
GAZİ ÜNİVERSİTESİ
Gazi
1926

ROBOT TUTUCULARIN OPTİMUM TASARIMI İÇİN ÇOK AMAÇLI HİBRİT BİR YÖNTEM ÖNERİSİ

Ayşenur AVDER

YÜKSEK LİSANS TEZİ
BİLİŞİM SİSTEMLERİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

EKİM 2019

Ayşenur AVDER tarafından hazırlanan "Robot Tutucuların Optimum Tasarımı İçin Çok Amaçlı Hibrit Bir Yöntem Önerisi" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Bilişim Sistemleri Anabilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Doç. Dr. İsmail ŞAHİN

Endüstriyel Tasarım Mühendisliği Anabilim Dalı, Gazi Üniversitesi

Bu tezin kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum .....

Başkan: Doç. Dr. Nursal ARICI

Bilgisayar Mühendisliği Anabilim Dalı, Gazi Üniversitesi

Bu tezin kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum .....

Üye: Dr. Öğr. Üyesi Ümit ATİLA

Bilgisayar Mühendisliği Anabilim Dalı, Karabük Üniversitesi

Bu tezin kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum .....

Tez Savunma Tarihi: 04/10/2019

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

.....

Doç. Dr. Aslıhan TÜFEKCİ

Bilişim Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Bilişim Enstitüsü Tez Yazım Kuralları'na uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçlarını bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu çalışmada sunduğum tezin özgün olduğunu

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarımı kabullendiğimi beyan ederim.

Ayşenur AVDER

04/10/2019

ROBOT TUTUCULARIN OPTİMUM TASARIMI İÇİN ÇOK AMAÇLI HİBRİT BİR YÖNTEM ÖNERİSİ

(Yüksek Lisans Tezi)

Ayşenur AVDER
GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

Ekim 2019

ÖZET

Son yıllardaki teknolojik gelişmelerle birlikte makine elemanlarının tasarımında optimizasyon tekniklerinin kullanımı giderek yaygınlaşmaktadır. Robot tutucular, günümüzde endüstriyel sistemlerde nesneleri kavrama, taşıma veya sabitleme amacıyla kullanılan araçlardır. Robot tutucuların endüstrideki yaygın kullanımı iyi bir tasarımın önemini arttırmaktadır. Bir robot tutucunun tasarımında tasarım bileşenlerinin ve sistemin çeşitli özellikleri, geliştirilmek istenen uygulamanın özellikleri ve amacı dikkate alınmaktadır. Özellikle robotik sistemlerde robot tutucuların, ele aldıkları nesneye zarar vermeden en az manevrayla kavrayabilmesi önemli bir konudur. Bundan dolayı robot tutucuların tasarım optimizasyonu son yıllarda popüler araştırma alanlarından birisi haline gelmiştir. Bu çalışmada, üç farklı robot tutucu konfigürasyonu üzerinde çok amaçlı hibrit bir optimizasyon yöntemi önerilmiştir. Hibrit çok amaçlı optimizasyon yöntemi olarak üç farklı hibrit algoritma kullanılmıştır. Bunlar, hibrit NSGAII-SPEA2, hibrit NSGAII-PESA2 ve hibrit NSGAII-MOEA/D algoritmalarıdır. Elde edilen optimizasyon sonuçları, literatürde aynı probleme farklı çözümler getiren bazı çalışmalarla karşılaştırılmıştır. Yapılan değerlendirmeler sonucunda geliştirilen hibrit yaklaşımın başarımının belirlenen konfigürasyonlar üzerinde daha yüksek olduğu görülmüştür. Gelecekte robot tutucuların tasarım problemine yönelik olarak yapılacak optimizasyon çalışmalarında iyi bir çözüm yöntemi olarak kullanılabileceği düşünülmektedir.

Bilim Kodu: 92424
Anahtar Kelimeler: Robot tutucular, çok amaçlı optimizasyon, mekanik dizayn
Sayfa Adedi: 77
Danışman: Doç. Dr. İsmail ŞAHİN

THE PROPOSAL OF MULTI-OBJECTIVE HYBRID METHOD FOR OPTIMUM DESIGN OF ROBOT GRIPPERS

(M. Sc. Thesis)

Ayşenur AVDER

GAZI UNIVERSITY

INFORMATICS INSTITUTE

October 2019

ABSTRACT

With the technological developments in recent years, the use of optimization techniques in the design of machine elements is becoming increasingly widespread. Robot grippers are tools used to grasp, transport or fix objects in industrial systems. The widespread use of robot grippers in the industry increases the importance of a good design. In the design of a robot gripper, the various features of the design components and the system, the characteristics and purpose of the application to be developed are considered. Particularly in robotic systems, it is an important issue that robot holders can grasp at least maneuver without damaging the object they are dealing with. Design optimization of robot grippers has become one of the popular research areas in recent years. In this study, a multi-objective hybrid optimization method has been proposed on three different robot gripper configurations. Three different hybrid algorithms were used as hybrid multi-objective optimization method. These are hybrid NSGAII-SPEA2, hybrid NSGAII-PESA2 and hybrid NSGAII-MOEAD algorithms. The results of the optimization are compared with some studies in the literature that bring different solutions to the same problem. As a result of the evaluations, the performance of the hybrid approach developed was higher than the configurations determined. In the future, it is thought that can be used as a good solution method in the optimization studies to be done for the design problem of robot grippers.

Science Code : 92424

Key Words : Robot grippers, multi-objective optimization, mechanic design

Page Number : 77

Supervisor : Assoc. Prof. İsmail ŞAHİN

TEŞEKKÜR

Tez çalışmam boyunca değerli yardım ve desteklerini esirgemeyen danışmanım Doç. Dr. İsmail ŞAHİN'e, çalışmamda değerli yardım ve katkılarıyla beni yönlendiren Dr. Murat DÖRTERLER'e ve her zaman beni anlayışla karşılayan, çalışmalarım süresince desteklerini ve katkılarını benden esirgemeyen sevgili aileme teşekkürlerimi sunarım.

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış bazı simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklama**
f	Amaç fonksiyonu
G	Kısıt fonksiyonu
Y_g	Tutucu uçlarının maksimum yer değiştirme miktarı
Y_{max}	Kavrama nesnesinin maksimum boyutu
Y_{min}	Kavrayıcı nesnenin asgari boyutu
Z_{max}	Tutucu işleticisinin maksimum yer değiştirmesi

Kısaltmalar	**Açıklama**
ACO	Karınca koloni algoritması (Ant colony optimization)
FFGA	Fonseca ve Fleming'in çok amaçlı evrimsel algoritması
HLGA	Haleja ve Lin'in ağırlıklı toplam temelli yaklaşımı
IGD	Ters kuşak mesafesi (Inverted generational distance)
MODE	Çok amaçlı farksal gelişim (Multi-objective differential evolution)
MOEA/D	Ayrıştırmaya dayalı çok amaçlı evrimsel algoritma
MOGA	Çok amaçlı genetik algoritma (Multi-objective genetic algorithm)
NPGA	Hücrelendirilmiş pareto genetik algoritması
NSGA	Baskılanamayan sıralamalı genetik algoritma
NSGA-II	Seçkin baskılanamayan sıralamalı genetik algoritma

Kısaltmalar	Açıklama
PESA2	Pareto zarflama temelli seçim algoritması
SPEA2	Güçlü pareto evrimsel algoritması
TLBO	Öğretme-öğrenme tabanlı optimizasyon algoritması
VEGA	Vektör hesaplamalı genetik algoritma

1. GİRİŞ

Mühendislik tasarımı, belirlenen ihtiyaçların karşılanması amacıyla bir sistem, bileşen veya bir sürecin ortaya konması olarak tanımlanabilir. Mühendislik sistemlerinin tasarımı, çeşitli mühendislik alanlarından gelen tasarımcılar arasında işbirliği gerektiren disiplinler arası bir süreçtir. Deneylerle analiz ve doğrulamaya tabi tutulabilecek modeller geliştirmek için varsayımlarda bulunulmaktadır [1].

Optimizasyon, verilen bir amaç ya da amaçlar doğrultusunda belirli kısıtların sağlanarak en uygun çözümlerin elde edilme sürecidir. Gerçek dünyada karşılaşılan tasarım problemlerinin birçoğu, çok sayıda amacın aynı anda çözülmesini gerektirdiğinden dolayı bu tip problemler çok amaçlı optimizasyon problemi olarak değerlendirilmektedir. Çok amaçlı optimizasyonda birbirleri ile çelişen amaçlar içerisinde en uygun tasarım çözümlerini bulmak için rasyonel bir yaklaşım sunulmaktadır. Evrimsel algoritmalar ise doğadaki evrim stratejisini temel alarak oluşturulmuş algoritmalardır ve çok amaçlı optimizasyon yöntemleri ile birlikte kullanımından dolayı çok amaçlı evrimsel algoritmalar olarak tanımlanmaktadır. Çok amaçlı evrimsel algoritmalar, özellikle son yıllarda karmaşık optimizasyon problemlerinin çözümünde iyi bir alternatif çözüm olarak değerlendirilmektedir.

Robot tutucu mekanizmaları, günümüz endüstriyel sistemlerinde yaygın olarak kullanılan mekanizmalardır. Robot uygulamalarında uç eleman olarak kullanılan robot tutucular, montaj ve kaynak işlemleri, parçaların taşınması, yüksek sıcaklık kaynakları ile radyoaktif maddelerin tutulması gibi pek çok alanda kullanılmaktadır.

Endüstriyel sistemlerde optimizasyon çalışmalarının öneminin artmasına paralel olarak robotik araştırma alanlarında da bir nesnenin kavranması olayı önemli bir araştırma konusu haline gelmiştir. Özellikle robot tutucuların, nesneye herhangi bir zarar vermeden tutup kavrayabilmesi problemi önemli robot tasarım optimizasyonu problemleri arasında yer almaktadır. Bu problem için optimum tasarımı bulmak üzere literatürde pek çok farklı çalışma yapılmıştır. Robot tutucuların tasarım problemi ilk olarak Osyczka [2] tarafından doğrusal olmayan bir optimizasyon problemi olarak formülize edilmiştir. Daha sonra Osyzcka ve Krenich [3] tarafından bu problem çok amaçlı bir optimizasyon problemi olarak ele alınarak problem üzerinde genetik algoritma uygulanmıştır. Saravanan ve

diğerleri [4] yaptıkları çalışmada üç farklı robot tutucu konfigürasyonu belirlemişlerdir ve her bir konfigürasyon için üç farklı yöntemi (MOGA, NSGA-II ve MODE) karşılaştırmalı olarak uygulamışlardır. Datta ve Deb [5] ise iki farklı robot tutucu konfigürasyonu belirlemişlerdir ve her bir konfigürasyon üzerinde NSGA-II algoritmasını uygulamışlardır.

Bu tez çalışmasında üç farklı robot tutucu konfigürasyonu üzerinde çok amaçlı hibrit bir optimizasyon çalışması gerçekleştirilmiştir. Belirlenen tasarım konfigürasyonlarına üç farklı hibrit optimizasyon algoritması uygulanmıştır. Bu hibrit algoritmalar şunlardır: hibrit NSGAII-SPEA2, hibrit NSGAII-PESA2 ve hibrit NSGAII-MOEAD. Referans olarak Osyczka ve Krenich [3] yaptıkları çalışma dikkate alınmıştır ve burada tanımlanan robot tutucu konfigürasyonları, amaç fonksiyonları ve kısıtları kullanılmıştır. Robot tutucu tasarım problemine ilk kez uygulanan bu çok amaçlı hibrit optimizasyon çalışmasının sonuçlarının yapılan diğer çalışmaların sonuçlarına kıyasla daha iyi olduğu gözlemlenmiştir. Bu tez çalışmasının literatüre katkısı, robot tutucu tasarım problemine yönelik olarak daha önce uygulanmamış çok amaçlı hibrit optimizasyon yönteminin, bundan sonraki robot tasarım problemlerinin çözümünde alternatif bir yöntem olarak kullanılabilecek olmasıdır.

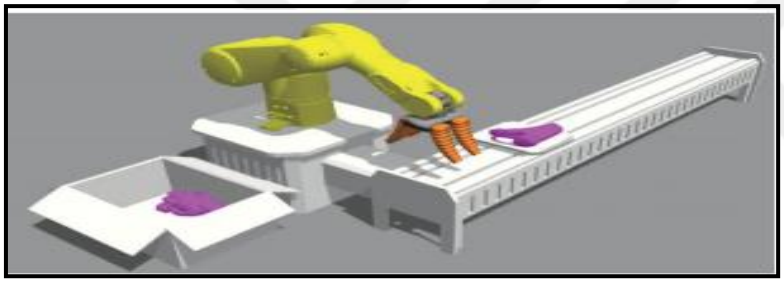
Bu tezin ilerleyen kısımları şu şekildedir. İkinci bölümde robot tutucular ve temel özellikleri, robot tasarım optimizasyonuna yönelik olarak literatürde yer alan çalışmalar ve robot tutucuların tasarım optimizasyonu problemi açıklanmıştır. Üçüncü bölümde tez çalışmasında kullanılan algoritmalar ayrıntılı bir şekilde sunulmuştur. Dördüncü bölümde tez çalışması kapsamında yapılan deneysel çalışmalar ve sonuçları literatürdeki benzer çalışmalarla karşılaştırmalı bir şekilde sunulmuştur ve uygulanan hibrit algoritmaların problem üzerindeki performansı değerlendirilmiştir. Son bölümde ise tez çalışmasının değerlendirilmesi ve gelecek çalışmalar için öneriler yapılmıştır.

2. LİTERATÜR TARAMASI

2.1. Robot Tutucular

Tutucular, kavranacak nesne ile bu nesneyi kavrayacak taşıma aracı arasındaki bağlantıyı sağlayan alt sistemlerdir. Nesnenin taşıma aracına göre konumunun belirlenmesi ve değiştirilmesinde kullanılırlar [6].

Robotik teknolojilerde tutucular, en fazla çeşitliliğe sahip fonksiyonel birimlere aittir. Bunun nedeni, robotların esnek makineler olmalarına rağmen tutucuların çok daha spesifik görevleri gerçekleştirmeleridir. Çok sayıda farklı gereksinimler ile iyi uyarlanmış ve güvenilir sistemler üretme arzusu robot tutucu tasarımının önemi gittikçe arttırmaktadır [6].



Şekil 2.1. Robotik kavrama sisteminin temel yapılandırması [7]

Robot uygulamalarında uç eleman olarak kullanılan tutucular montaj ve kaynak işlemleri, parçaların taşınması, yüksek sıcaklık kaynakları ile radyoaktif maddelerin tutulması, bombaların patlatılması ve mayınlar ile gemi enkazlarının araştırılması gibi pek çok alanda yaygın olarak kullanılmaktadır.

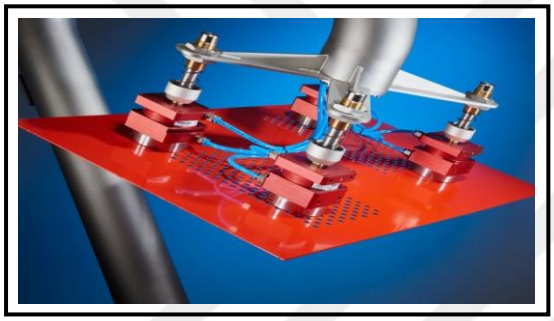
2.1.1. Robot tutucuların temel özellikleri

Tutucular iş parçalarını kavrayabilir, konumunu değiştirebilir ve yönlendirebilirler. Ayrıca herhangi bir parçanın sistemde olup olmadığını belirten algılayıcıları içerebilmektedirler. Yapı olarak bir tutucu mümkün olduğunca hafif olmalıdır, çünkü bir makinenin maksimum taşıma yükü bir iş parçasının ağırlığını içermektedir. Ağır yükler tutuculara ve robotlarda yüksek anları önlemek için mümkün olduğunca tutma eksenine yakın tutulmalıdır. Maliyet etkinliği için de bir tutucu basit bir şekilde tasarlanmış ve üretilmiş, bakımı kolay ve güvenilir olmalıdır [8].

2.1.2. Robot tutucuların sınıflandırılması

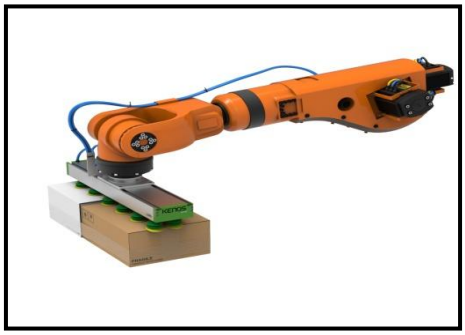
Robot tutucular, tek yüzeyli tutucular, sıkıştırma tutucuları ve esnek tutucular olmak üzere üçe ayrılmaktadır.

Tek yüzeyli tutucular, başka yollarla taşınması zor ve ağır olan tek boyutlu bileşenlerin taşınmasında kullanılırlar. Robotik tutucular için daha yaygın olan itme kuvveti yerine çekme kuvvetini kullanarak bileşenleri tutmaktadırlar [9]. Manyetik, vakum ve yapışkan tutucular olmak üzere 3'e ayrılırlar. Manyetik tutucular, sadece metal içeren nesneleri toplamak için kullanılırlar ve toplama ile bırakma işlemleri sırasında kontrol edilmeleri kolaydır [6].



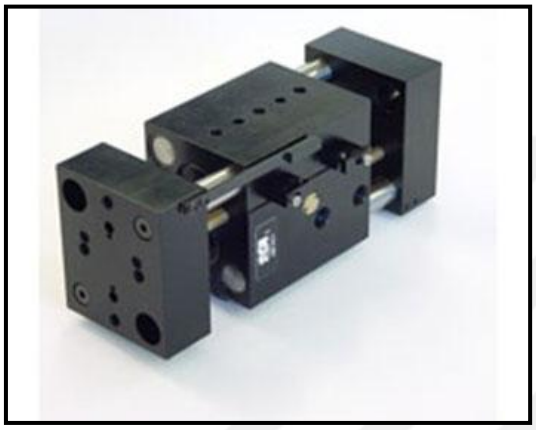
Şekil 2.2. Tek yüzeyli tutuculardan manyetik tutucu örneği [10]

Vakum tutucular, vantuz şeklindedirler ve vantuz kısımları lastikten yapılmıştır. Vantuzlar yani emme kapları maddeleri toplamak için düşük basınçlı cihazlar ile tüplere bağlanır ve serbest bırakılmaları için vantuzlar içine hava pompalanmaktadır [11]. Yapışkan tutucular ise yapışkan maddeler ve kumaş gibi esnek malzemelerin taşınmasında kullanılırlar.



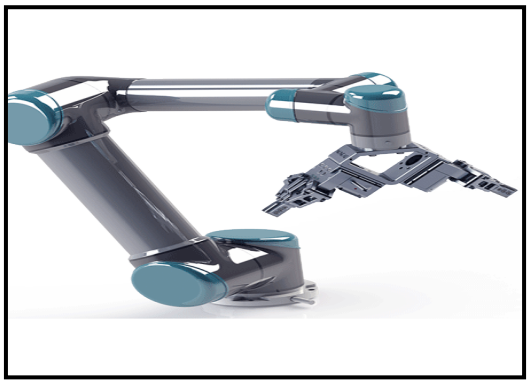
Şekil 2.3. Tek yüzeyli tutuculardan vakum tutucu örneği [12]

Sıkıştırma tutucuları, nispeten daha basit bir dizayna sahiptirler ve bu nedenle daha düşük maliyetlidirler. Nesne yüzeylerinden birden fazlasına dahili veya harici basınç uygulayarak toplanan nesneye baskı uygulamaktadırlar. Bu tür tutucular, kavrama işlemi için pnömatik ve hidrolik teknik olmak üzere iki teknik kullanmaktadırlar. Büyük kuvvetlere ihtiyaç duymayan küçük nesneler için pnömatik teknik kullanılır. Büyük kuvvet gerektiren ağır nesneler için ise hidrolik teknik kullanılır. Pnömatik teknik, düşük fiyat, düşük ağırlık ve kullanım kolaylığı açısından daha yaygındırlar [9].



Şekil 2.4. Pnömatik teknik kullanan bir sıkıştırma tutucusu [13]

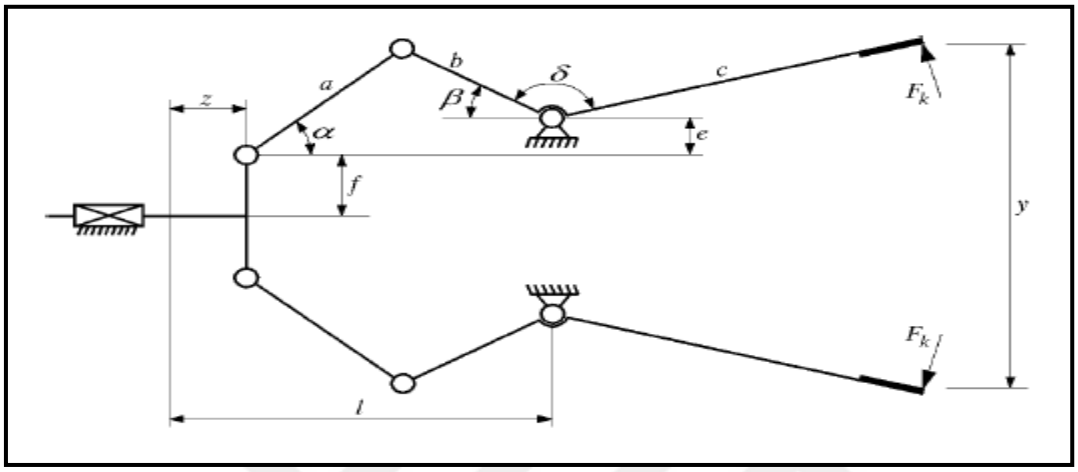
Esnek tutucular ise, her bir parmak üzerinde birkaç bağlantıdan oluşan iki veya daha fazla parmaktan oluşmaktadırlar. Bir dizi farklı ürünü işlemeye girerler. Bu türden çeşitli tutucular çeşitli araştırmalarla üretilmiştir. Esnek tutuculardan olan çok parmaklı tutucular, iki parmaktan daha fazlasına sahip bir insan eline benzer kavrama şekline sahiptir. Bu tür bir kavrayıcı, parmakların tek tek kontrol edilebildiği bağlantılardan dolayı çok karmaşık şekillerle nesneleri kavrayabilirler [9].



