade, N., Beasley, J. E., Sharaiha, Y. M., "Heuristics for cardinality constrained portfolio optimisation", *Computers & Operations Research*, 27 (13), 1271-1302, (2000).

Chang, Tun-Jen, Yang, Sang-Chin, Chang, Kuang-Jung, "Portfolio optimization problems in different risk measures using genetic algorithm", *Expert Systems with Applications*, 36 (7), 10529-10537, (2009).

Chen, Yun-Chia, Liang, Chia-Chien, Liu, "An artificial bee colony algorithm for the cardinality-constrained portfolio optimization problems", *Evolutionary Computation (CEC), 2012 IEEE Congress on*, 1-8, (2012).

Chen, A. H. L., Liang, Y. C., Liu, C. C., "Portfolio optimization using improved artificial bee colony approach", *Proceedings of the 2013 IEEE Conference on Computational Intelligence for Financial Engineering and Economics, CIFEr 2013 - 2013 IEEE Symposium Series on Computational Intelligence, SSCI 2013*, 60-67, (2013).

Chen, W., Zhang, R. T., Cai, Y. M., Xu, F. S., "Particle swarm optimization for constrained portfolio selection problems", *Proceedings of the 2006 International Conference on Machine Learning and Cybernetics*, 2006, 2425-2429, (2006).

Chen, W., Zhang, W. G., "The admissible portfolio selection problem with transaction costs and an improved PSO algorithm", *Physica A*, 389 (10), 2070-2076, (2010).

Chu, Shu-Chuan, Tsai, Pei-wei, Pan, Jeng-Shyang, "Cat Swarm Optimization", (eds:Yang, Q., Webb, G.), *PRICAI 2006: Trends in Artificial Intelligence: 9th Pacific Rim International Conference on Artificial Intelligence Guilin, China, August 7-11, 2006 Proceedings*, Berlin, Heidelberg: Springer Berlin Heidelberg, 854-858, (2006).

Corazza, M., Fasano, G., Gusso, R., "Particle Swarm Optimization with non-smooth penalty reformulation, for a complex portfolio selection problem", *Applied Mathematics and Computation*, 224 (0), 611-624, (2013).

Cura, T., "Particle swarm optimization approach to portfolio optimization", *Nonlinear Anal-Real*, 10 (4), 2396-2406, (2009a).

Cura, Tunchan, "Particle swarm optimization approach to portfolio optimization", *Nonlinear Analysis: Real World Applications*, 10 (4), 2396-2406, (2009b).

Deb, K., "An efficient constraint handling method for genetic algorithms", *Computer Methods in Applied Mechanics and Engineering*, 186 (2-4), 311-338, (2000a).

Deb, Kalyanmoy, "An efficient constraint handling method for genetic algorithms", *Computer Methods in Applied Mechanics and Engineering*, 186 (2–4), 311-338, (2000b).

Deng, G. F., Lin, W. T., Lo, C. C., "Markowitz-based portfolio selection with cardinality constraints using improved particle swarm optimization", *Expert Systems with Applications*, 39 (4), 4558-4566, (2012a).

Deng, Guang-Feng, Lin, Woo-Tsong, "Ant Colony Optimization for Markowitz Mean-Variance Portfolio Model", (eds:Panigrahi, B. K., Das, S., Suganthan, P. N., Dash, S. S.), *Swarm, Evolutionary, and Memetic Computing: First International Conference on Swarm, Evolutionary, and Memetic Computing, SEMCCO 2010, Chennai, India, December 16-18, 2010. Proceedings*, Berlin, Heidelberg: Springer Berlin Heidelberg, 238-245, (2010).

Deng, Guang-Feng, Lin, Woo-Tsong, Lo, Chih-Chung, "Markowitz-based portfolio selection with cardinality constraints using improved particle swarm optimization", *Expert Systems with Applications*, 39 (4), 4558-4566, (2012b).

Dorigo, M., Gambardella, L. M., "Ant colony system: a cooperative learning approach to the traveling salesman problem", *IEEE Transactions on Evolutionary Computation*, 1 (1), 53-66, (1997).

Dorigo, M., Maniezzo, V., Colorni, A., "Ant system: optimization by a colony of cooperating agents", *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 26 (1), 29-41, (1996a).

Dorigo, M., Maniezzo, V., Colorni, A., "Ant system: optimization by a colony of cooperating agents", *IEEE Trans Syst Man Cybern B Cybern*, 26 (1), 29-41, (1996b).

Ertenlice, O., Kalayci, C. B., "A survey of swarm intelligence for portfolio optimization: Algorithms and applications", *Swarm and Evolutionary Computation*, 39, 36-52, (2018).