Şekil 2.5. Örnek bir esnek tutucu [14]

2.2. Robot Tutucu Tasarım Optimizasyonu

Tez çalışması kapsamında Osyzcka ve Krenich'in [3] sunmuş olduğu robot tutucu modeli kullanılmıştır.



Şekil 2.6. Robot tutucu tasarımının şeması [3]

Bu çalışmada Osyzcka ve Krenich [3] tarafından sunulan ilk üç robot tutucu konfigürasyonu üzerinde çalışılmıştır. Farklı konfigürasyonlar kullanılmasının amacı, ikiden fazla kritere sahip robot tutucu modelleri için karar verme sürecinin özellikle Pareto-optimal çözüm seti çok büyük olduğunda zor bir iş olmasıdır. Bir robot tutucu tasarım probleminin, bu şekilde farklı konfigürasyonlar halinde ele alınması ile söz konusu problemi daha kolay çözmek hedeflenmektedir. Bu çalışmada kullanılan konfigürasyonların her biri iki farklı amaç fonksiyonu içermektedir.

Birinci konfigürasyonda, tutucu tarafından uygulanan en büyük ve en küçük kuvvetler arasındaki farkın minumum olması ile tutucu işleticisinin (gripper actuator)uyguladığı tutucu hareket kuvveti (P) ile tutucu uçlarının uyguladığı kuvvetin birbirine oranının minumum olması hedeflenmektedir.

İkinci konfigürasyonda, tutucu işleticisi (gripper actuator) ile tutucu uçları (gripper ends) arasındaki kaydırma iletim oranı (shift transmission ratio) minumum olması ile tutucu işleticisinin (gripper actuator) uyguladığı tutucu hareket kuvveti (P) ile tutucu uçlarının uyguladığı kuvvetin birbirine oranının minumum olması amaçlanmıştır.

Üçüncü konfigürasyonda ise ikinci konfigürasyonda olduğu gibi tutucu işleticisi ile tutucu uçları arasındaki kaydırma iletim oranını minumum hale getirmek ve tutucuların tüm elemanlarının uzunluklarının minumum olması hedeflenmektedir.

2.2.1. Birinci konfigürasyon

Bu modelde iki amacın optimizasyonu hedeflenmiştir:
1.Herhangi bir robot tutucu probleminin en önemli noktası, tutulacak nesne üzerinde sabit bir sıkı tutuşu sağlamaktır. Kavrama kuvvetinin maksimum ve minimum değerleri arasındaki farkın en aza indirilmesi, kavrama gücünün minimum dalgalanmasına yol açacaktır. Bu nedenden dolayı birinci amaç fonksiyonu Eş. 2.1'de gösterildiği gibi tutucu uçları arasındaki belli bir mesafe için tutucu tarafından uygulanan maksimum ve minumum kuvvetlerin farkının minumum olması üzerine oluşturulmuştur.

$$f_1(\mathrm{x}) = \min(|max_z F_k(\mathrm{x},\mathrm{z}) - min_z F_k(\mathrm{x},\mathrm{z})|) \tag{2.1}$$

2.Robot tutucu problemlerinin bir diğer önemli noktası da belli bir çalıştırma kuvveti (actuating force) için elde edilen kavrama kuvvetidir. Minumum kavrama kuvveti (gripping force) değeri ne kadar büyük olursa mekanizma o kadar verimli olmaktadır. Bundan dolayı Eş. 2.2'de gösterilen ikinci amaç fonksiyonu, tutucu işleticisinin (gripper actuator)uyguladığı tutucu hareket kuvveti (P) ile tutucu uçlarının uyguladığı kuvvetin birbirine oranı minumum olacak şekilde oluşturulmuştur.

$$f_2(\mathrm{x}) = \min\left(\frac{P}{min_z F_k(x,z)}\right) \tag{2.2}$$

Tasarımda yedi adet tasarım değişkeni bulunmaktadır. Bu tasarım değişkenleri ve değer aralıkları Eş. 2.3-2.9'da gösterilmiştir.

$$10.0 \leq a \leq 250.0 \tag{2.3}$$
$$10.0 \leq b \leq 250.0 \tag{2.4}$$
$$100.0 \leq c \leq 300.0 \tag{2.5}$$
$$0.0 \leq e \leq 50.0 \tag{2.6}$$
$$10.0 \leq f \leq 250.0 \tag{2.7}$$
$$100.0 \leq l \leq 300.0 \tag{2.8}$$

$$1.0 \leq \delta \leq \pi \tag{2.9}$$

Tasarım değişkenlerinden *a,b,c,e,f* ve *l* tutucu üzerindeki bağlantıların uzunluklarını temsil etmektedir (mm). δ ise b ve c elementleri arasındaki bağlantı açısıdır (rad).

Tasarımda beş adet de geometrik parametre kullanılmıştır. Bu parametrelerden Y_{min}, kavrayıcı nesnenin asgari boyutu (minimal dimension of gripping object), Y_{max} kavrama nesnesinin maksimum boyutu (maximal dimension of gripping object), Y_g tutucu uçlarının maksimum yer değiştirme miktarı (maximal range of the gripper ends displacement), Z_{max} tutucu işleticisinin maksimum yer değiştirmesi (maximal displacement of the gripper actuator) ve *P* tutucu hareket kuvveti (actuating force of the gripper) olarak tanımlanmaktadır. Bu parametrelerin değerleri: Y_{min}*=50mm;* Y_{max}*= 100mm*; Y_g*=150mm*; Z_{max}*=50mm; P=100N dir.*

Tasarımda kullanılan bazı matematiksel hesaplamalar Eş. 2.10-2.15'te gösterilmektedir:

$$g^2 = (l - z)^2 + e^2 \tag{2.10}$$

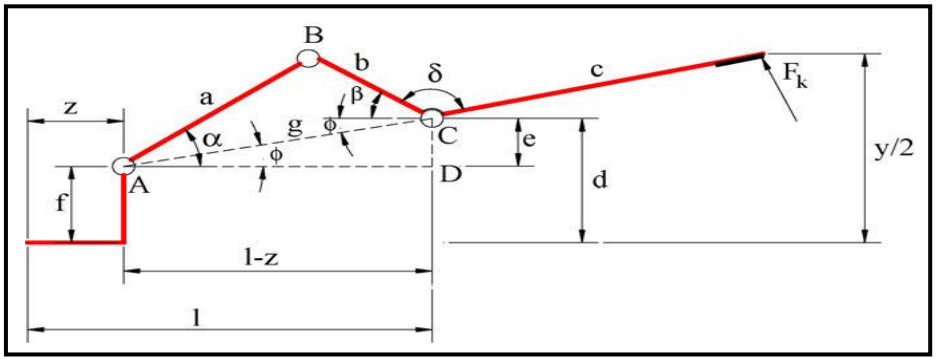
$$\alpha = arccos\left(\frac{a^2+g^2-b^2}{2ag}\right) + \theta \tag{2.11}$$

$$\beta = arccos\left(\frac{b^2+g^2-a^2}{2bg}\right) - \theta \tag{2.12}$$

$$\theta = arctan\left(\frac{e}{l-z}\right) \tag{2.13}$$

$$F_k = \frac{Pb\ sin\ (\alpha+\beta)}{2c\ cos\ (\alpha)} \tag{2.14}$$

$$y(x,z) = 2[e + f + c\sin(\beta + \delta)] \tag{2.15}$$



Şekil 2.7. Robot tutucu mekanizmasının geometrik bağlantıları [4]

Tasarımda yedi adet de kısıt kullanılmıştır ve bu kısıtlara dair eşitlikler Eş. 2.16-2.22'de gösterildiği gibidir:

$g_1(x): Y_{min} - y(x, Z_{max}) \geq 0$ (2.16)

$g_2(x): y(x, Z_{max}) \geq 0$ (2.17)

$g_3(x): y(x, 0) - Y_{max} \geq 0$ (2.18)

$g_4(x): Y_g - y(x, 0) - \geq 0$ (2.19)

$g_5(x): (a + b)^2 - l^2 - e^2 \geq 0$ (2.20)

$g_6(x): (l - Z_{max})^2 + (a - e)^2 - b^2 \geq 0$ (2.21)

$g_7(x): l - Z_{max} \geq 0$ (2.22)

2.2.2. İkinci konfigürasyon

Bu modelde iki amacın optimizasyonu hedeflenmiştir:

1.Tutucu problemlerinde sabit bir kavrama durumunun sağlanabilmesi için tutucu işleticisi (gripper actuator) ile tutucu uçları (gripper ends) arasındaki kaydırma iletim oranı (shift transmission ratio) minumum olmalıdır. Bundan dolayı ikinci konfigürasyon için ilk amaç fonksiyonu Eş. 2.23'teki gibi tanımlanmıştır.

$$f_1(x) = min\left(\left|\frac{y(x,Zmax) - y(x,Zmin)}{Zmax - Zmin}\right|\right) \quad (2.23)$$

2.Minumum kavrama değeri ne kadar büyük olursa mekanizma o kadar verimli olmaktadır. Bu durumu sağlayabilmek için ikinci amaç fonksiyonu, tutucu işleticisinin (gripper actuator)uyguladığı tutucu hareket kuvveti (P) ile tutucu uçlarının uyguladığı kuvvetin birbirine oranı minumum olacak şekilde Eş. 2.24'te belirlenmiştir.

$$f_2(x) = min\left(\frac{P}{\min_z F_k(x,z)}\right) \quad (2.24)$$

Bu konfigürasyonda, birinci konfigürasyondaki kısıtlara ek olarak, maksimum ve minumum kavrama kuvvetleri arasındaki farkın F_{1g}=4 değerinden küçük olmaması gerektiği varsayılarak Eş. 2.25'te gösterilen kısıtlama tanımlanmıştır:

$$g_8(x): F_{1g} - [\max_z F_k(x,z) - \min_z F_k(x,z)] \geq 0 \quad (2.25)$$

$F_{1g,}$ birinci amaç fonksiyonunun değerinin üst limiti olarak varsayılmaktadır.

2.2.3. Üçüncü konfigürasyon

Bu konfigürasyonda aşağıdaki amaçların optimizasyonu hedeflenmiştir:
1.Robot tutucu tasarımında, sabit bir kavrama durumunun sağlanabilmesi için tutucu işleticisi (gripper actuator) ile tutucu uçları (gripper ends) arasındaki kaydırma iletim oranı (shift transmission ratio) minumum olmalıdır. Bu nedenden dolayı, birinci amaç fonksiyonu Eş. 2.26'daki gibi tanımlanmıştır:

$$f_1(x)= min\left(\left|\frac{y(x,Zmax)-y(x,Zmin)}{Zmax-Zmin}\right|\right) \quad (2.26)$$

2.Tutucunun tüm elemanlarının uzunlukları Eş. 2.27'de gösterildiği gibi minumum olmalıdır.

$$f_2(x)=min(a+b+c+e+f+l) \quad (2.27)$$

Bu konfigürasyonda, kuvvet aktarım oranının varsayılan alt sınır olan F_{2d}=0.5 değeri ile bir kısıtlayıcı olarak değerlendirileceği varsayılmaktadır. Bundan dolayı, ikinci konfigürasyondaki kısıtlara ek olarak Eş. 2.28'de gösterilen bir kısıt daha eklenmiştir.

$$g_{13}(x)= \frac{P}{\min z F_k(x,z)} - F_{2d} \geq 0 \quad (2.28)$$

2.2.4. Dördüncü konfigürasyon

Bu modelde iki amacın optimizasyonu hedeflenmiştir:
1.Tutucunun tüm elemanlarının uzunluklarını tanımlayan fonksiyon Eş. 2.29'da gösterilmektedir.

$$f_1(x)= a+b+c+e+f+l \quad (2.29)$$

2.Tutucu eklemlerindeki maksimum kuvveti tanımlayan fonksiyon Eş. 2.30'da gösterilmektedir.

$f_2(x)=\max_j\{R_j\}$ (2.30)

Bu konfigürasyonda, kaydırma iletim oranının varsayılan alt sınır olan $F_{3d}=1.5$ değeri ile bir kısıtlayıcı olarak değerlendirileceği varsayılmıştır. Dolayısıyla, üçüncü konfigürasyondaki kısıtlara ek olarak Eş. 2.31'de gösterilen bir kısıt daha eklenmiştir.

$$g_{14}(x)= \left|\frac{y(x,Zmax)-y(x,Zmin)}{Zmax-Zmin}\right| - F_{3d} \geq 0 \quad (2.31)$$

2.2.5. Beşinci konfigürasyon

Bu modelde iki amacın optimizasyonu hedeflenmiştir:
1.Tutucu eklemlerindeki maksimum kuvveti tanımlayan fonksiyon Eş. 2.32'de gösterilmiştir.

$f_1(x)=\max_j\{R_j\}$ (2.32)

2.Tutucu mekanizmasının verimliliğini tanımlayan fonksiyon Eş. 2.33'teki gibi tanımlanmıştır.

$$f_2(x)=\sum_{z=0}^{Zmax}[{F_k}^{BT}(x,z) - {F_k}^{T}(x,z)] \quad (2.33)$$

2.3. Önceki Çalışmalar

Bu bölümde robot tutucuların tasarım problemi üzerine yapılan optimizasyon çalışmaları araştırılmıştır.

Robot tutucuların optimizasyonu üzerine yapılan ilk çalışma Cutkosky'nin [15], tutuculardaki kavramanın seçimi, modeli ve tasarımı üzerine yapmış olduğu çalışmadır. Cutkosky çalışmasında, kavrama sorunlarını çözmek için bir uzman sistem geliştirmiştir.

Osyczka ve Krenich [3] çok amaçlı tasarım optimizasyon tabanlı bir robot tutucu tasarımı önermişlerdir. Çalışmalarında farklı robot tutucu konfigürasyonları üzerinde evrimsel

algoritmalardan genetik algoritmayı kullanmışlardır. Çalışmanın sonucunda bilinen en iyi değerlere sahip bir tasarım elde etmişlerdir.

Zitar [16], katı nesnelerin minumum kuvvet değerleri ile tutuşunu sağlamak amacıyla tek amaçlı bir optimizasyon üzerinde çalışmışlardır. Çalışmalarında yöntem olarak karınca koloni algoritması (ant colony optimization-ACO) kullanılmıştır. Çalışmanın sonucunda karınca koloni algoritması ile sert nesneleri kavramak için optimum parmak kuvvet değerleri bulunmuş ve aynı zamanda karınca koloni algoritmasının probleme uygulanabilirliği için en uygun parametre setinin her lokasyondaki karınca sayısından bağımsız olduğunu tespit etmişlerdir.

Lanni ve Ceccarelli [17], yaptıkları çalışmada iki parmaklı tutucuların yakalama mekanizmaları için optimum bir tasarım üzerine çalışmışlardır. Çalışmanın temel amacı, mevcut çözümlerin iyileştirilmesi de dahil olmak üzere optimum özelliklere sahip bir tutucu mekanizmasının kinematik tasarımını gerçekleştirmektir. Çalışmada kavrama indeksi, mekanizma yerleşimi, hızlanma miktarı ve hızı dikkate alan dört adet amaç fonksiyonu kullanılmış ve bu amaç fonksiyonlarına dayanan önerdikleri yeni bir matematiksel model aracılığı ile optimizasyon yapılmıştır. Elde edilen yeni tasarım mekanizmasının, literatürdeki benzer örneklerle karşılaştırıldığında daha iyi yapı ve işletim özelliklerini gösterdiği tespit edilmiştir.

Saravanan ve diğerleri [4] yaptıkları çalışmada, üç farklı robot tutucu konfigürasyonu belirlemişlerdir ve her bir konfigürasyonun tasarım değişkeni vektörü için üç farklı yöntem (MOGA,NSGA-II,MODE algoritmaları) denemişlerdir. Algoritmalar farklı amaç fonksiyonlarının oluşturduğu üç farklı konfigürasyon üzerinde uygulanmıştır ve beş amaç fonksiyonu, yedi tasarım değişkeni ve dokuz kısıt fonksiyonuna sahip bu problem için pareto optimal çözümleri elde etmek amaçlanmıştır. Çalışmanın sonucunda, kullanılan üç algoritmadan NSGA-II algoritmasının bu optimizasyon probleminde diğerlerine göre daha başarılı olduğu tespit edilmiştir.

Li ve diğerleri [18], robot tutucuların tasarım problemini çok amaçlı bir optimizasyon problemi olarak ele almışlardır. Çalışmalarında robot tutucuların bağlantı uzunluklarını ve eklem açılarını optimize etmek için robot tutuculardaki uç tutma mekanizmasını ve çok amaçlı optimizasyon algoritmalarından da NSGA-II algoritmasını kullanmışlardır. Tasarım

değişkenlerini, ağır tutucuların geometrik boyutlarına göre tanımlamışlardır ve amaç fonksiyonlarını kavrama kuvvetlerine ve eklemler arasındaki kuvvet iletim ilişkilerine göre belirlemişlerdir. Kısıtlamalar ise, fiziksel koşullar ve kavrayıcıların yapısı tarafından sağlanmıştır. Çalışmalarının sonucunda, bilinen en iyi boyutsal değerlere sahip küçük ve büyük ölçekli tutucu tasarımları elde etmişlerdir.

Rao ve diğerleri [19], yaptıkları çalışmada aralarında robot tutucuların da olduğu çeşitli mekanik tasarım problemlerinin optimizasyonu için "Öğretme-Öğrenme Tabanlı Optimizasyon (TLBO)" olarak adlandırdıkları yeni bir optimizasyon yöntemini önermişlerdir. Çalışmada tek amaçlı optimizasyon yöntemi kullanılmıştır. TLBO yönteminin etkinliği, en iyi çözüm, ortalama çözüm, yakınsama oranı ve hesaplama çabasına dayalı olarak diğer popülasyon tabanlı optimizasyon algoritmaları ile karşılaştırılmıştır. Önerilen TLBO yönteminin, belirlenen mekanik tasarım optimizasyon problemleri için diğer optimizasyon yöntemlerinden daha etkili ve verimli olduğu sonucuna varılmıştır.

Datta ve Deb [5], robot tutucuların tasarım problemine yönelik olarak çok amaçlı bir optimizasyon çalışması gerçekleştirmişlerdir. Çalışmada robot tutucuların tasarım problemi, iki adet amaç fonksiyonu ve bir takım kısıtlamaları içeren iki farklı konfigürasyon ile çözülmeye çalışılmıştır. İki farklı konfigürasyon, NSGA-II algoritması ile tasarlanmış ve elde edilen sonuçlar önceki bir çalışma ile karşılaştırılmıştır. Elde edilen sonuçlara göre, her iki tutucu konfigürasyonu için, önerilen metodoloji önceki çalışmanın sonuçlarını geride bırakmaktadır. Ayrıca Pareto-optimal çözümler, amaç fonksiyonları ve değişken değerler arasında anlamlı bir ilişki kurmak için kapsamlı bir şekilde incelenmiştir. Ek olarak, tutucu konfigürasyonlarından birinin diğerini, her iki amaç fonksiyonu açısından tamamen geride bıraktığı ve böylece uygulamadaki konfigürasyonlardan birinin kullanımına yönelik bir sonuca ulaşıldığı görülmektedir.

Pahwa ve diğerleri [20], boyutsal varyasyonun robot tutucuların performansına yönelik etkisini araştırmak için geometrik parametrelerin optimizasyonu üzerinde çalışmışlardır. Problem simülasyon yazılımlarından COMSOL kullanılarak tasarlanmış ve simüle edilmiştir. Çalışmanın sonucunda elektrotermal mikro tutucuların performansının boyutsal değişimlerden büyük ölçüde etkilendiği tespit edilmiştir.

Dao ve Huang [21] yaptıkları çalışmada, nesneleri kavramak için iki esnek elemanlı bir tutucunun çok amaçlı optimizasyon ile optimum tasarımı üzerine çalışmışlardır. Çalışmada bulanık mantık tabanlı Fuzzy-Taguchi yöntemi kullanmışlardır. Girdi olarak yatay ve dikey kuvveti temsil eden iki adet parametre kullanmışlardır. Bu parametreler burulma yayının torku ve gerilmesi olarak iki farklı amaç fonksiyonu ile formüle edilmişlerdir. Çalışmanın sonucunda, belirlenen tutucu mekanizması için bilinen en iyi değerlere sahip yatay ve dikey kuvvet parametre değerleri bulunmuştur.

Ciocarlie ve diğerleri [22] parmak ucu ve zarf kavraması yapan bir robot tutucunun optimum tasarımı üzerine çalışmışlardır. Çalışmada aktif tendonun rotası ve pasif uzatma kuvvetleri sağlayan yayların parametreleri optimize edilmiştir. Çalışmanın sonucunda optimize edilmiş tendonların boyutları ve harekete geçirme parametrelerinin kombinasyonu ile bilinen en iyi tasarıma ulaşmışlardır.

Avder ve diğerleri [23], robot tutucu uçlarının nesneye uyguladığı kuvvetin dalgalanma miktarı ile tutucu işleticisi ve tutucu uçları arasındaki güç aktarım oranını optimize etmek için çok amaçlı bir optimizasyon çalışması gerçekleştirmiştir. Çalışmada yöntem olarak SPEA2 algoritması kullanılmıştır. Çalışmanın sonucunda elde edilen sonuçlar önceki bazı çalışmalarla karşılaştırılmıştır ve SPEA2 algoritması ile daha üstün sonuçlar elde edildiği görülmüştür.

3. KULLANILAN YÖNTEMLER VE METODOLOJİ

Bu bölümde robot tutucu tasarım probleminin çözümünde kullanılan çok amaçlı optimizasyon algoritmalarından NSGA-II (Elitist Non-dominated Sorting Genetic Algorithm), MOEA-D (Multi-objective Evolutionary Algorithm Based on Decomposition), PESA (Pareto Envelope-based Selection) ve SPEA2 (Strength Pareto Evolutionary Algorithm) algoritmaları açıklanmıştır.

3.1. Çok Amaçlı Optimizasyon

Birden fazla amaç fonksiyonunu sistemli ve eş zamanlı olarak optimize etme işlemine "çok amaçlı optimizasyon" denir [24]. Çok amaçlı optimizasyon problemlerinde eş zamanlı olarak birden fazla amaç optimize edilmeye çalışılmaktadır ve optimize edilmek istenen bu amaçlar birbirleriyle çelişebilmektedir. Ana hedef, çelişen amaç fonksiyonlarının varlığında en uygun tasarım çözümlerini bulmak için rasyonel bir yaklaşım sağlamaktır [25].

Genel olarak, çok amaçlı bir optimizasyon problemi, minumum değere indirgenmesi veya maksimum değere ulaştırılması hedeflenen birden fazla amaç fonksiyonunu içermektedir. Çok amaçlı optimizasyon, Eş. 3.1-3.4'teki gibi tanımlanmaktadır:

min/max: $y = f(x) = (f_1(x), f_2(x), \dots\dots\dots\dots f_k(x))$ (3.1)

koşul: $e(x) = (e_1(x), e_2(x), \dots\dots\dots\dots\dots e_m(x))$ (3.2)

$x = (x_1, x_2, \dots\dots\dots\dots x_n) \in X$; karar vektörü (3.3)

$y = (y_1, y_2, \dots\dots\dots\dots y_k) \in Y$; amaç vektörü (3.4)

$X, karar\ uzayı$

$Y, amaç\ uzayı$

Kısıtlar, *e(x)* ise $e(x) \leq 0$ koşulu altında tanımlanan problemler için erişilebilir çözümler kümesini gösterir [26].

3.1.1. Çok amaçlı optimizasyon terminolojisi

Tanım-1-(Erişebilir Küme-Feasible Set)

X_f erişebilir kümesi, *e(x)* kısıt fonksiyonlarının koşullarını sağlayan *x* karar vektörlerinden oluşmaktadır ve Eş. 3.5'teki gibi tanımlanmaktadır:

$$X_f = \{x \in X \mid e(x) \leq 0\} \tag{3.5}$$

X_f, amaç uzayı içinde erişebilir bölge ise Y_f, Eş. 3.6'daki gibi tanımlanmıştır:

$$Y_f = f(X_f) = Ux \in X_f\{f(x)\} \tag{3.6}$$

Genellik kaybı olmadan, burada maksimizasyon problemi olduğu varsayılır. Minimizasyon veya karmaşık maksimizasyon-minimizasyon problemleri için de yapılan bu tanım benzerdir [26].

Tanım-2-(Dominantlık-Pareto Dominance)

Karar uzayı içerisinde *a* ve *b* karar vektörleri olduğu varsayılırsa dominantlık tanımı şu şekilde yapılır:

$a \prec b$ *a, b*'ye göre dominanttır. Bu durumda *f(a) < f(b)* 'dir.

$a \preceq b$ *a, b*'nin zayıf dominantıdır. Bu durumda $f(a) \precneqq f(b)$'dir.

$a \sim b$ *a, b*'ye eşittir. Bu durumda $f_1(a) \precneqq f_2(b)$ ve $f_1(a) \precneqq f_2(b)$ 'dir [26].

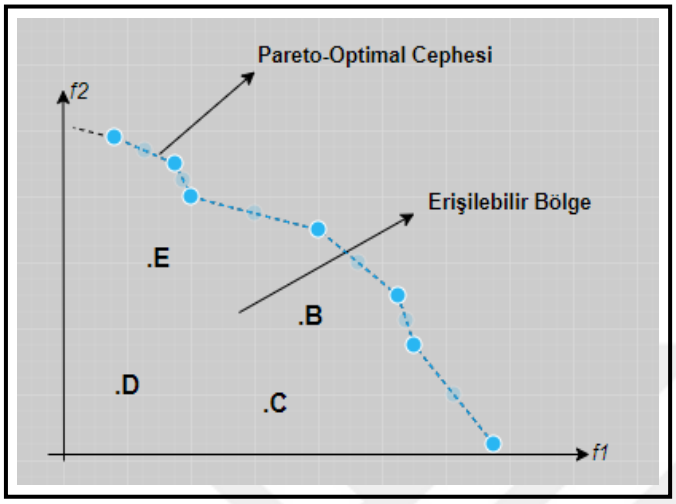
Tanım-3-(Pareto Optimal)

Pareto optimal çözüm, arama uzayında herhangi bir diğer çözüm tarafından domine edilemeyen çözümdür. En iyi çözüm, amaçların herhangi biri içinde en kötü olmayan ve en azından bir amaç fonksiyonu içinde diğerlerinden daha iyi çalışan çözüm demektir [27].

Tanım-4-(Pareto-Optimal Cephe)

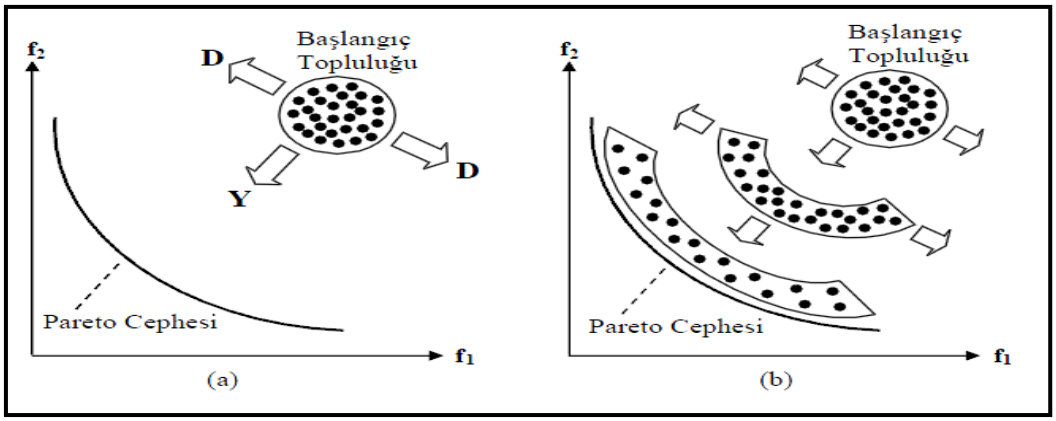
Çok amaçlı optimizasyonda amaç fonksiyonları genellikle istenen bir sonucun farklı bir özelliğini dikkate alır. Çünkü çoğu zaman bu amaçlar, tüm fonksiyonları eş zamanlı olarak

optimize eden tek bir sonucun bulunduğu bir çelişki içerisindedirler. Bundan dolayı da bir optimal çözümler kümesi söz konusu olur. Bu kümeye pareto optimallik kavramından yola çıkılarak pareto optimal cephe denilmektedir [28].



Şekil 3.1. Pareto optimal eğrisi [27]

Çok amaçlı optimizasyon problemlerinde birden fazla pareto optimal çözüm vardır. Bu problemlerde, çözüm kümesinde bir amaca göre iyi olan çözüm diğer amaca göre kötü olabilir. Bundan dolayı, çok amaçlı optimizasyon problemlerinin asıl amacı, pareto cephesini bulmak veya buna yaklaşmak ve bu cephe üzerinde düzgün bir dağılım sağlayarak karar vericiye alternatif karar seçenekleri sunmaktır. Örneğin, Şekil 3.2'de çok amaçlı bir minimizasyon problemi gösterilmektedir. Şekil 3.2(a)'da D düzgün dağılımı, Y yakınsamayı temsil etmektedir ve çözüm kümesinin düzgün dağılımının sağlanabilmesi için başlangıç popülasyonu D yönünde hareket ettirilmelidir. Pareto cephesine yakınsamanın sağlanabilmesi için de başlangıç popülasyonunun Y yönüne hareket ettirilmelidir. Şekil 3.2(b)'de problemde düzgün dağılım ve yakınsama arasındaki denge düzgün olarak ayarlandığında çözüm kümesinin durumunu göstermektedir [29].



Şekil 3.2. (a) Başlangıç popülasyonu (b) Çözüm kümesi [27]

3.1.2. Çok amaçlı evrimsel algoritmalar

Çok amaçlı optimizasyon iki optimizasyon stratejisinin birleşimi üzerine ortaya çıkmıştır. Bu stratejiler şu şekildedir:

1. Elde edilen çözümlerin, gerçek optimal çözümlerden uzaklığını minimize etmek

2. Çözümlerin elde edilen kümesinde, çözümlerin yayılımını maksimize etmek [27].

Evrimsel algoritmaların çok amaçlı optimizasyon problemlerinin çözümünde kullanılması bu stratejilerin içerisindeki koşulları sağlamak üzere pek çok yöntemin bulunmasını sağlamıştır.

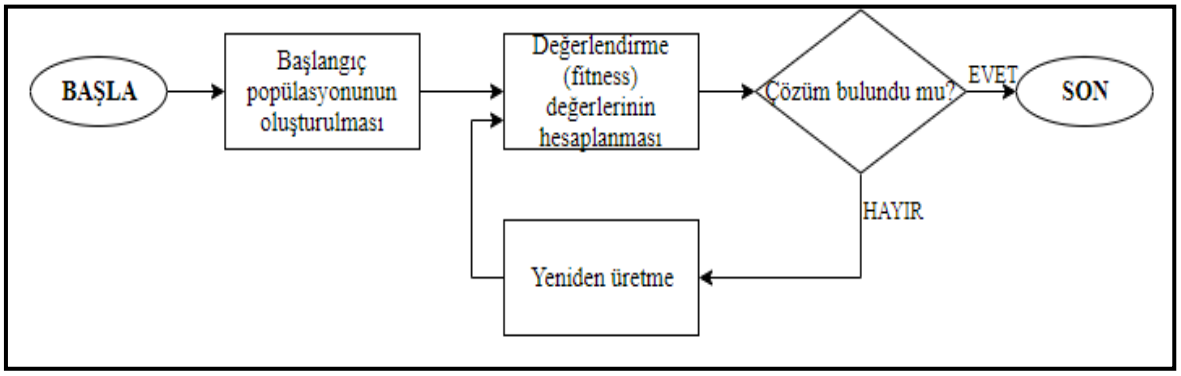
Çok amaçlı evrimsel algoritmalarda, başlangıç popülasyonu genellikle rastgele çözümler üretilerek oluşturulur. Daha sonra başlangıç popülasyonundaki her bir bireyin uygunluk (fitness) değerleri hesaplanır ve seçme operatörü aracılığıyla seçilen çözümler eşleme havuzuna atılır.

Eşleme havuzundaki çözümler çaprazlama ve mutasyon işlemlerine tabi tutulur. Çaprazlama işleminde, öncelikle bireyler belirlenen seçim yöntemi ile seçilir. Ardından rastgele bir çaprazlama noktası seçildikten sonra seçilen bireyler arasında bilgi alışverişi yaparak yeni birey ya da bireyler elde edilir. Çaprazlama işlemi sırasında kullanılan çaprazlama oranı, ebeveyn bireylerin çaprazlama işlemine hangi sıklıkla maruz kalacağını belirlemektedir. Çaprazlama oranının çok yüksek seçilmesi iyi çözümlerin bozulmasına neden olmaktadır. Çaprazlama işlemi sırasında kullanılan çaprazlama dağılım indeksi ise yeni oluşan bireylerin dağılımını kontrol etmek için kullanılır. Bu indeks değeri ne kadar büyük seçilirse ebeveyn bireylere daha benzer bireyler elde edilir.

Mutasyon işleminde ise çaprazlama sonucunda elde edilen birey ya da bireyler rastgele olarak değiştirilir. Mutasyondaki temel amaç, oluşan yeni çözümlerin, önceki çözümlerden farklı olmasını sağlamak ve çözümlerin yerel optimum değere ulaşmasını engellemektir. Mutasyon oranı ise, seçilen bireylerde rastgele seçilen bir gen için değişiklik yapılıp yapılmayacağına karar vermekte kullanılmaktadır. Mutasyon oranı, algoritmanın rastgele bir arama yöntemine dönüşmemesi için genellikle düşük tutulur. Mutasyon işleminde kullanılan dağılım oranı ise mutasyon işlemi sonucunda oluşan yeni bireylerin dağılımını kontrol etmek için kullanılır. Bu değer ne kadar büyük seçilirse ebeveyn bireylere o kadar benzer bireyler elde edilir.

Elitizm işleminde de çaprazlama ve mutasyon işlemleri sonucunda oluşan yeni nesle, bir önceki nesildeki en iyi bireylerin bir kısmı herhangi bir işleme uğramadan olduğu gibi aktarılır. Böylece, popülasyondaki iyi bireyler korunmaktadır. Algoritmanın sonlandırılması için de bir durdurma kriteri belirlenmektedir. Durdurma kriteri sağlanıncaya ya da daha iyi bir sonuç alınamayıncaya kadar algoritmanın çalışması devam eder.

Evrimsel bir algoritmanın işleyişi Şekil 3.3'teki gibidir:



Şekil 3.3. Evrimsel bir algoritmanın işleyişi

Evrimsel çok amaçlı optimizasyon özellikle son yıllarda uygulama ve araştırma alanında popüler ve kullanışlı olmasından dolayı çok sayıda yöntemin ortaya çıkmasını sağlamıştır. Bu yöntemler içinde en bilinen ve en çok kullanılanları; FFGA (Fonseca's and Fleming's multiobjective EA) [30], NPGA (The Niched Pareto Genetic Algorithm) [31], HLGA (Haleja's and Lin's weighted-sum based approach) [32], VEGA (Vector Evaluated Genetic Algorithm) [33], NSGA (The Nondominated Sorting Genetic Algorithm) [34], NSGA II

(Elitist Nondominated Sorting Genetic Algorithm) [35], SPEA (The Strength Pareto Evolutionary Algorithm) [36] olarak sayılabilir [26].

3.1.3. Baskılanamayan çözümler

Çok amaçlı optimizasyonda iki çözüm arasındaki baskınlık ilişkisi aşağıdaki gibi tanımlanmıştır:

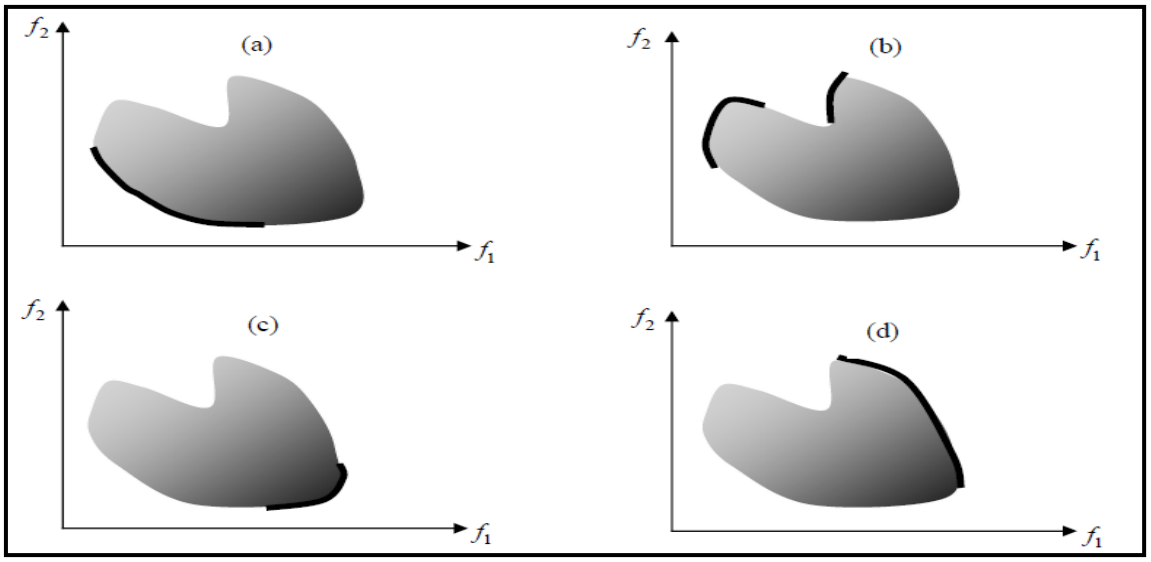
1) x_1 çözümü hiçbir amaç yönünden x_2'den daha kötü değildir. Bu karşılaştırma yapılırken amaç fonksiyonunun değerine veya grafikteki yerine bakılır.

2) x_1 çözümü en az bir amaç bakımından x_2'ye göre daha üstündür.

Yukarıdaki her iki şart da sağlandığı takdirde "x_1 çözümü, x_2 çözümüne baskındır." denilmektedir.

Popülasyon içerisindeki her bir çözüm bu iki kritere bağlı olarak karşılaştırılmaktadır. Herhangi bir x_1 çözümü belirlenen bir amaç fonksiyonu için, x_2 çözümünden kesin olarak daha iyi sonuç verebiliyorsa bunun tersi de doğru olabilir. Ancak herhangi bir x_1 elemanı belirlenen bir amaç fonksiyonu için x_2 çözümünden daha iyi sonuç veremiyorsa bunun tersi her zaman doğru olmayabilir.

Popülasyon içerisindeki çözümler bir grafik üzerinde noktalarla ifade edilmektedir. Popülasyon içerisindeki her bir çözüm baskınlık açısından diğer çözümlerle karşılaştırılır. Bu şekilde baskın olan ve olmayan çözümler belirlenir. Bu karşılaştırmaların sonucunda hiçbir çözümün birbirini baskılayamadığı bir çözüm kümesi oluşmaktadır. Bu çözüm kümesine "baskılanamayan küme" denir. Baskılanamayan küme içerisindeki çözümler ancak bu kümenin dışındaki elemanları baskılayabilmektedir. Baskılanamayan elemanlar için noktalar kullanılarak bir eğri çizildiğinde bu eğriye Non-Dominated Front veya Pareto Optimal Front denilmektedir [27].



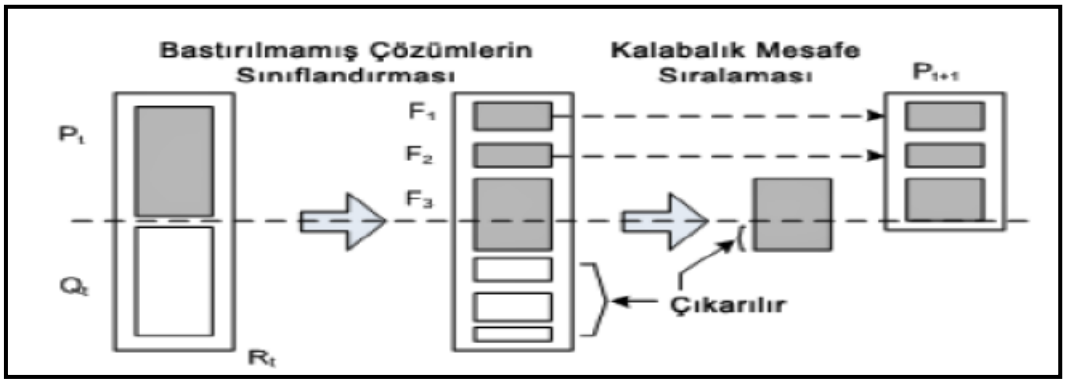
Şekil 3.4. İki amaç fonksiyonu için olabilecek dört farklı durumda elde edilen Pareto eğrileri. a.min f_1 ve min f_2, b.min f_1 ve max f_2, c.max f_1 ve min f_2 d. max f_1 ve max f_2 [37]

3.2.Kullanılan Algoritmalar

Bu bölümde robot tutucuların çok amaçlı optimizasyonu için önerilen hibrit optimizasyon yönteminde kullanılan algoritmalar açıklanmıştır.

3.2.1. Seçkin baskılanamayan sıralamalı genetik algoritma (Elitist non-dominated sorting genetic algorithm-NSGA-II)

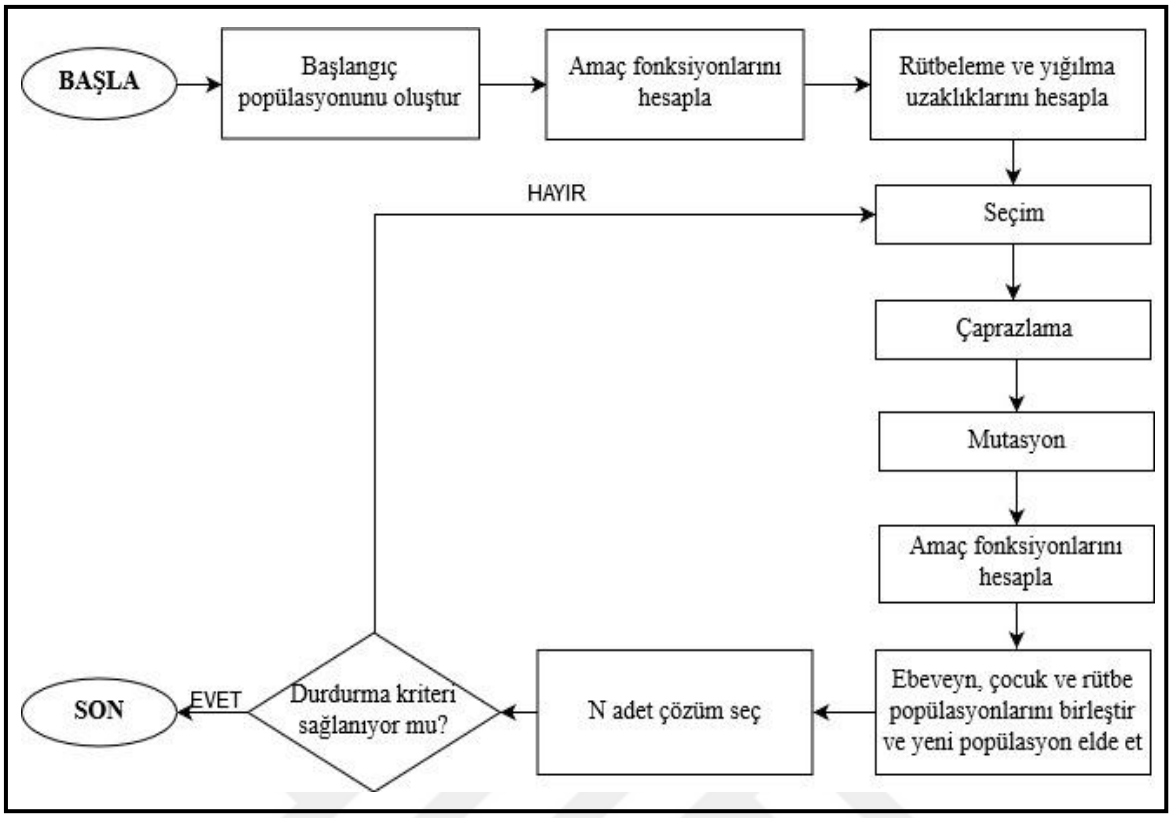
NSGA-II algoritması, Deb ve diğerleri [35] tarafından geliştirilen çok amaçlı bir algoritmadır. Algoritmanın temelleri, daha önce Srinivas ve Deb [38] tarafından geliştirilen NSGA algoritmasının eksik taraflarının giderilmesine dayanır. Strateji bakımından NSGA algoritmasına benzese de algoritmik bakımdan farklı ve yeni yöntemler içermektedir.