Fernandez, A., Gomez, S., "Portfolio selection using neural networks", *Computers & Operations Research*, 34 (4), 1177-1191, (2007).

Fernández, Alberto, Gómez, Sergio, "Portfolio selection using neural networks", *Computers & Operations Research*, 34 (4), 1177-1191, (2007).

Ge, M., "Artificial bee colony algorithm for portfolio optimization", *5th International Conference on Intelligent Control and Information Processing, ICICIP 2014 - Proceedings*, 449-453, (2015).

Golmakani, H. R., Fazel, M., "Constrained Portfolio Selection using Particle Swarm Optimization", *Expert Systems with Applications*, 38 (7), 8327-8335, (2011).

Hong-mei, Liu, Zhuo-fu, Wang, Hui-min, Li, "Artificial bee colony algorithm for real estate portfolio optimization based on risk preference coefficient", *Management Science and Engineering (ICMSE), 2010 International Conference on*, 1682-1687, (2010).

Jorion, Philippe, *Value at risk: the new benchmark for controlling market risk*, Irwin Professional Pub., (1997).

Kalayci, C. B., Ertenlice, O., Akyer, H., Aygoren, H., "An artificial bee colony algorithm with feasibility enforcement and infeasibility toleration procedures for cardinality constrained portfolio optimization", *Expert Systems with Applications*, 85 (Supplement C), 61-75, (2017).

Kao, Y. C., Cheng, H. T., "Bacterial Foraging Optimization Approach to Portfolio Optimization", *Computational Economics*, 42 (4), 453-470, (2013).

Karaboga. (2005a). An idea based on honey bee swarm for numerical optimization: Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.

Karaboga, D., Akay, B., "A modified Artificial Bee Colony (ABC) algorithm for constrained optimization problems", *Applied Soft Computing*, 11 (3), 3021-3031, (2011).

Karaboga, Dervis. (2005b). An idea based on Honey Bee Swarm for Numerical Optimization. Kayseri: Erciyes University, Engineering Faculty, Computer Engineering Department.

Karimkashi, S., Kishk, A. A., "Invasive Weed Optimization and its Features in Electromagnetics", *IEEE Transactions on Antennas and Propagation*, 58 (4), 1269-1278, (2010).

Kennedy, J., Eberhart, R., "Particle swarm optimization", *Neural Networks, 1995. Proceedings., IEEE International Conference on*, 4, 1942-1948 vol.1944, (1995).

Khalidji, M., Zeiaee, M., Taei, A., Jahed-Motlagh, M. R., Khaloozadeh, H., "Dynamically weighted continuous ant colony optimization for biobjective portfolio selection using value-at-risk", *Proceedings - 2009 3rd Asia International Conference on Modelling and Simulation, AMS 2009*, 230-235, (2009).

Konno, Yamazaki, "Mean-Absolute Deviation Portfolio Optimization Model and Its Applications to Tokyo Stock-Market", *Management Science*, 37 (5), 519-531, (1991a).

Konno, Hiroshi, Yamazaki, Hiroaki, "Mean-Absolute Deviation Portfolio Optimization Model and Its Applications to Tokyo Stock Market", *Management Science*, 37 (5), 519-531, (1991b).

Koshino, M., Murata, H., Kimura, H., "Improved particle swarm optimization and application to portfolio selection", *Electron Comm Jpn 3*, 90 (3), 13-25, (2007).

Kumar, D., Mishra, K. K., "Portfolio optimization using novel co-variance guided Artificial Bee Colony algorithm", *Swarm and Evolutionary Computation*, 33, 119-130, (2017).

Liu, R., Wang, X., "Using elitist particle swarm optimization to facilitate real estate portfolio based on information entropy", *Proceedings of International Conference on Risk Management and Engineering Management*, 633-638, (2008).

Lwin, K., Qu, R., "A hybrid algorithm for constrained portfolio selection problems", *Applied Intelligence*, 39 (2), 251-266, (2013).

Lwin, Khin, Qu, Rong, Kendall, Graham, "A learning-guided multi-objective evolutionary algorithm for constrained portfolio optimization", *Applied Soft Computing*, 24 (0), 757-772, (2014).

Magill, Michael JP, Constantinides, George M, "Portfolio selection with transactions costs", *Journal of Economic Theory*, 13 (2), 245-263, (1976).

Markowitz, Harry, "Portfolio Selection", *The Journal of Finance*, 7 (1), 77-91, (1952a).

Markowitz, Harry M., "Portfolio Selection", *The Journal of Finance*, 7 (1), 77-91, (1952b).

Markowitz, Harry M., *Portfolio Selection: Efficient Diversification of Investments*, New York: Yale University Press, (1959).

Mashayekhi, Zahra, Omrani, Hashem, "An integrated multi-objective Markowitz–DEA cross-efficiency model with fuzzy returns for portfolio selection problem", *Applied Soft Computing*, 38, 1-9, (2016).

Mishra, S. K., Panda, G., Majhi, R., "Constrained portfolio asset selection using multiobjective bacteria foraging optimization", *Operational Research*, 14 (1), 113-145, (2014a).

Mishra, S. K., Panda, G., Majhi, R., "A comparative performance assessment of a set of multiobjective algorithms for constrained portfolio assets selection", *Swarm and Evolutionary Computation*, 16, 38-51, (2014b).

Moral-Escudero, R., Ruiz-Torrubiano, R., Suarez, A., "Selection of Optimal Investment Portfolios with Cardinality Constraints", *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, 2382-2388, (2006).

Mozafari, M., Tafazzoli, S., Jolai, F., "A new IPSO-SA approach for cardinality constrained portfolio optimization", *International Journal of Industrial Engineering Computations*, 2 (2), 249-262, (2011).

Niu, B., Fan, Y., Xiao, H., Xue, B., "Bacterial foraging based approaches to portfolio optimization with liquidity risk", *Neurocomputing*, 98 (0), 90-100, (2012).

Niu, Ben, Xue, Bing, Li, Li, Chai, Yujuan, "Symbiotic Multi-swarm PSO for Portfolio Optimization", (eds:Huang, D.-S., Jo, K.-H., Lee, H.-H., Kang, H.-J., Bevilacqua, V.), *Emerging Intelligent Computing Technology and Applications. With Aspects of Artificial Intelligence: 5th International Conference on Intelligent Computing, ICIC 2009 Ulsan, South Korea, September 16-19, 2009 Proceedings*, Berlin, Heidelberg: Springer Berlin Heidelberg, 776-784, (2009).

Pai, G. A. V., Michel, T., "Evolutionary Optimization of Constrained k-means Clustered Assets for Diversification in Small Portfolios", *Ieee Transactions on Evolutionary Computation*, 13 (5), 1030-1053, (2009).

Passino, K. M., "Biomimicry of bacterial foraging for distributed optimization and control", *IEEE Control Systems*, 22 (3), 52-67, (2002a).

Passino, K. M., "Biomimicry of bacterial foraging for distributed optimization and control", *Ieee Contr Syst Mag*, 22 (3), 52-67, (2002b).

Rockafellar, R Tyrrell, Uryasev, Stanislav, "Optimization of conditional value-at-risk", *Journal of risk*, 2, 21-42, (2000).

Sadigh, A. N., Mokhtari, H., Iranpoor, M., Fatemi Ghomi, S. M. T., "Cardinality constrained portfolio optimization using a hybrid approach based on particle swarm optimization and hopfield neural network", *Advanced Science Letters*, 17 (1), 11-20, (2012).

Samuelson, P., "The Fundamental Approximation Theorem of Portfolio Analysis in Terms of Means Variances and Higher Moments", *Review of Economic Studies*, 25, 65-86, (1958).

Simaan, Y., "Estimation risk in portfolio selection: The mean variance model versus the mean absolute deviation model", *Management Science*, 43 (10), 1437-1446, (1997).

Socha, Krzysztof, Dorigo, Marco, "Ant colony optimization for continuous domains", *Eur J Oper Res*, 185 (3), 1155-1173, (2008).

Speranza, M., "A heuristic algorithm for a portfolio optimization model applied to the Milan stock market", *Computers & Operations Research*, 23 (5), 433-441, (1996).

Sun, J., Fang, W., Wu, X. J., Lai, C. H., Xu, W. B., "Solving the multi-stage portfolio optimization problem with a novel particle swarm optimization", *Expert Systems with Applications*, 38 (6), 6727-6735, (2011).

Suthiwong, Dit, Sodanil, Maleerat, "Cardinality-constrained Portfolio optimization using an improved quick Artificial Bee Colony Algorithm", *Computer Science and Engineering Conference (ICSEC), 2016 International*, 1-4, (2016).

Tan, Zhu, "Fireworks Algorithm for Optimization", (eds:Tan, Y., Shi, Y., Tan, K. C.), *Advances in Swarm Intelligence: First International Conference, ICSI 2010, Beijing, China, June 12-15, 2010, Proceedings, Part I*, Berlin, Heidelberg: Springer Berlin Heidelberg, 355-364, (2010).

Tan, L., Niu, B., Lin, F., Duan, Q., Li, L., "Modified bacterial foraging optimization for constrained portfolio optimization", *Information Technology Journal*, 12 (23), 7918-7921, (2013).