Şekil 3.5. NSGA-II prosedürü [35]

Algoritma rastgele olarak belirlenen N tane çözümün bulunduğu bir P_t popülasyonu ile başlamaktadır. Bu popülasyondaki çözümlere çaprazlama ve mutasyon işlemleri uygulanarak N tane çözümlü yeni bir Q_1 popülasyonu oluşturulur. Ardından P_t ve Q_1 popülasyonları birleştirilerek yeni bir R popülasyonu oluşturulur.

R popülasyonu içerisindeki çözümlerin her biri diğerleriyle karşılaştırılarak baskınlık durumlarına göre gruplanır ve F_i kümeleri oluşturulur. Başlangıç popülasyonunun N tane çözümden oluşan P_t olduğu düşünülürse bir sonraki jenerasyondaki P_{t+1} popülasyonunun oluşturulmasında ilk alt küme olan F_1 kümesindeki çözümler kullanılmaktadır. F_1 alt kümesindeki çözümlerin sayısı, başlangıçtaki çözümlerin sayısından (N) az ise F_2 kümesindeki çözümlere geçilir. Bu şekilde N tane çözüm bulununcaya kadar alt kümelerdeki çözümler seçilir ve N tane çözüme ulaşılınca diğer alt kümelerdeki çözümlere geçilmez. Yeni oluşturulan P_{t+1} popülasyonundaki çözümlerin sayısı N olmalıdır, eğer çözümlerin sayısı N'den fazla ise bu durumda yığılma uzaklığı hesaplaması kullanılır. Bu aşamada seçimin yapıldığı kümedeki çözümlerin, ilk olarak birinci amaç fonksiyonunun alt ve üst sınır değerlerine göre yığılma uzaklıkları hesaplanır ve çözümler bu uzaklık değerlerine göre büyükten küçüğe doğru sıralanır. En iyi ve en kötü çözümlerin uzaklığı belirlenir ve daha sonra da aradaki çözümlerin uzaklık değerleri belirlenir. Daha sonra aynı kümedeki çözümlerin ikinci amaç fonksiyonunun alt ve üst sınır değerlerine yığılma uzaklıkları hesaplanır ve çözümler bu uzaklık değerlerine göre sıralanır. Çözümlerin birinci ve ikinci amaç fonksiyonuna göre olan uzaklıkları toplanır ve toplam yığılma uzaklığı hesaplanır. Bu işlem amaç fonksiyonlarının sayısı kadar tekrarlanır. P_{t+1} popülasyonu için gerekli olan N adet çözüm sayısı kadar çözüm, bu uzaklık değerlerine göre belirlenir. Uzaklık değeri en büyük olanlardan başlanmak üzere N adet çözüm bu şekilde seçilir. Elde edilen yeni P_{t+1} popülasyonunda seçimin yapıldığı alt kümedeki çözümlerin yığılma uzaklık değerleri bilinmektedir ancak daha önceden seçilen çözümlerin de uzaklık değerlerinin bilinmesi gerekir. Bunun için kalabalık turnuva seçimi(crowded tournament selection) yöntemi kullanılmaktadır. Bu yönteme göre, rastgele iki adet çözüm seçilir ve bu iki çözüm baskınlık sıra değerlerine göre karşılaştırılır. Baskınlık sıra değeri büyük olan çözüm seçilir ancak iki çözümün baskınlık sıra değeri eşit ise bu durumda yığılma uzaklıkları karşılaştırılır. Uzaklık değeri büyük olan çözüm seçilir ve seçilen bu çözümlerden bir eşleme kümesi oluşturulur. Eşleme kümesindeki bu çözümlere çaprazlama ve mutasyon işlemleri uygulanır ve yeni bir Q_{t+1} popülasyonu elde edilir. Tüm bu işlemler durdurma kriteri sağlanıncaya kadar devam edecektir [35].



Şekil 3.6. NSGA-II algoritmasının akış diyagramı

3.2.2. Ayrıştırmaya dayalı çok amaçlı evrimsel algoritma (Multi-objective evolutionary algorithm based on decomposition-MOEA/D)

İlk olarak Zhang ve Li [39] tarafından önerilen ayrıştırmaya dayanan çok amaçlı evrimsel algoritma (MOEA-D), optimizasyon problemlerinin çözümünde sıklıkla kullanılan çok amaçlı evrimsel algoritmalardan birisidir.

MOEA-D algoritmasında, çok amaçlı bir optimizasyon problemi, ayrıştırma yaklaşımları kullanılarak bir dizi tek optimizasyon alt problemlerine dönüştürülür ve daha sonra bu alt problemleri aynı anda optimize etmek için evrimsel algoritmalar kullanılır [40].

Algoritmanın genel yapısı şu şekilde tanımlanmaktadır [39]: $\lambda^1,.....\lambda^n$ çift yayılım ağırlık vektörü ve $z^*=(z_1^*,.....z_n^*)$ referans noktası olsun. Ele alınan çok amaçlı optimizasyon probleminin alt problemlere ayrıştırılabilmesi için Tchebycheff yaklaşımı kullanılmaktadır. Bu yaklaşımda j.alt probleminin amaç fonksiyonu Eşitlik 3.7 ‘deki gibi tanımlanmaktadır:

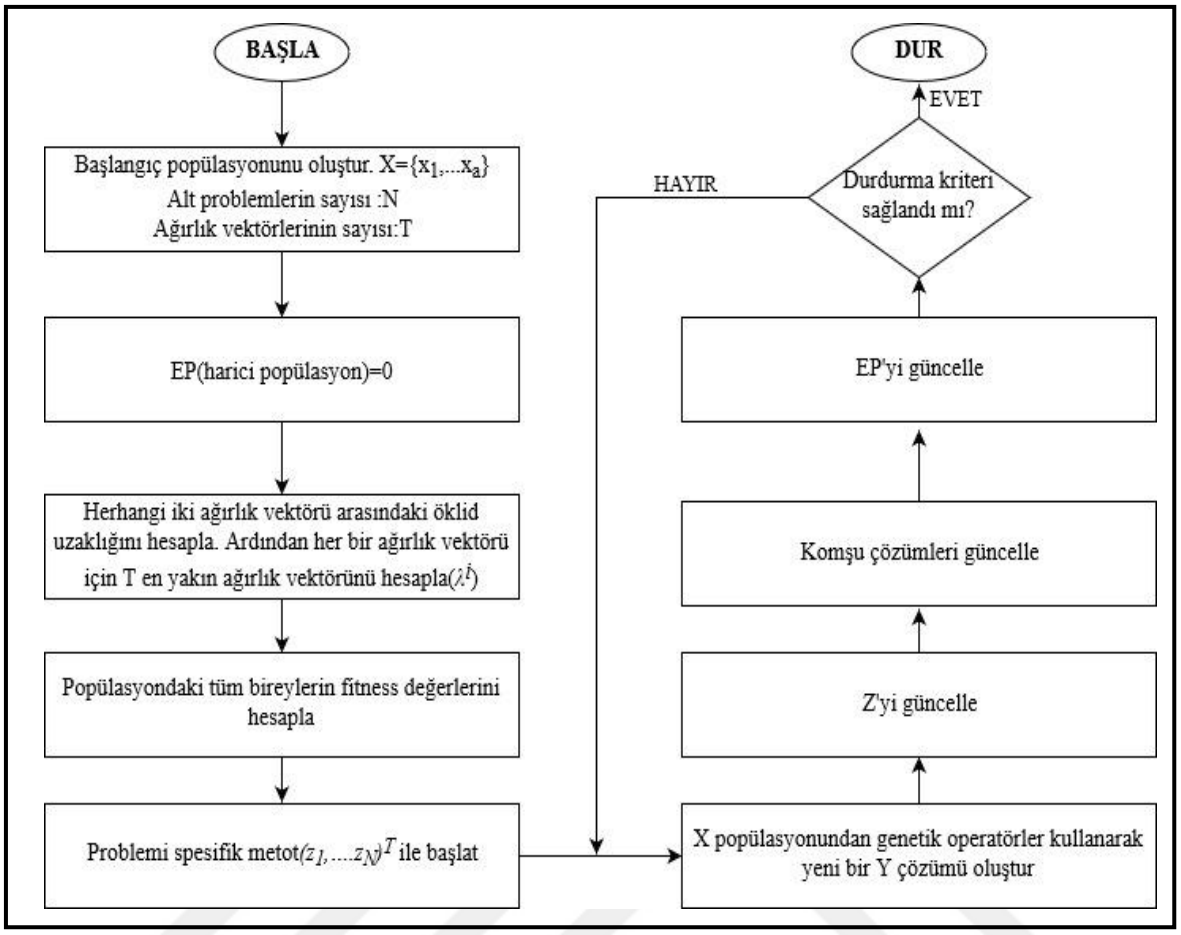
$$g^{te}(x|\ \lambda^j,z^*)=max\{\ \lambda_i^j\ |f_i(x)\ z_i^*\}\quad 1\leq i\leq m \tag{3.7}$$

Eşitlik 3.7'de yer alan $\lambda^j=(\lambda_1^j,\ldots\ldots\lambda_n^j)^T$ ağırlık vektörüne karşılık gelmektedir. MOEA-D algoritması, tüm bu alt amaç fonksiyonlarını tek bir çalıştırmada eş zamanlı olarak optimize etmektedir. Algoritma sırasıyla aşağıdaki adımları izlemektedir:

Girdi: *Başlangıç popülasyonu:$X=\{x_1,\ldots.x_N\}$*
Alt problemlerin sayısı:N
Ağırlık vektörlerinin sayısı:T
Harici popülasyon:EP=0
Spesifik metot($z_1,\ldots.z_N$)

Çıktı: *EP*

for *i=1.....N*
B(i)←ÖklidUzaklığıHesapla
λ^i←EnYakınTağırlıkVektörü
end
P_0←BaşlangıçPopülasyonuOluştur(N)
WHILE*(t<max.jen)*
for *i=1…….N*
k←seçim(B(i))
l←seçim(B(i))
y←YeniÇözümOluştur(x^k,x^l)
for *j=1,.....N*
$f_j(y)$←FitnessDegeriHesapla(z_j)
end
Her j Є B(i)
if($g^{te}(y|\lambda^j,z)\leq g^{te}(x|\lambda^j,z^*)$)
x^j←KomşuÇözümleriGüncelle(y)
FV^j←KomşuÇözümleriGüncelle(y^j)
end
EP←Güncelle(EP)
end
end



Şekil 3.7. MOEA-D algoritmasının akış diyagramı

3.2.3. Güçlü pareto evrimsel algoritma (Strength pareto evolutionary algorithm-SPEA2)

SPEA algoritması, Zitzler, M.Laumanns ve L. Thiele tarafından 1999 yılında geliştirilmiştir. 2001 yılında algoritmadaki bazı eksiklikler giderilerek SPEA2 algoritması sunulmuştur [36].

Algoritmanın işleyişi şu şekildedir: Öncelikle N adet bireyden oluşan P_0 başlangıç popülasyonu oluşturulur. Daha sonra N′ boyutunda P'_0 arşiv popülasyonu üretilir. Başlangıçta P'_0 popülasyonu boştur.

Ardından her bir bireye bir uygunluk (fitness) değeri atanır. Burada uygunluk değeri atanırken hesaplamalara yoğunluk bilgisi de dahil edilmektedir. Başlangıç olarak her bir i bireyine güçlü Eş. 3.8'te gösterildiği gibi bir değer atanır.

$$S(i) = |\{ j \mid j \in Pt + Pt \wedge i \succ j \}| \quad (3.8)$$

Eş. 3.8'deki |.| simgesi eleman sayısını, + simgesi birleştirmeyi, ≻ simgesi ise pareto baskınlık ilişkisini ifade etmektedir. Ham bir fitness değeri, popülasyondaki her bir bireyin güç değerine göre hesaplanmaktadır. Bu değer, hem ana popülasyon hem de arşiv popülasyonunda yer alan çözümlerin güçlü yönleri dikkate alınarak hesaplanmaktadır. Popülasyondaki bireylerin aynı ham fitness değerlerine sahip olmaları durumunda yoğunluk tahmin tekniği kullanılmaktadır. Teknik olarak genellikle k en yakın komşu tekniği kullanılmaktadır. Bu tekniğe göre popülasyon içerisinde yer alan her bir bireyin arşiv popülasyonundaki her bir bireye göre olan mesafesi amaç uzayı içerisinde hesaplanır. Ardından bu mesafe değerleri artan sırayla olacak şekilde bir listede saklanır. *k.* eleman aranan mesafeyi verir ve σ_i^k şeklinde ifade edilir. Burada k, ana popülasyonun büyüklüğü ile arşiv popülasyonunun büyüklüğünün toplamının karekköküne eşit olmaktadır. Mesafe belirlendikten sonra Eş. 3.9'daki denklem kullanılarak yoğunluk hesaplaması yapılır.

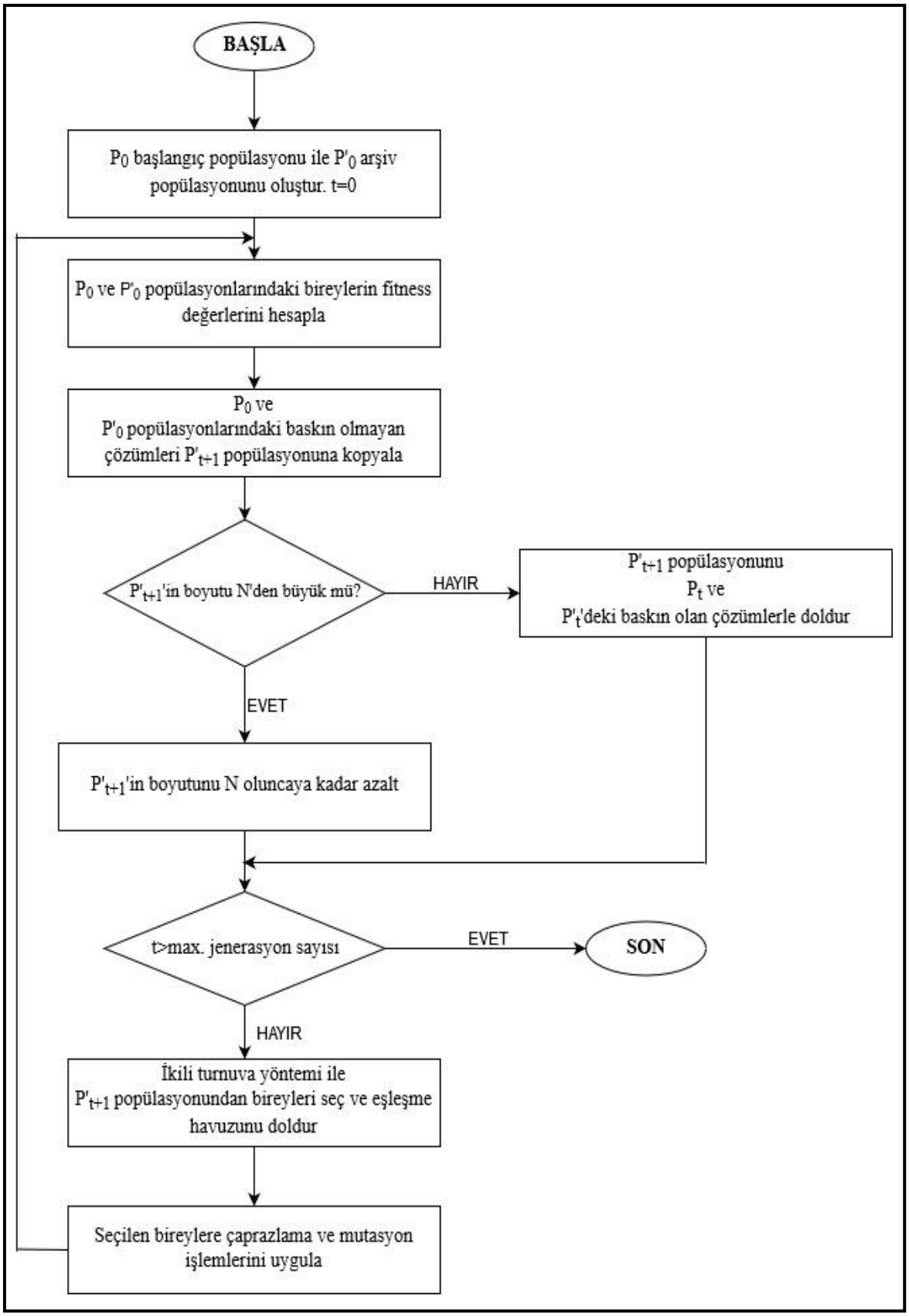
$$D(i) = 1 / \sigma_i^k + 2 \tag{3.9}$$

Yoğunluk ölçüsü *D(i)*, ham fitness değeri olan *R(i)*'ye eklenerek bir bireyin uygunluk değeri Eş. 3.10'da gösterilen denklem kullanılarak hesaplanır.

$$F(i) = R(i) + D(i) \tag{3.10}$$

Uygunluk değerlerine göre arşiv popülasyonu yeniden oluşturulur. P_t popülasyonundaki baskın olmayan tüm çözümler (t, jenerasyon sayısı) P'_{t+1} popülasyonuna kopyalanır. Eğer P'_{t+1} popülasyonunun boyutu maksimum popülasyonu boyutu N'i aşarsa bu durumda P'_{t+1}'in boyutu azaltılır. P'_{t+1}'in boyutunun N'den az olması durumunda ise P'_{t+1} popülasyonu P_t popülasyonu ve arşiv kümesindeki baskın olan çözümlerle doldurulur.

Daha sonra eşleme havuzunu doldurmak üzere P'_{t+1} popülasyonundan ikili turnuva seçimi yöntemi ile bireyler seçilir. Seçilen bireylere çaprazlama ve mutasyon işlemleri uygulanır ve P'_{t+1} popülasyonu güncellenir. Durdurma kriteri sağlanıncaya kadar yapılan işlemler tekrarlanır [41].



Şekil 3.8. SPEA2 algoritmasının akış diyagramı

3.2.4. Pareto zarflama temelli seçim algoritması (Pareto envelope-based selection algorithm-PESA2)

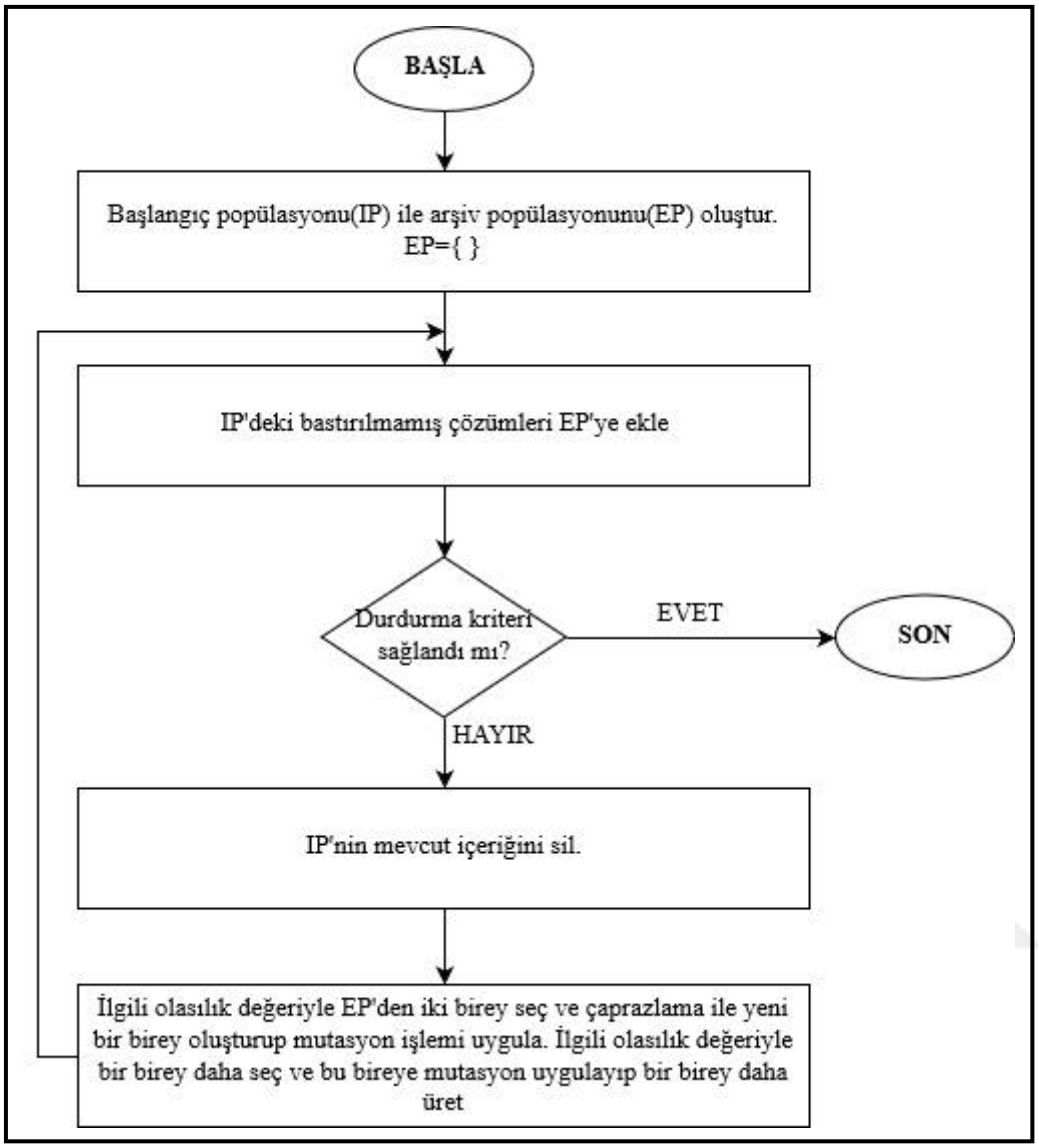
Pareto Zarflama Temelli Seçim Algoritması (PESA2), Corne ve diğerleri [42] tarafından önerilen bir genetik algoritma türüdür. Çok amaçlı optimizasyon problemlerinin çözümünde sıklıkla kullanılan PESA, seçim ve çeşitliliğin korunması amacıyla hiper grid şeması kullanmaktadır. Seçim işlemi için, popülasyon içerisindeki bastırılmış çözümlerin derecelendirmelerine bağlı bir katsayı kullanılır. Çeşitliliğin sağlanması için de diğer genetik algoritmalardaki gibi çaprazlama ve mutasyon işlemleri uygulanmaktadır.

PESA2'de çaprazlama ve mutasyon oranları gibi standart parametreler kullanılır. Bunlara ek olarak popülasyon boyutuyla ilgili iki ve hiper grid kalabalık stratejisi ile ilgili olarak bir parametre daha kullanılmaktadır.

Algoritmanın işleyişi şu şekildedir:

IP dahili (başlangıç) popülasyonunun boyutu P_I olmak üzere tüm kromozomları üretilir. *EP* harici popülasyonunun boyutu P_E olarak tanımlanır, ancak başlangıçta *EP* boştur. Daha sonra yeni aday çözümlerin o anki popülasyonu olan *IP* tek tek arşivde birleştirilir. Eğer *IP* içerisindeki bir aday çözüm bastırılmamışsa ve arşivdeki herhangi bir çözüm tarafından da bastırılamıyorsa arşive girebilmektedir. Bir aday çözüm arşive girdikten sonra, bu aday tarafından arşivde bastırılan çözümler arşiv içerisinden çıkarılmaktadır. Arşive bireylerin eklenmesi sırasında arşivin başlangıçta belirlenen boyutu P_E aşılırsa, arşivdeki çözümlerden biri ya da birkaçı çıkarılır.

Durdurma kriteri sağlanmamışsa *IP*'nin içeriği silinir ve P_i yeni aday çözümleri oluşturuluncaya kadar, p_c olasılığı ile arşiv popülasyonu *EP*'den iki birey seçilir ve bu bireylerden çaprazlamayla yeni bir çözüm üretilerek üzerine mutasyon işlemi uygulanır. Ardından *pc* olasılığı ile *EP*'den bir birey daha seçilir ve yeni bir birey oluşturmak için seçilen bu bireye mutasyon işlemi uygulanır. Bu işlemin ardından durdurma kriteri sağlanıncaya kadar arşiv birleştirme işlemine geri dönülür ve yapılan işlemler tekrarlanır. Durdurma kriteri sağlandığında algoritma sonlandırılır ve *EP*'deki çözümler sonuç olarak verilir [43].



Şekil 3.9. PESA algoritmasının akış diyagramı

3.3. MOEA Framework

MOEA Framework, çok amaçlı evrimsel algoritmalar ve diğer genel amaçlı optimizasyon algoritmalarını geliştirmek ve deneyimlemek amacıyla kullanılan ücretsiz ve açık kaynak kodlu bir Java kütüphanesidir. JMetal ve PISA kütüphanelerini de destekleyen MOEA Framework, 30 adet çok amaçlı optimizasyon algoritmasına sahiptir [44].

MOEA, temel bir algoritma seti, test problemleri ve arama operatörleri içermektedir. Ancak ilave bileşenleri içerecek şekilde kolayca genişletilebilmektedir. Servis Sağlayıcı Arayüzü (Service Provider Interface-SPI) sayesinde yeni algoritmalar ve problemler framework içerisine sorunsuz bir şekilde entegre edilebilmektedir. MOEA Framework

kütüphanesinin iyi yapılandırılmış nesne yönelimli tasarımı, mevcut bileşenleri birleştirerek yeni optimizasyon algoritmalar oluşturmaya da olanak sağlamaktadır [44].

MOEA Framework üzerinde 3 temel sınıf bulunmaktadır: Executor, Instrumenter, Analyzer. Executor sınıfı, algoritmaları istenilen koşullara göre ayarlayıp çalıştırmak için kullanılmaktadır. Instrumenter sınıfı, Executor sınıfı ile birlikte çalışmaktadır ve Executor sınıfı ile elde edilen verilerin toplanmasını sağlamaktadır. Analyzer sınıfı ise algoritmaların çalışmasının sonucunda elde edilen sonuçları veya bir algoritmanın farklı parametrelerle buldukları sonuçları karşılaştırmada kullanılmaktadır [45].

MOEA Framework üzerinde yeni bir problem tanımlanırken "Problem" arayüzü kullanılmaktadır. Ancak yeni bir problem tanımlamak için bu arayüzü kullanmak şart değildir, bunun yerine "AbstractProblem" sınıfı genişletilerek de yeni bir problem tanımlanabilmektedir. "Problem" sınıfı kendi içerisinde "Solution" sınıfını kullanmaktadır. Bu sınıf aracılığıyla problemlerdeki karar değişkenleri ve değer aralıkları, amaç fonksiyonları ve kısıtlar tanımlanabilmektedir. "Problem" sınıfı içinde kullanılan "evaluate" metodu aracılığıyla da probleme ait olan karar değişkenleri bir diziye atanır ve bu değişkenlerin değerlerine göre de amaç fonksiyonlarının değerleri hesaplanmaktadır. Ardından da amaç fonksiyonların değerleri "Solution" sınıfına amaç olarak atanmaktadır. Bu aşamalardan sonra problemin tanımlanması bitirilir ve problem MOEA Framework tarafından yürütülmek üzere kullanılır [45].

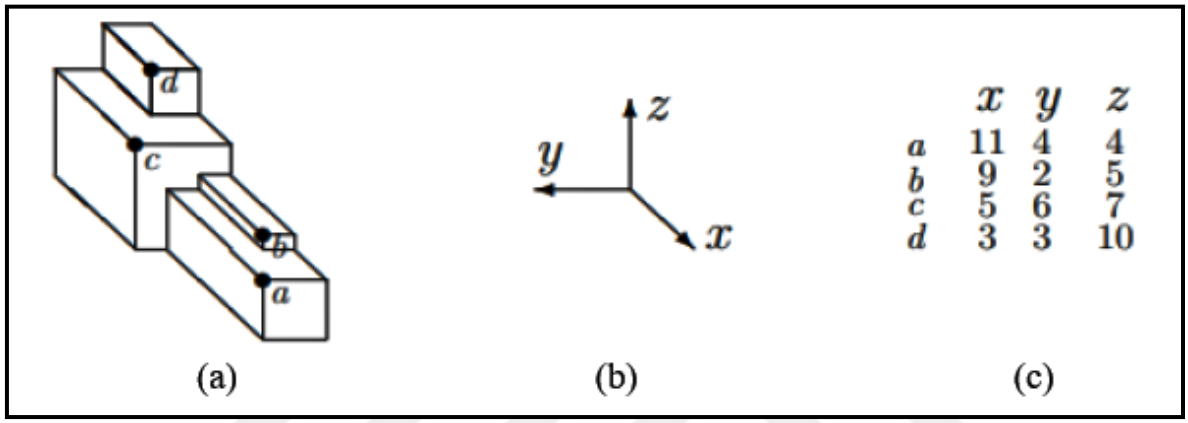
3.4. Performans Metrikleri

Çok amaçlı optimizasyon algoritmalarının performanslarının değerlendirilmesinde literatürde kullanılan pek çok metrik çeşidi vardır. Bu tez çalışması kapsamında optimizasyon algoritmalarının performanslarının değerlendirilmesinde hypervolume ve inverted generational metrikleri kullanılmıştır.

3.4.1. Hypervolume

Çok amaçlı optimizasyon algoritmalarının performans değerlendirilmesinde kullanılan en popüler metriklerden biri hypervolume metriğidir. Aynı zamanda S-metrik veya Lebesgue ölçüsü olarak da bilinmektedir [46].

Çözümler amaç uzayındaki noktalar olarak kabul edilirse hypervolume, bir çözüm kümesinin içerdiği boyutsal alandır, yani bazı referans noktalarına göre kümenin n boyutlu hacmidir. Çok amaçlı optimizasyona uyarlandığında, çözümler n boyutlu amaç fonksiyonlarının değerleri nokta olarak değerlendirilebilir. Yani bir kümenin hypervolume değeri, çözümlerin baskın olduğu alanların toplam boyutudur. Şekil 3.10'da üç boyutta bir hypervolume gösterilmektedir. Burada, noktalar daireler ile işaretlenir ve harflerle etiketlenir. Setin hypervolume değeri, a-d noktalarının kapsadığı alanın hacmidir [47].



	x	y	z
a	11	4	4
b	9	2	5
c	5	6	7
d	3	3	10

Şekil 3.10. Üç boyutta örnek bir hypervolume [48]

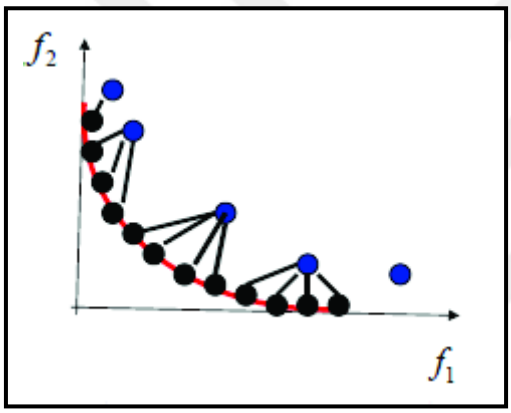
Hypervolume, çözümlerin Pareto optimal cephesinin yayılmasının bir ölçüsünü ve aynı zamanda Pareto-optimal cephesindeki setin mesafesini ölçerek bunları tek bir değerde saklamaktadır. Bir kümenin hypervolume değeri, bir referans noktasına, genellikle alan içinde en uygun olmayan noktaya ya da "mümkün olan en kötü nokta" ya göre ölçülmektedir. Referans noktası seçimi, hypervolume değerlerinin karşılaştırmasından kaynaklı sonuçları etkileyebileceğinden çok önemlidir. Hala tartışılan bir sorun olsa da önerilen, her bir amaç için bilinen en kötü değeri almak ya da uygun bir miktara kaydırmaktır[49].

3.4.2. Ters kuşak mesafesi (Inverted generational distance-IGD)

Çok amaçlı optimizasyon algoritmalarının performans değerlendirilmesinde kullanılan popüler metriklerden biri de ters kuşak mesafesi metriğidir. Ters kuşak mesafesi metriği, iki önemli avantaja sahiptir. Bunlardan birincisi, hesaplama verimliliğidir. Birçok karmaşık problemde dahi kolayca hesaplanabilmektedir. Bir diğer avantajı ise, elde edilen çözümlerin kalitesini göstermesidir, yani elde edilen çözümlerin Pareto-optimal cephesine

yakınlaşması ve bu cephedeki çeşitliliğini yansıtmasıdır. Bu özellikleri sayesinde son zamanlarda ters kuşak mesafesi, evrimsel çok amaçlı optimizasyon algoritmalarının performansının değerlendirilmesinde sıklıkla kullanılmaya başlanmıştır [50].

Ters kuşak mesafesi, hem yakınsama hem de çeşitliliği aynı anda mevcut referans setine bağlı olarak gösteren bir metriktir. Şekil 3.11'de ters kuşak mesafesinin örnek bir gösterimi bulunmaktadır. Burada, tüm siyah noktalar Pareto optimal cephesi boyunca eşit olarak dağıtılan Pareto çözümleridir. Mavi noktalar ise herhangi bir algoritma tarafından üretilen çözümlerdir. Daha küçük bir ters kuşak mesafesi değeri, Pareto optimal cepheye doğru daha iyi bir yakınsama anlamına gelmektedir [51].



Şekil 3.11. Ters kuşak mesafesi [51]

4. DENEYSEL ÇALIŞMALAR

Bu çalışmada, robot tutucuların tasarım optimizasyonu probleminin çözümü çok amaçlı hibrit optimizasyon yöntemi ile çözülmüştür. Robot tutucuların tasarımına dair ikinci bölümde açıklanan ilk üç konfigürasyon ele alınmış ve her bir konfigürasyon için çok amaçlı hibrit optimizasyon yöntemi uygulanmıştır. Hibrit çok amaçlı optimizasyon yöntemi olarak üç farklı hibrit yaklaşımı kullanılmıştır. Bunlar, NSGAII-SPEA2, NSGAII-PESA2 ve NSGAII-MOEAD algoritmalarıdır. Belirlenen bu hibrit algoritmalar, üç farklı konfigürasyon üzerinde uygulanmış ve elde edilen sonuçlar literatürdeki benzer çalışmaların sonuçları ile karşılaştırılmıştır.

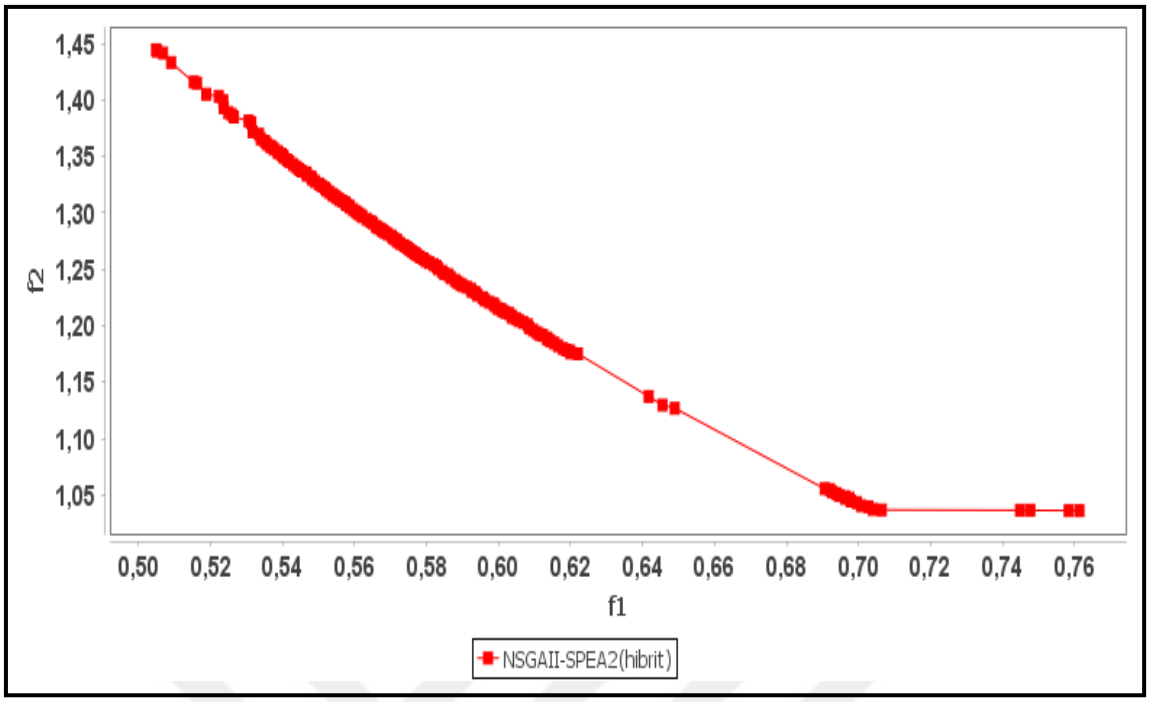
4.1. Konfigürasyonlar İçin Yapılan Optimizasyon Çalışmaları

Bu bölümde ikinci bölümde açıklaması yapılan robot tutucu konfigürasyonlarından ilk üç konfigürasyon üzerinde yapılan optimizasyon çalışmaları ve sonuçları yer almaktadır.

4.1.1. Birinci konfigürasyon için yapılan deneysel çalışmalar

Bu konfigürasyon için belirlenen iki amaç fonksiyonunun optimizasyonu için ilk olarak NSGAII-SPEA2 hibrit algoritması uygulanmıştır. Hibrit algoritmanın uygulanmasında popülasyon büyüklüğü 200 ve maksimum jenerasyon sayısı 30000 olarak belirlenmiştir. Çaprazlama yöntemi olarak tek noktalı çaprazlama yöntemi tercih edilmiş ve çaprazlama oranı 1,0, dağılım indeksi 15,0 olarak belirlenmiştir. Mutasyon yöntemi olarak ise polinom mutasyon yöntemi uygulanmıştır ve mutasyon oranı 1/7, dağılım oranı ise 20,0 olarak alınmıştır.

NSGAII-SPEA2 hibrit algoritması ile yapılan optimizasyon çalışmasında toplam 150 adet pareto-optimal çözüm elde edilmiştir. Bu çözümlere dair Pareto-optimal eğrisi Şekil 4.1'de gösterilmektedir.



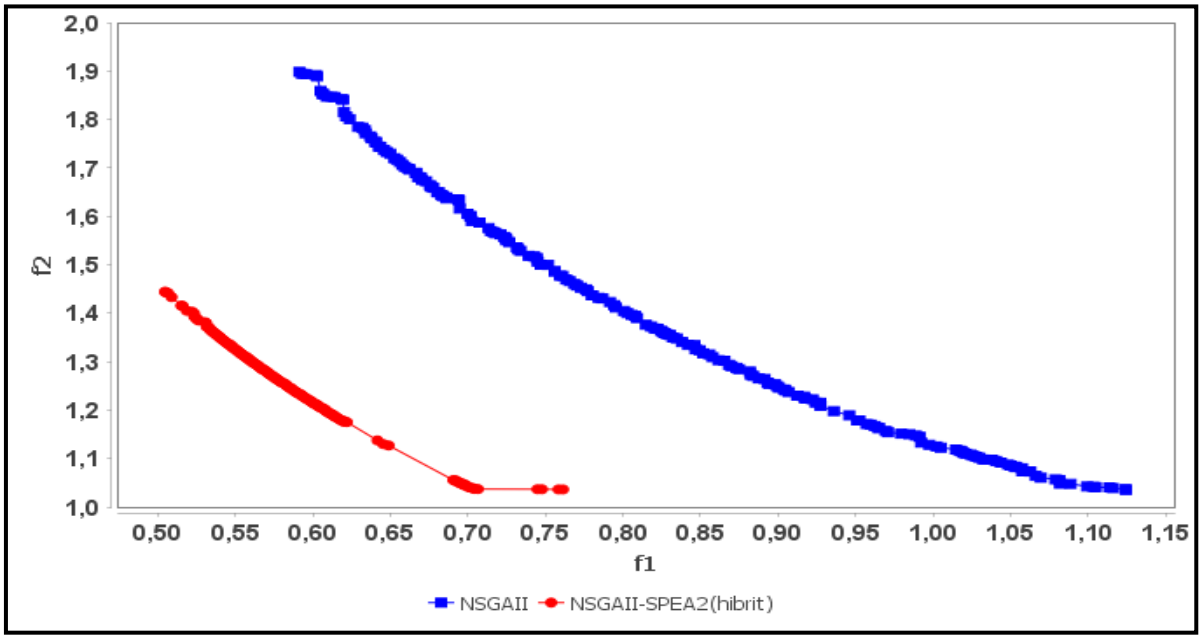
Şekil 4.1. Birinci konfigürasyon için hibrit NSGAII-SPEA2 algoritması ile elde edilen optimizasyon sonuçları

Çizelge 4.1'de ise bu çalışma ve daha önce yapılan diğer üç çalışmanın en iyi çözümüne ait tasarım değişkenlerinin değerleri ile optimizasyon çalışmalarında belirlenen kriterler karşılaştırılmıştır.