Tuba, M., Bacanin, N., "Upgraded firefly algorithm for portfolio optimization problem", *Proceedings - UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, UKSim 2014*, 113-118, (2014a).

Tuba, M., Bacanin, N., "Artificial bee colony algorithm hybridized with firefly algorithm for cardinality constrained mean-variance portfolio selection problem", *Appl. Math. Inf. Sci*, 8 (6), 2831-2844, (2014b).

Wang, Wenjun, Wang, Hui, Wu, Zhijian, Dai, Hubei, "A Simple and Fast Particle Swarm Optimization and Its Application on Portfolio Selection", *Intelligent Systems and Applications, 2009. ISA 2009. International Workshop on*, 1-4, (2009).

Wang, Zhen, Liu, Sanyang, Kong, Xiangyu, "Artificial Bee Colony Algorithm for Portfolio Optimization Problems", *International Journal of Advancements in Computing Technology*, 4 (4), 8-16, (2012).

Woodside-Oriakhi, M., Lucas, C., Beasley, J. E., "Heuristic algorithms for the cardinality constrained efficient frontier", *Eur J Oper Res*, 213 (3), 538-550, (2011).

Xu, Fasheng, Chen, Wei, "Stochastic portfolio selection based on velocity limited particle swarm optimization", *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*, 1, 3599-3603, (2006).

Yaakob, S. B., Watada, J., "A hybrid particle swarm optimization approach to mixed integer quadratic programming for portfolio selection problems", *International Journal of Simulation: Systems, Science and Technology*, 11 (5), 68-74, (2010).

Yang, Xin-She, "Firefly Algorithms for Multimodal Optimization", (eds:Watanabe, O., Zeugmann, T.), *Stochastic Algorithms: Foundations and Applications*, 5792, Springer Berlin Heidelberg, 169-178, (2009).

Yang, Xin-She, "A New Metaheuristic Bat-Inspired Algorithm", (eds:González, J. R., Pelta, D. A., Cruz, C., Terrazas, G., Krasnogor, N.), *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, 284, Berlin, Heidelberg: Springer Berlin Heidelberg, 65-74, (2010a).

Yang, Xin-She, "Firefly Algorithm", *Engineering Optimization*, John Wiley & Sons, Inc., 221-230, (2010b).

Yin, X., Ni, Q., Zhai, Y., "A novel PSO for portfolio optimization based on heterogeneous multiple population strategy", *2015 IEEE Congress on Evolutionary Computation, CEC 2015 - Proceedings*, 1196-1203, (2015).

Young, Martin R., "A Minimax Portfolio Selection Rule with Linear Programming Solution", *Management Science*, 44 (5), 673-683, (1998).

Zhu, H. H., Wang, Y., Wang, K. S., Chen, Y., "Particle Swarm Optimization (PSO) for the constrained portfolio optimization problem", *Expert Systems with Applications*, 38 (8), 10161-10169, (2011).

Zhu, Hanhong, Chen, Yun, Wang, Kesheng, "Swarm Intelligence Algorithms for Portfolio Optimization", (eds:Tan, Y., Shi, Y., Tan, K. C.), *Advances in Swarm Intelligence: First International Conference, ICSI 2010, Beijing, China, June 12-15, 2010, Proceedings, Part I*, 6145, Berlin, Heidelberg: Springer Berlin Heidelberg, 306-313, (2010).

7. ÖZGEÇMİŞ

Adı Soyadı	:	Ökkeş Ertenlice
Doğum Yeri ve Tarihi	:	Gaziantep 02.03.1991
Lisans Üniversite	:	Pamukkale Üniversitesi
Elektronik posta	:	okkesertenlice@gmail.com
İletişim Adresi	:	Kayaönü Mah. Kayaönü Sok. NO:15/A 27500 Gaziantep

Yayın Listesi :

• Kalayci, C. B., Ertenlice, O., Akyer, H., & Aygoren, H. (2017). An artificial bee colony algorithm with feasibility enforcement and infeasibility toleration procedures for cardinality constrained portfolio optimization. Expert Systems with Applications, 85, 61-75.

• Kalayci, C. B., Ertenlice, O., Akyer, H., & Aygoren, H. (2017). A review on the current applications of genetic algorithms in mean-variance portfolio optimization Ortalama-varyans portföy optimizasyonunda genetik algoritma uygulamaları üzerine bir literatür araştırması: Pamukkale Journel of Enginering Science

• Ertenlice, O., & Kalayci, C. B. (2018). A survey of swarm intelligence for portfolio optimization: Algorithms and applications. Swarm and evolutionary computation, 39, 36-52.