Çizelge 4.1. Literatürdeki çalışmaların sonuçları ile hibrit NSGAII-SPEA2 ile elde edilen sonuçların karşılaştırılması (birinci konfigürasyon)

Tasarım Değişkenleri	***GA[3]***	***NSGAII[4]***	***NSGAII[5] (large comp.)***	***NSGAII[5] (small comp.)***	***NSGAII-SPEA2***
a	135,0	88,1	250,0	250,0	237,46
b	90,53	52,98	250,0	233,0	217,58
c	102,2	100,0	243,6	210,3	199,56
e	0,0	29,95	0,0	14,0	15,53
f	1,28	69,89	37,0	15,0	67,26
l	170,57	126,86	100,0	180,0	187,82
δ	1,8	3,14	1,72	1,85	2,14
Popülasyon Boyutu	400	100	200	200	200
Çaprazlama Oranı	0,6	0,9	0,9	0,9	1,0
Mutasyon Oranı	0,08	1,0	0,1	0,1	1/7
Jenerasyon Sayısı	400	150	N/A	1000	30000
$f_1(x)$	3,05	3,7168	1,55	0,734	0,704
$f_2(x)$	2,0	1,5767	0,994	0,998	1,037

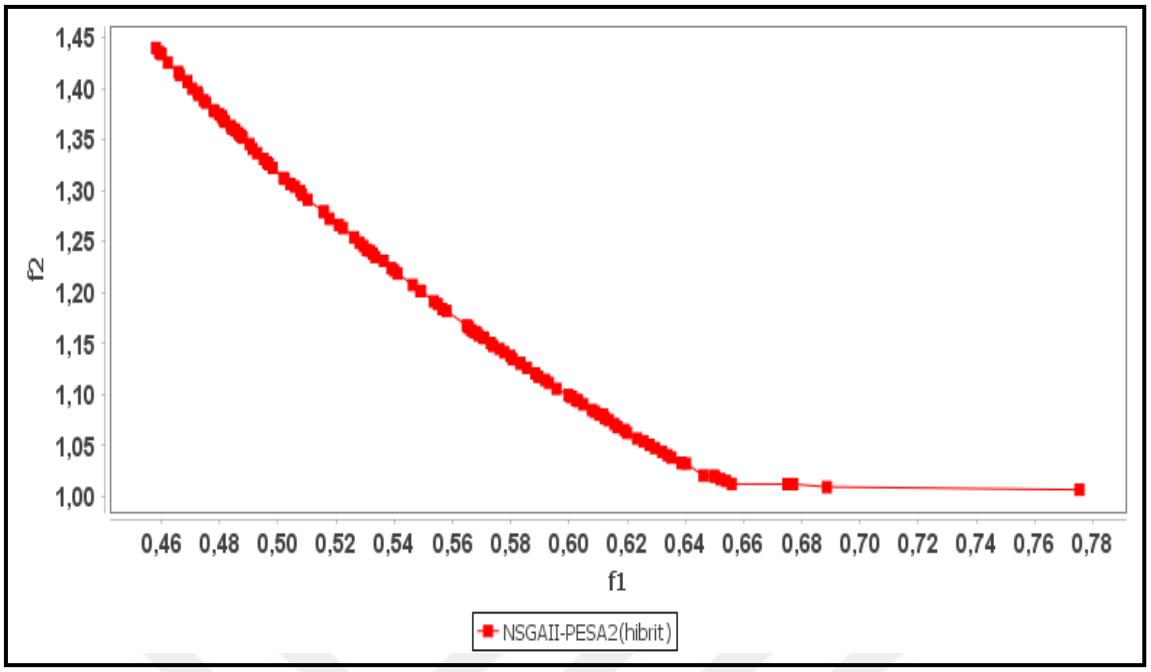
Datta ve Deb'in çalışmasında [5] iki farklı çözüm kümesi kullanılmıştır(small-large computations). Ancak Datta ve Deb'in [5] çalışmasındaki sonuçlar net olmadığından NSGA-II algoritması aynı parametrelerle birinci konfigürasyon üzerinde uygulanmıştır. 30000 iterasyon için hibrit NSGAII-SPEA2 ile NSGA-II algoritmasına ait Pareto optimal eğrileri Şekil 4.2'de gösterilmektedir. İki algoritmaya ait Pareto optimal eğrileri karşılaştırıldığında hibrit NSGAII-SPEA2 algoritması ile daha iyi sonuçların elde edildiği görülmektedir.



Şekil 4.2. Birinci konfigürasyon için hibrit NSGAII-SPEA2 ve NSGA-II algoritması ile elde edilen sonuçlar

Bu konfigürasyon için ikinci algoritma olarak hibrit NSGAII-PESA2 algoritması uygulanmıştır. Hibrit algoritmanın uygulanmasında popülasyon büyüklüğü 200 ve maksimum jenerasyon sayısı 30000 olarak belirlenmiştir. Ayrıca hibrit algoritmanın PESA2 algoritması kısmında kullanılan arşiv büyüklüğü ise 100 olarak belirlenmiştir. Çaprazlama yöntemi olarak tek noktalı çaprazlama yöntemi tercih edilmiş ve çaprazlama oranı 1,0, dağılım indeksi 15,0 olarak belirlenmiştir. Mutasyon yöntemi olarak ise polinom mutasyon yöntemi uygulanmıştır ve mutasyon oranı 1/7, dağılım oranı ise 20,0 olarak alınmıştır.

NSGAII-PESA2 hibrit algoritması ile yapılan optimizasyon çalışmasında toplam 128 adet pareto-optimal çözüm elde edilmiştir. Bu çözümlere dair Pareto-optimal eğrisi Şekil 4.3'te gösterilmektedir.



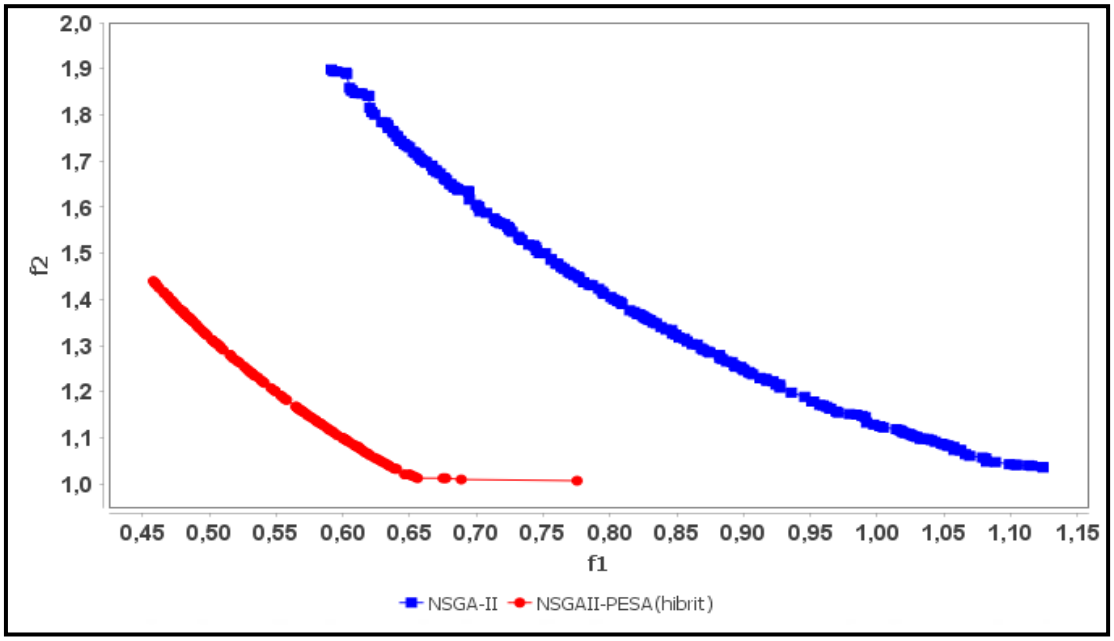
Şekil 4.3. Birinci konfigürasyon için hibrit NSGAII-PESA2 algoritması ile elde edilen optimizasyon sonuçları

Çizelge 4.2'de literatürdeki benzer çalışmaların sonuçları ile NSGAII-PESA2 algoritmasıyla elde edilen sonuçlar karşılaştırılmıştır.

Çizelge 4.2. Literatürdeki çalışmaların sonuçları ile hibrit NSGAII-PESA2 ile elde edilen sonuçların karşılaştırılması (birinci konfigürasyon)

Tasarım Değişkenleri	***GA[3]***	***NSGAII[4]***	***NSGAII[5] (large comp.)***	***NSGAII[5] (small comp.)***	***NSGAII-PESA2***
a	135,0	88,1	250,0	250,0	234,35
b	90,53	52,98	250,0	233,0	228,92
c	102,2	100,0	243,6	210,3	204,22
e	0,0	29,95	0,0	14,0	1,85
f	1,28	69,89	37,0	15,0	40,46
l	170,57	126,86	100,0	180,0	182,50
δ	1,8	3,14	1,72	1,85	1,92
Popülasyon Boyutu	400	100	200	200	200
Çaprazlama Oranı	0,6	0,9	0,9	0,9	1,0
Mutasyon Oranı	0,08	1,0	0,1	0,1	1/7
Jenerasyon Sayısı	400	150	N/A	1000	30000
$f_1(x)$	3,05	3,7168	1,55	0,734	0,688
$f_2(x)$	2,0	1,5767	0,994	0,998	1,009

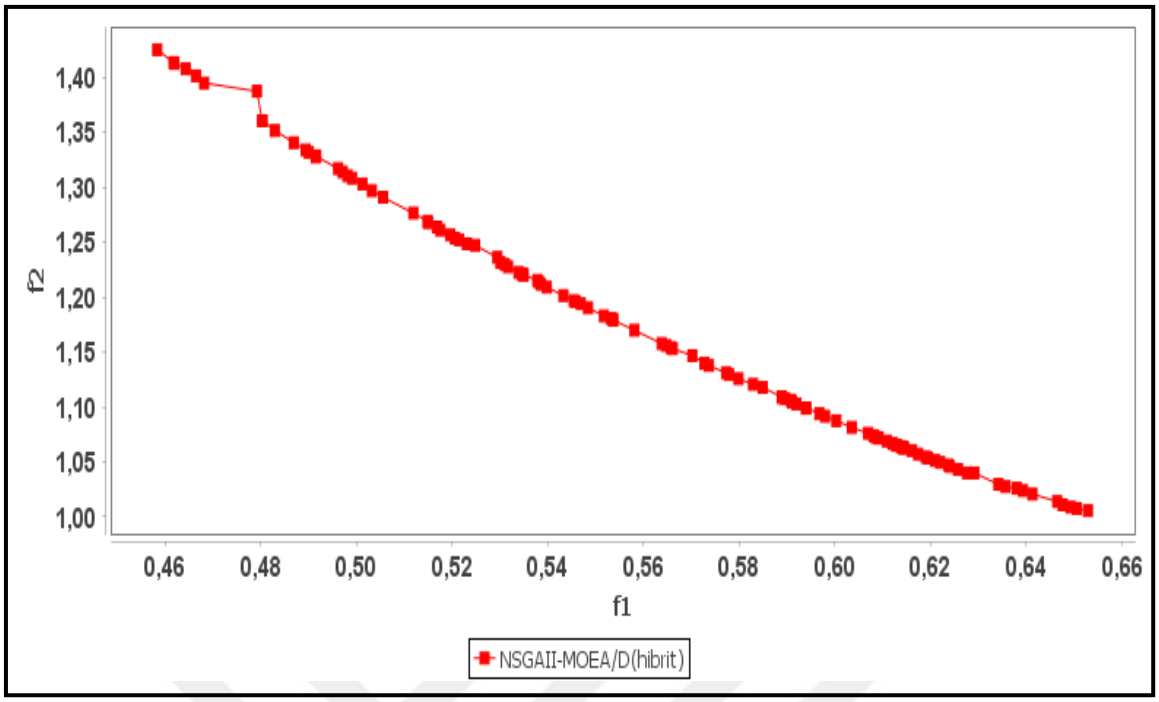
Datta ve Deb'in [5] çalışmasındaki sonuçlar net olmadığı için NSGA-II algoritması aynı parametrelerle birinci konfigürasyon üzerinde uygulanmıştır. 30000 iterasyon için hibrit NSGAII-PESA2 ile NSGA-II algoritmasına ait Pareto optimal eğrileri Şekil 4.4 'te gösterilmektedir. İki algoritmaya ait Pareto optimal eğrileri karşılaştırıldığında hibrit NSGAII-PESA2 algoritması ile daha iyi sonuçlar elde edilmiştir.



Şekil 4.4. Birinci konfigürasyon için hibrit NSGAII-PESA2 ve NSGA-II algoritması ile elde edilen sonuçlar

Bu konfigürasyon için son algoritma olarak da hibrit NSGAII-MOEA/D algoritması uygulanmıştır. Hibrit algoritmanın uygulanmasında popülasyon büyüklüğü 200 ve maksimum jenerasyon sayısı 30000 olarak belirlenmiştir. Çaprazlama yöntemi olarak tek noktalı çaprazlama yöntemi tercih edilmiş ve çaprazlama oranı 1,0, dağılım indeksi 15,0 olarak belirlenmiştir. Mutasyon yöntemi olarak ise polinom mutasyon yöntemi uygulanmıştır ve mutasyon oranı 1/7, dağılım oranı ise 20,0 olarak alınmıştır.

NSGAII-MOEA/D hibrit algoritması ile yapılan optimizasyon çalışmasında toplam 113 adet pareto-optimal çözüm elde edilmiştir. Bu çözümlere dair Pareto-optimal eğrisi Şekil 4.5'te gösterilmektedir.



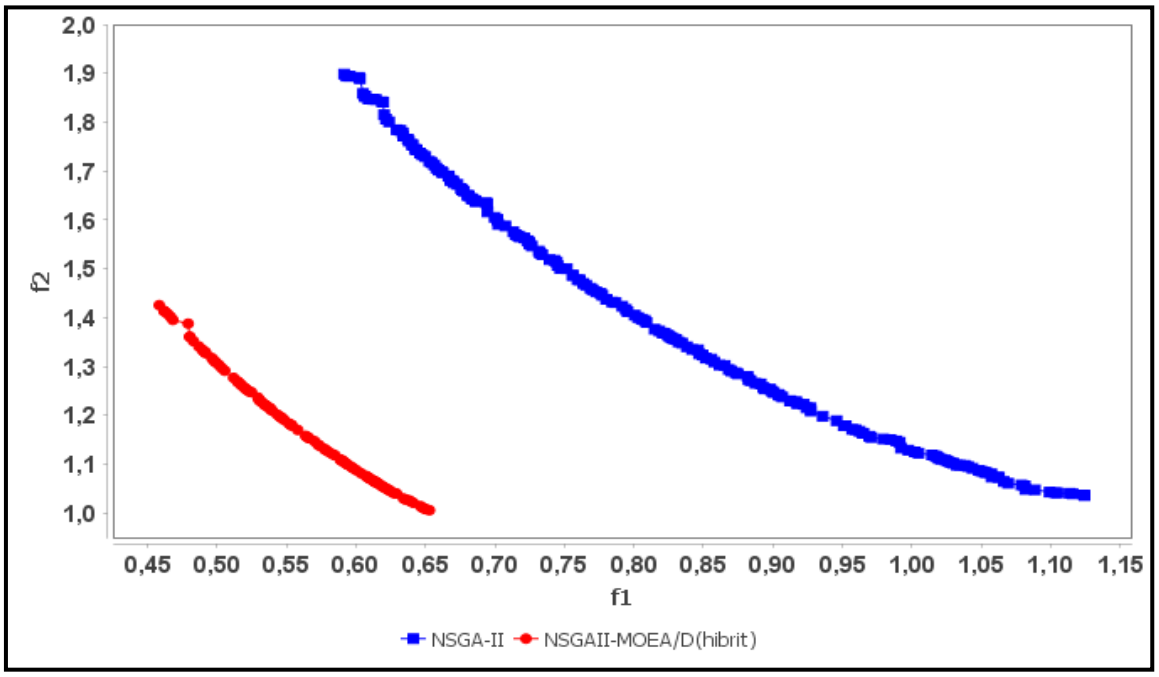
Şekil 4.5. Birinci konfigürasyon için hibrit NSGAII-MOEA/D algoritması ile elde edilen optimizasyon sonuçları

Çizelge 4.3'te literatürdeki benzer çalışmaların sonuçları ile NSGAII-MOEA/D algoritmasıyla elde edilen sonuçlar karşılaştırılmıştır.

Çizelge 4.3. Literatürdeki çalışmaların sonuçları ile hibrit NSGAII-MOEA/D ile elde edilen sonuçların karşılaştırılması (birinci konfigürasyon)

Tasarım Değişkenleri	***GA[3]***	***NSGAII[4]***	***NSGAII[5] (large comp.)***	***NSGAII[5] (small comp.)***	***NSGAII-MOEA/D***
a	135,0	88,1	250,0	250,0	237,75
b	90,53	52,98	250,0	233,0	221,04
c	102,2	100,0	243,6	210,3	210,78
e	0,0	29,95	0,0	14,0	15,74
f	1,28	69,89	37,0	15,0	17,92
l	170,57	126,86	100,0	180,0	141,61
δ	1,8	3,14	1,72	1,85	1,8
Popülasyon Boyutu	400	100	200	200	200
Çaprazlama Oranı	0,6	0,9	0,9	0,9	1,0
Mutasyon Oranı	0,08	1,0	0,1	0,1	1/7
Jenerasyon Sayısı	400	150	N/A	1000	30000
$f_1(x)$	3,05	3,7168	1,55	0,734	0,652
$f_2(x)$	2,0	1,5767	0,994	0,998	1,005

Datta ve Deb'in [5] çalışmasındaki sonuçlar net olmadığından dolayı önceki iki hibrit algoritmada olduğu gibi hibrit NSGAII-MOEA/D algoritması ile NSGA-II algoritması aynı parametrelerle birinci konfigürasyon üzerinde uygulanmıştır. 30000 iterasyon için hibrit NSGAII-MOEA/D ile NSGA-II algoritmasına ait Pareto optimal eğrileri Şekil 4.6'da gösterilmektedir. İki algoritmaya ait Pareto optimal eğrileri karşılaştırıldığında hibrit NSGAII- MOEA/D algoritması ile daha iyi sonuçlar elde edildiği görülmektedir.

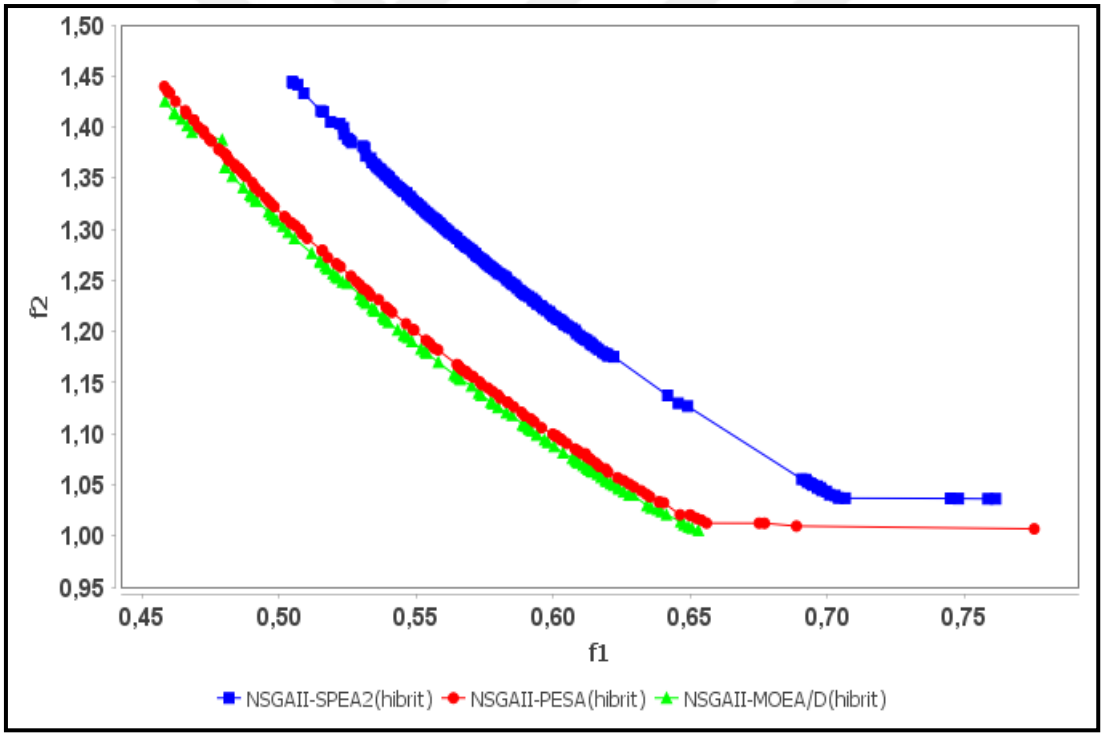


Şekil 4.6. Birinci konfigürasyon için hibrit NSGAII-MOEA/D ve NSGA-II algoritması ile elde edilen sonuçlar

Birinci konfigürasyon için üç hibrit algoritmanın sonuçlarının karşılaştırması Çizelge 4.4'te verilmiştir. Şekil 4.7'de ise üç hibrit algoritmanın optimizasyon sonuçlarına dair pareto-optimal eğrileri gösterilmektedir. Pareto optimal eğrileri üzerinden bir değerlendirme yapıldığında hibrit NSGA-II-MOEA/D algoritmasıyla daha iyi sonuçlar elde edildiği görülmüştür.

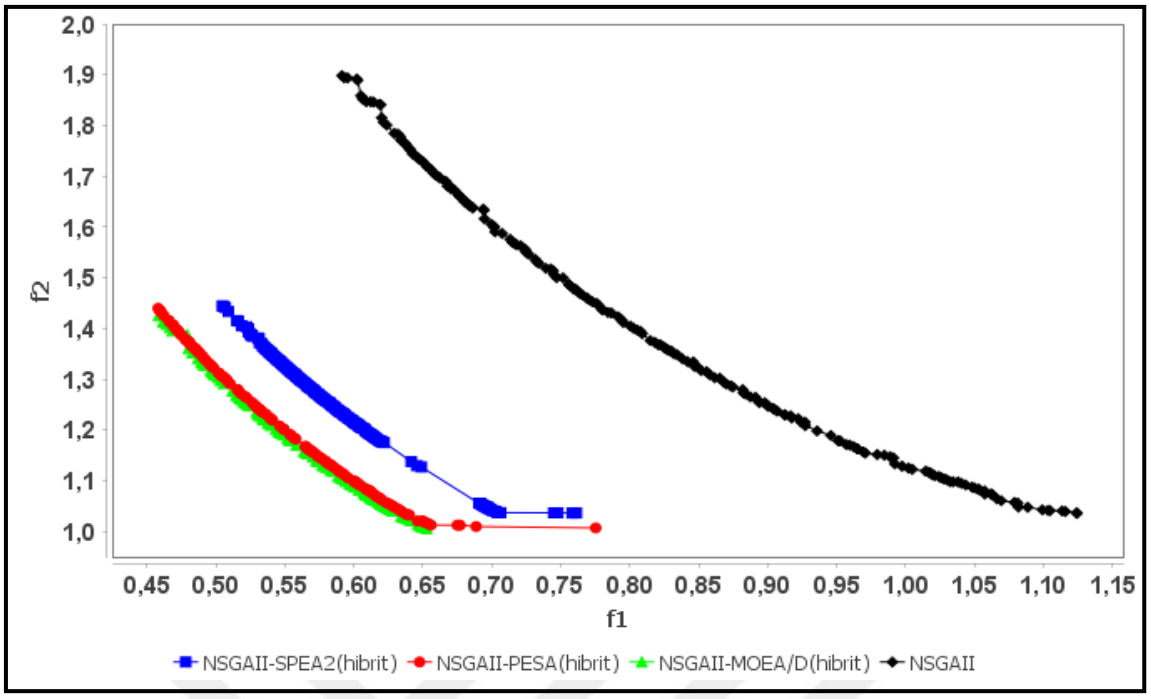
Çizelge 4.4. Birinci konfigürasyon için hibrit algoritma sonuçlarının karşılaştırılması

Tasarım Değişkenleri	*NSGAII-SPEA2*	*NSGAII-PESA2*	*NSGAII-MOEA/D*
a	237,46	234,35	237,75
b	217,58	228,92	221,04
c	199,56	204,22	210,78
e	15,53	1,85	15,74
f	67,26	40,46	17,92
l	187,82	182,50	141,61
δ	2,14	1,92	1,8
Popülasyon Boyutu	200	200	200
Çaprazlama Oranı	1,0	1,0	1,0
Mutasyon Oranı	1/7	1/7	1/7
Jenerasyon Sayısı	30000	30000	30000
$f_1(x)$	0,704	0,688	0,652
$f_2(x)$	1,037	1,009	1,005



Şekil 4.7. Birinci konfigürasyon için tüm hibrit algoritmaların Pareto-optimal eğrileri

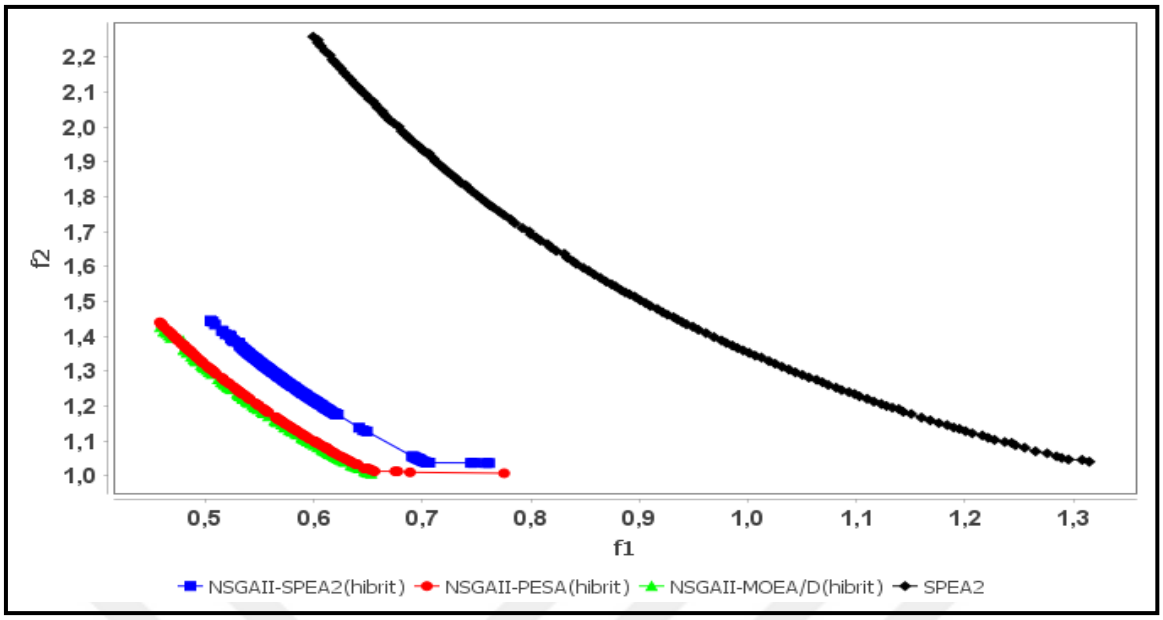
Şekil 4.8'de 30000 iterasyon için tüm hibrit algoritmalar ile NSGA-II algoritmasına ait Pareto optimal eğrileri gösterilmektedir. Buna göre pareto optimal eğrileri incelendiğinde her üç hibrit algoritma ile elde edilen sonuçların NSGA-II algoritması ile elde edilen sonuçlara göre daha iyi olduğu gözlemlenmiştir.



Şekil 4.8. Birinci konfigürasyon için tüm hibrit algoritmalar ile NSGA-II algoritmasıyla elde edilen sonuçlar

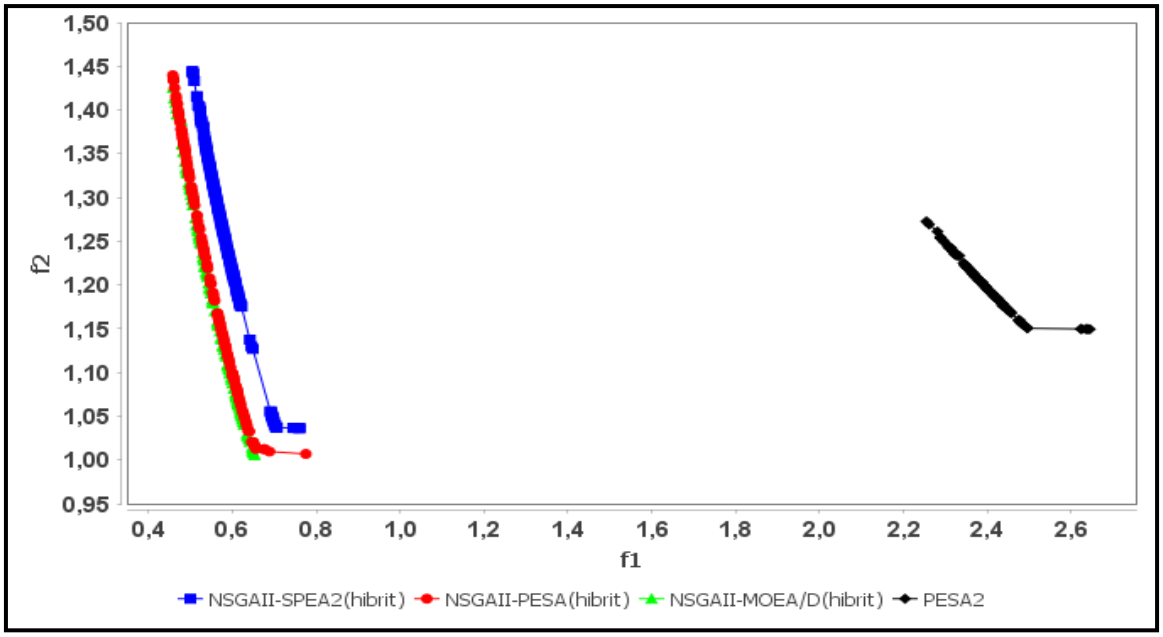
Önerilen hibrit yöntemlerin verimliliğini gösterebilmek amacıyla NSGA-II algoritmasına ek olarak SPEA2, PESA2 ve MOEA/D algoritmaları da eşit koşullarda birinci konfigürasyon üzerinde uygulanmıştır. Bu algoritmalarda çaprazlama işlemi için çaprazlama oranı 1,0, dağılım indeksi ise 15,0 kullanılmıştır. Mutasyon işlemi için ise mutasyon oranı 1/7, dağılım oranı ise 20,0 parametreleri kullanılmıştır. Söz konusu algoritmalar bu parametre değerleriyle hibrit algoritmalarda olduğu gibi 30000 jenerasyon ile birinci konfigürasyon üzerinde uygulanmıştır.

İlk olarak 30000 iterasyon için tüm hibrit algoritmalar ile SPEA2 algoritmasına ait Pareto-optimal eğrileri Şekil 4.9'da gösterilmektedir. SPEA2 algoritması ile 150 adet pareto-optimal çözüm elde edilmiştir. Pareto-optimal eğrileri incelendiğinde her bir hibrit algoritmanın sonuçlarının SPEA2 algoritmasından elde edilen sonuçlara göre daha üstün olduğu görülmektedir.



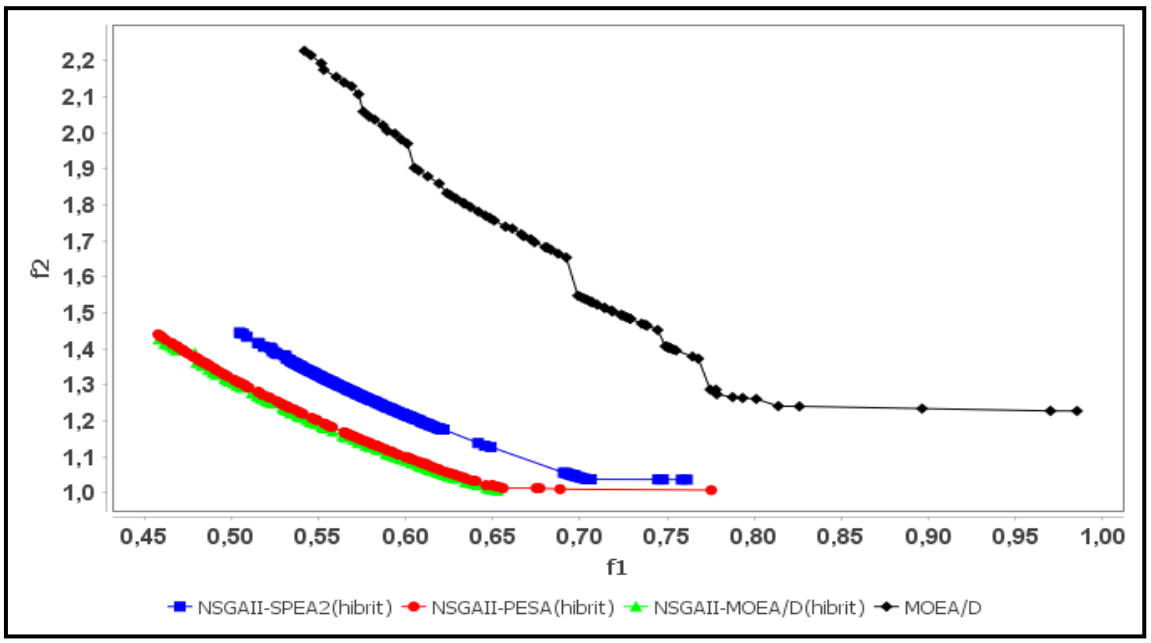
Şekil 4.9. Birinci konfigürasyon için tüm hibrit algoritmalar ile SPEA2 algoritmasıyla elde edilen sonuçlar

Şekil 4.10'da 30000 iterasyon için tüm hibrit algoritmalar ile PESA2 algoritmasına ait Pareto-optimal eğrileri görülmektedir. PESA2 algoritması ile 118 adet pareto-optimal çözüm elde edilmiştir. Pareto-optimal eğrileri incelendiğinde hibrit algoritma ile elde edilen sonuçların PESA2 algoritmasından daha iyi olduğu gözlemlenmiştir.



Şekil 4.10. Birinci konfigürasyon için tüm hibrit algoritmalar ile PESA2 algoritmasıyla elde edilen sonuçlar

Son olarak 30000 jenerasyon için tüm hibrit algoritmalar ile MOEA/D algoritması ile elde edilen sonuçlar karşılaştırılmıştır. MOEA/D algoritması ile 83 adet pareto-optimal çözüm elde edilmiştir. Şekil 4.11'de bu karşılaştırmaya dair Pareto-optimal eğrileri görülmektedir. Buna göre, her bir hibrit algoritma ile elde edilen sonuçların MOEA/D algoritması ile elde edilen sonuçlara bariz bir üstünlük kurduğu görülmektedir.

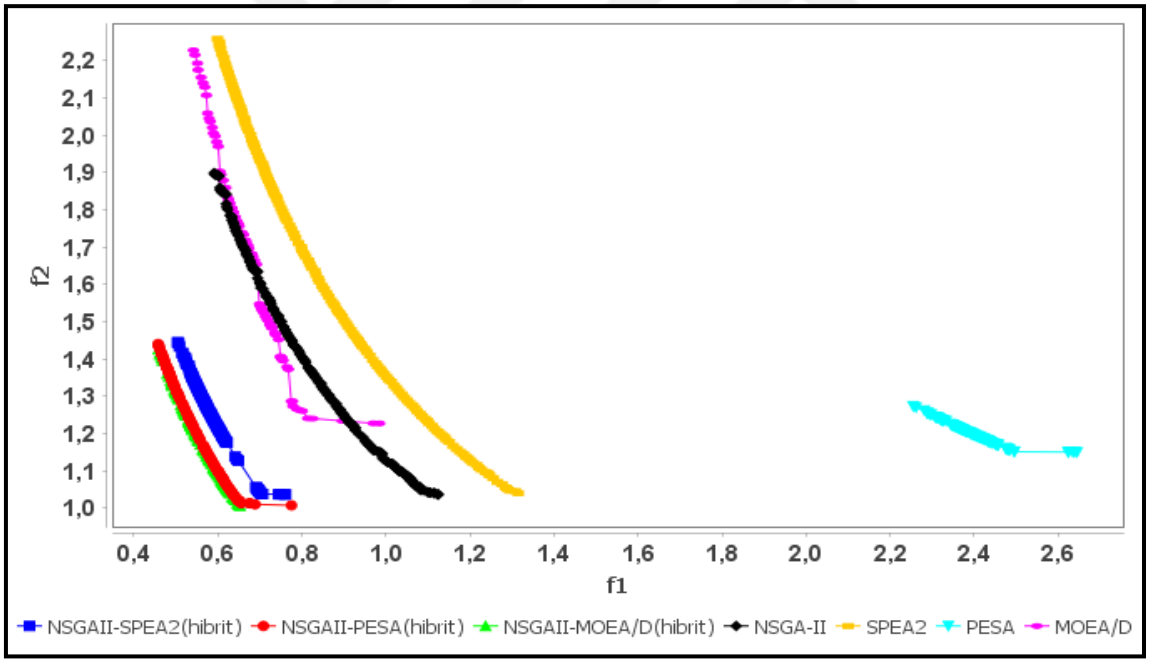


Şekil 4.11. Birinci konfigürasyon için tüm hibrit algoritmalar ile MOEA/D algoritmasıyla elde edilen sonuçlar

Çizelge 4.5'te tüm hibrit algoritmalar ile NSGA-II, SPEA2, PESA2 ve MOEA/D algoritması ile elde edilen bazı optimal çözümler gösterilmektedir. Şekil 4.12'de ise birinci konfigürasyon için yukarıda ayrı ayrı değerlendirilmesi yapılan ve aynı parametrelerle çalıştırılan hibrit ve hibrit olmayan tüm algoritmalara dair Pareto-optimal eğrileri tek bir grafik üzerinde gösterilmektedir. Bu Pareto-optimal eğrileri incelendiğinde önerilen hibrit algoritmaların hepsinin hibrit olmayan algoritmalardan daha iyi sonuç verdiği görülmektedir. Hibrit algoritmalar kendi aralarında değerlendirildiğinde de hibrit NSGAII-MOEA/D algoritması ile daha iyi sonuçlar elde edildiği görülmüştür.

Çizelge 4.5. Birinci konfigürasyon için hibrit algoritmalar ile diğer algoritmalar ile edilen bazı optimal çözümler

Tasarım Değişkenleri	*NSGAII-SPEA2*	*NSGAII-PESA2*	*NSGAII-MOEA/D*	*NSGA-II*	*SPEA2*	*PESA2*	*MOEA/D*
a	237,4	234,3	237,7	226,1	192,6	149,9	225,96
b	217,5	228,9	221,0	180,1	170,8	114,7	186,86
c	199,5	204,2	210,7	224,7	250,4	110,4	220,07
e	15,53	1,85	15,74	30,46	8,11	21,94	22,32
f	67,26	40,46	17,92	53,96	22,55	60,96	122,81
l	187,8	182,5	141,6	221,5	199,3	161,1	226,56
δ	2,14	1,92	1,8	2,22	1,96	2,46	2,53
Popülasyon B.	200	200	200	200	200	200	200
Çaprazlama Or.	1,0	1,0	1,0	1,0	1,0	1,0	1,0
Mutasyon Or.	1/7	1/7	1/7	1/7	1/7	1/7	1/7
Jenerasyon Say.	30000	30000	30000	30000	30000	30000	30000
$f_1(x)$	0,704	0,688	0,652	0,714	0,718	2,261	0,718
$f_2(x)$	1,037	1,009	1,005	1,569	1,885	1,269	1,503



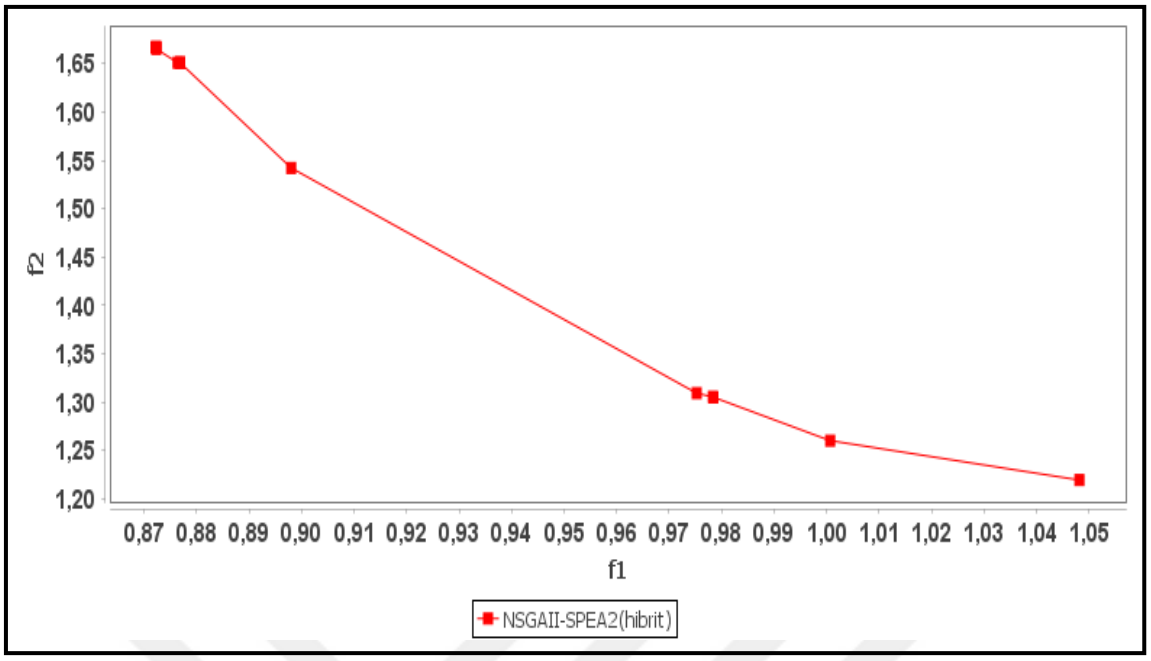
Şekil 4.12. Birinci konfigürasyon için tüm hibrit algoritmalar ile hibrit olmayan algoritmalara dair Pareto-optimal eğrileri

4.1.2. İkinci konfigürasyon için yapılan deneysel çalışmalar

Bu konfigürasyon için yapılan deneysel çalışmaların sonuçları, birinci konfigürasyonda olduğu gibi literatürdeki benzer çalışmaların sonuçları ile karşılaştırılmıştır. Ancak bu konfigürasyon sonuçlarında, birinci konfigürasyon sonuçlarından farklı olarak karşılaştırılan çalışmalar arasında Datta ve Deb'in [5] çalışması yer almamaktadır. Bunun sebebi, Datta ve Deb'in [5] çalışmasında bu konfigürasyonun kullanılmamasıdır.

Bu konfigürasyon için tanımlanan iki amaç fonksiyonunun optimizasyonu için öncelikle hibrit NSGAII-SPEA2 algoritması uygulanmıştır. Hibrit algoritmanın ilk kısmını oluşturan NSGAII algoritması için popülasyon büyüklüğü 200 ve maksimum jenerasyon sayısı 30000 olarak belirlenmiştir. Çaprazlama yöntemi olarak tek noktalı çaprazlama yöntemi kullanılmıştır ve çaprazlama oranı 0,9, dağılım indeksi ise 20,0 olarak alınmıştır. Mutasyon yöntemi olarak ise polinom mutasyon yöntemi uygulanmıştır ve mutasyon oranı 0.1, dağılım oranı ise 100,0 olarak belirlenmiştir. Hibrit algoritmanın ikinci kısmını oluşturan SPEA2 algoritması için ise çaprazlama yöntemi olarak yine tek noktalı çaprazlama yöntemi tercih edilmiştir ve çaprazlama oranı 0,9, dağılım indeksi ise 20,0 olarak belirlenmiştir. Mutasyon yöntemi olarak da polinom mutasyon yöntemi kullanılmıştır ve mutasyon oranı 0,1, dağılım oranı ise 100,0 olarak belirlenmiştir.

NSGAII-SPEA2 hibrit algoritması ile yapılan optimizasyon çalışmasında toplam 15 adet pareto-optimal çözüm elde edilmiştir. Bu çözümlere dair Pareto-optimal eğrisi Şekil 4.13'te gösterilmektedir.



Şekil 4.13. İkinci konfigürasyon için hibrit NSGAII-SPEA2 algoritması ile elde edilen optimizasyon sonuçları

Çizelge 4.6'da literatürdeki benzer çalışmaların sonuçları ile NSGAII-SPEA2 algoritmasıyla elde edilen sonuçlar karşılaştırılmıştır. Elde edilen sonuçlara göre amaç fonksiyonlarının aldığı değerlere bakıldığında hibrit NSGAII-SPEA2 algoritmasıyla elde edilen $f_1(x)$ ve $f_2(x)$ değerlerinin, referans olarak alınan çalışma olan Osyzcka ve Krenich'in [3] $f_1(x)$ ve $f_2(x)$ değerlerinden daha iyi olduğu görülmektedir.

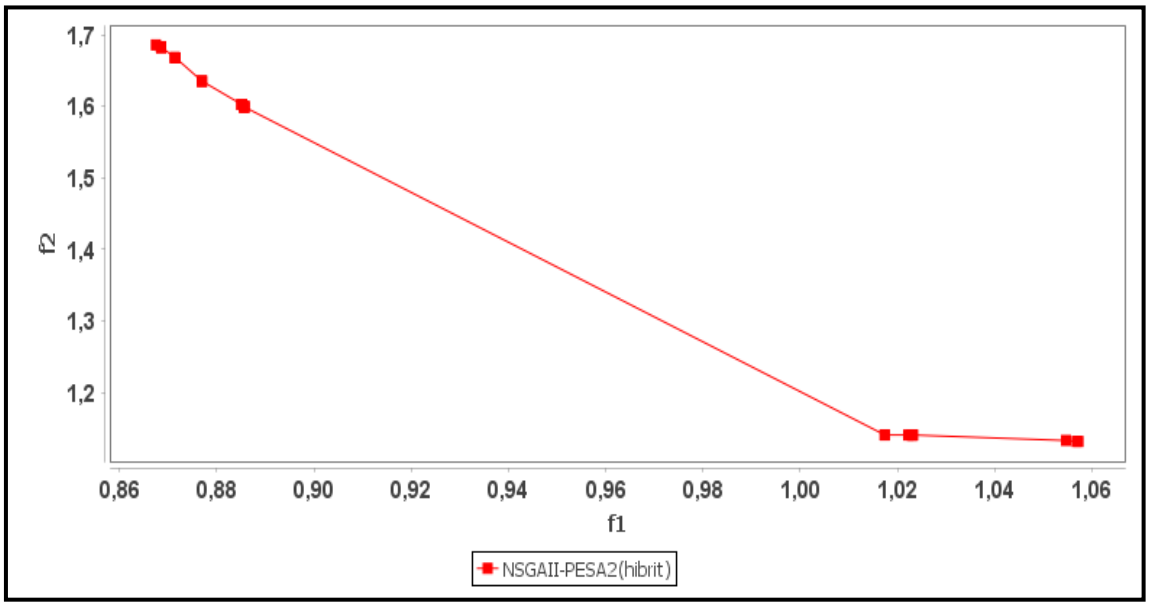
Çizelge 4.6. Literatürdeki çalışmaların sonuçları ile hibrit NSGAII-SPEA2 ile elde edilen sonuçların karşılaştırılması (ikinci konfigürasyon)

Tasarım Değişkenleri	***GA[3]***	***NSGAII[4]***	***NSGAII-SPEA2***
a	135,0	88,1	143,05
b	90,53	52,98	87,86
c	102,2	100,0	100,03
e	0,0	29,95	49,99
f	1,28	69,89	43,93
l	170,57	126,86	126,33
δ	1,8	3,14	2,63
Popülasyon Boyutu	400	100	200
Çaprazlama Oranı	0,6	0,9	0,9
Mutasyon Oranı	0,08	1,0	0,1
Jenerasyon Sayısı	400	150	30000
$f_1(x)$	1,67	0,04	0,978
$f_2(x)$	2,0	1,5767	1,305

Çizelge 4.6'da bir diğer çalışma olan Saravanan ve diğerlerinin [4] sonuçlarına bakıldığında ise hibrit algoritma ile edilen $f_2(x)$ değerinin, Saravanan ve diğerlerinin [4] çalışmasındaki $f_2(x)$ değerinden daha iyi olduğu, ancak $f_1(x)$ değerinin ise daha kötü olduğu görülmektedir. Bunun sebebi, Saravanan ve diğerlerinin [4] çalışmasında kullanılan farklı konfigürasyonlardır.

Bu konfigürasyon için ikinci algoritma olarak ise hibrit NSGAII-PESA2 algoritması uygulanmıştır. Hibrit algoritmanın ilk kısmını oluşturan NSGAII algoritması için popülasyon büyüklüğü 200 ve maksimum jenerasyon sayısı 30000 olarak belirlenmiştir. Çaprazlama yöntemi olarak tek noktalı çaprazlama yöntemi kullanılmıştır ve çaprazlama oranı 0,9, dağılım indeksi ise 20,0 olarak alınmıştır. Mutasyon yöntemi olarak ise polinom mutasyon yöntemi uygulanmıştır ve mutasyon oranı 0,1, dağılım oranı ise 100,0 olarak belirlenmiştir. Hibrit algoritmanın ikinci kısmını oluşturan PESA2 algoritması için ise çaprazlama yöntemi olarak tek noktalı çaprazlama yöntemi seçilip çaprazlama oranı 0,9, dağılım indeksi ise 20,0 olarak alınmıştır. Mutasyon yöntemi olarak da polinom mutasyon yöntemi kullanılmıştır ve mutasyon oranı 0,1, dağılım oranı ise 100,0 olarak belirlenmiştir.

NSGAII-PESA2 hibrit algoritması ile yapılan optimizasyon çalışmasında toplam 37 adet pareto-optimal çözüm elde edilmiştir. Bu çözümlere dair Pareto-optimal eğrisi Şekil 4.14'te gösterilmektedir.



Şekil 4.14. İkinci konfigürasyon için hibrit NSGAII-PESA2 algoritması ile elde edilen optimizasyon sonuçları

Çizelge 4.7'de literatürdeki benzer çalışmaların sonuçları ile NSGAII-PESA2 algoritmasıyla elde edilen sonuçlar karşılaştırılmıştır. Elde edilen sonuçlarda amaç fonksiyonlarının aldığı değerlere bakıldığında hibrit NSGAII-PESA2 algoritmasıyla elde edilen f_1(x) ve f_2(x) değerlerinin, referans olarak alınan çalışma olan Osyzcka ve Krenich'in [3] f_1(x) ve f_2(x) değerlerinden bariz bir şekilde daha iyi olduğu görülmektedir. Saravanan ve diğerlerinin çalışması ile kıyaslandığında ise f_1(x) değerinin, Saravanan ve diğerlerinin [4] değerlerinden daha düşük, f_2(x)'in ise daha iyi bir değer aldığı görülmektedir.

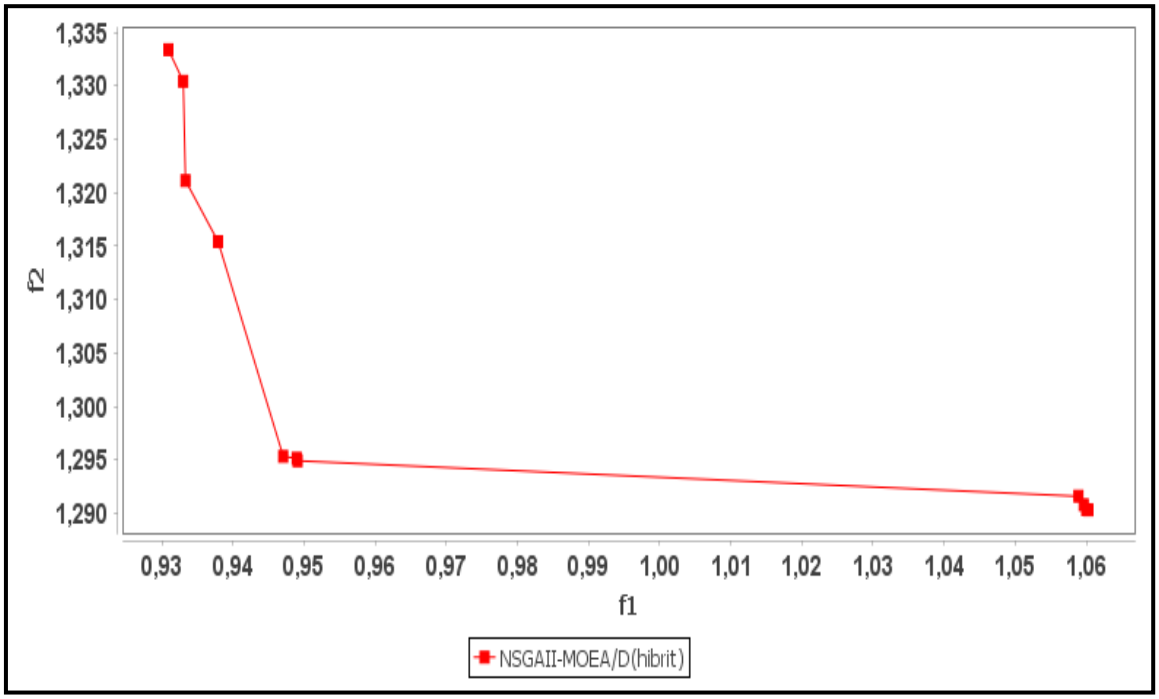
Çizelge 4.7. Literatürdeki çalışmaların sonuçları ile hibrit NSGAII-PESA2 ile elde edilen sonuçların karşılaştırılması (ikinci konfigürasyon)

Tasarım Değişkenleri	***GA[3]***	***NSGAII[4]***	***NSGAII-PESA2***
a	135,0	88,1	111,45
b	90,53	52,98	100,00
c	102,2	100,0	100,00
e	0,0	29,95	115,00
f	1,28	69,89	9,81
l	170,57	126,86	100,00
δ	1,8	3,14	2,26
Popülasyon Boyutu	400	100	200
Çaprazlama Oranı	0,6	0,9	0,9
Mutasyon Oranı	0,08	1,0	0,1
Jenerasyon Sayısı	400	150	30000
f_1(x)	1,67	0,04	1,017
f_2(x)	2,0	1,5767	1,140

Bu konfigürasyon için son olarak da hibrit NSGAII-MOEA/D algoritması uygulanmıştır. Bu konfigürasyon için uygulanan diğer hibrit algoritmalardakine benzer şekilde hibrit algoritmanın ilk kısmı olan NSGAII için başlangıç popülasyonunun büyüklüğü 200 ve maksimum jenerasyon sayısı 30000 olarak belirlenmiştir. Çaprazlama yöntemi olarak tek noktalı çaprazlama yöntemi tercih eidlmiştir ve çaprazlama oranı 0,9, dağılım indeksi ise 20,0 olarak alınmıştır. Mutasyon yöntemi olarak ise polinom mutasyon yöntemi uygulanmıştır ve mutasyon oranı 0,1, dağılım oranı ise 100,0 olarak belirlenmiştir. Hibrit algoritmanın ikinci kısmını oluşturan MOEA-D algoritması için ise çaprazlama yöntemi olarak tek noktalı çaprazlama yöntemi seçilip çaprazlama oranı 0,9, dağılım indeksi ise

20,0 olarak alınmıştır. Mutasyon yöntemi olarak da polinom mutasyon yöntemi kullanılmıştır ve mutasyon oranı 0,1, dağılım oranı ise 100,0 olarak belirlenmiştir.

NSGAII-MOEA/D hibrit algoritması ile yapılan optimizasyon çalışmasında toplam 11 adet pareto-optimal çözüm elde edilmiştir. Bu çözümlere dair Pareto-optimal eğrisi Şekil 4.15'te gösterilmektedir.



Şekil 4.15. İkinci konfigürasyon için hibrit NSGAII-MOEA/D algoritması ile elde edilen optimizasyon sonuçları

Çizelge 4.8'de literatürdeki benzer çalışmaların sonuçları ile NSGAII-MOEA/D algoritmasıyla elde edilen sonuçlar karşılaştırılmıştır. Elde edilen sonuçlarda amaç fonksiyonlarının aldığı değerlere bakıldığında hibrit NSGAII-MOEA/D algoritmasıyla elde edilen $f_1(x)$ ve $f_2(x)$ değerlerinin, referans olarak alınan çalışma olan Osyzcka ve Krenich'in [3] $f_1(x)$ ve $f_2(x)$ değerlerinden bariz bir şekilde daha iyi olduğu görülmektedir. Saravanan ve diğerlerinin [4] çalışması ile kıyaslandığında ise $f_1(x)$ değerinin, Saravanan ve diğerlerinin [4] değerlerinden daha düşük, $f_2(x)$'in ise daha iyi bir değer aldığı görülmektedir.

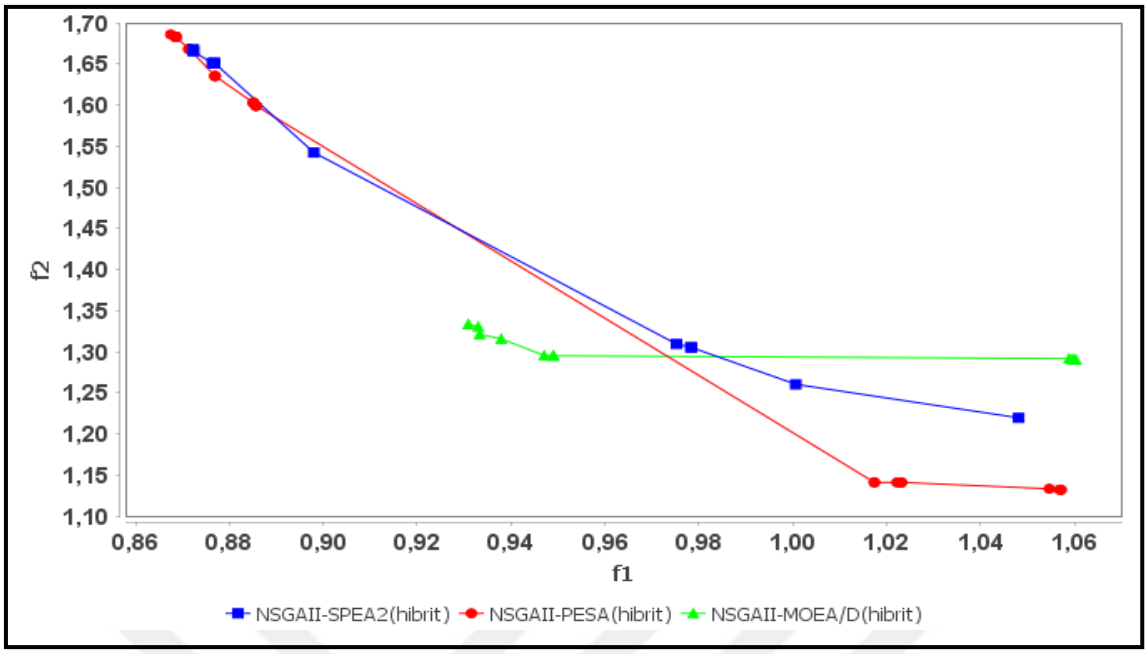
Çizelge 4.8. Literatürdeki çalışmaların sonuçları ile hibrit NSGAII-MOEA/D ile elde edilen sonuçların karşılaştırılması (ikinci konfigürasyon)

Tasarım Değişkenleri	*GA[3]*	*NSGAII[4]*	*NSGAII-MOEA/D*
a	135,0	88,1	109,06
b	90,53	52,98	99,99
c	102,2	100,0	100,00
e	0,0	29,95	2,74
f	1,28	69,89	93,14
l	170,57	126,86	122,00
δ	1,8	3,14	2,62
Popülasyon Boyutu	400	100	200
Çaprazlama Oranı	0,6	0,9	0,9
Mutasyon Oranı	0,08	1,0	0,1
Jenerasyon Sayısı	400	150	30000
$f_1(x)$	1,67	0,04	0,949
$f_2(x)$	2,0	1,5767	1,294

İkinci konfigürasyon için uygulanan üç hibrit algoritmanın sonuçlarının karşılaştırılması ise Çizelge 4.9'da verilmiştir. Şekil 4.16'da da her üç hibrit algoritmaya ait Pareto optimal eğrileri görülmektedir. Bu eğriler değerlendirildiğinde üç hibrit algoritma ile edilen sonuçların birbirine yakın olmakla birlikte NSGAII-PESA2 algoritması daha iyi sonuçların ile elde edildiği görülmektedir.

Çizelge 4.9. İkinci konfigürasyon için hibrit algoritma sonuçlarının karşılaştırılması

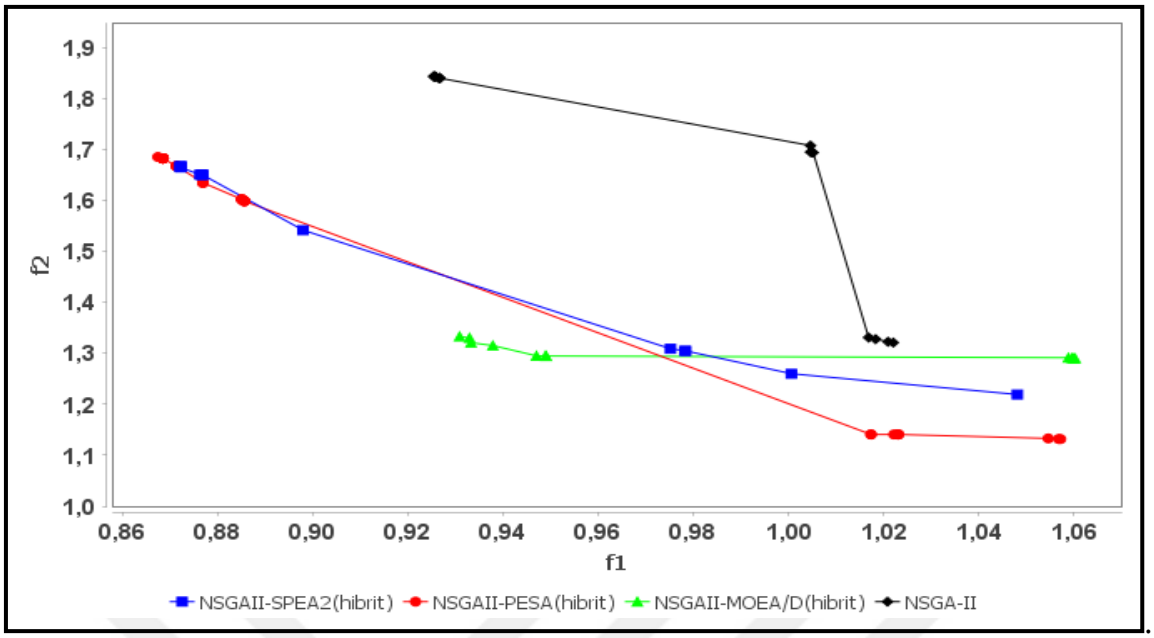
Tasarım Değişkenleri	*NSGAII-SPEA2*	*NSGAII-PESA2*	*NSGAII-MOEA/D*
a	143,05	111,45	109,06
b	87,86	100,00	99,99
c	100,03	100,00	100,00
e	49,99	115,00	2,74
f	43,93	9,81	93,14
l	126,33	100,00	122,00
δ	2,63	2,26	2,62
Popülasyon Boyutu	200	200	200
Çaprazlama Oranı	0,9	0,9	0,9
Mutasyon Oranı	0,1	0,1	0,1
Jenerasyon Sayısı	30000	30000	30000
$f_1(x)$	0,978	1,017	0,949
$f_2(x)$	1,305	1,140	1,294



Şekil 4.16. İkinci konfigürasyon için tüm hibrit algoritmaların Pareto-optimal eğrileri

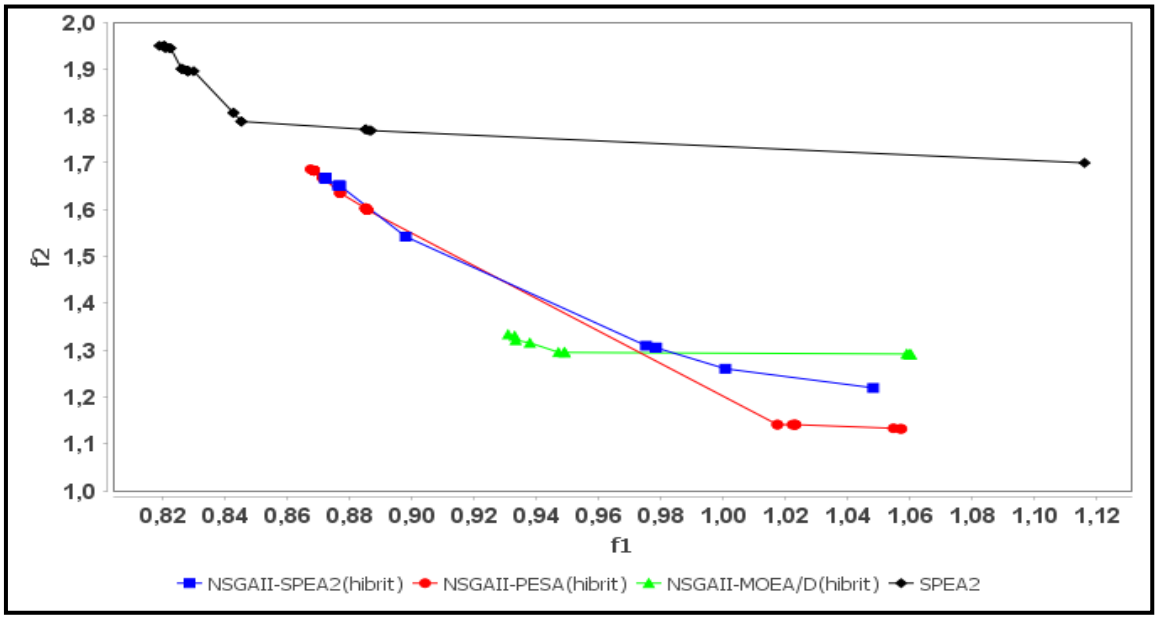
Önerilen hibrit algoritmaların performansları, hibrit olmayan NSGA-II, SPEA2, PESA2 ve MOEA/D algoritmalarının performansları ile de karşılaştırılmıştır. Bunun için hibrit olmayan algoritmalar ikinci konfigürasyon üzerinde hibrit algoritmaların çalıştığı aynı parametrelerle çalıştırılmıştır.

İlk olarak hibrit olmayan NSGA-II algoritması ikinci konfigürasyon üzerinde 30000 jenerasyon ile çalıştırılmıştır. Şekil 4.17'de tüm hibrit algoritmalar ile hibrit olmayan NSGA-II algoritmasına dair Pareto-optimal eğrileri görülmektedir. Bu eğriler incelendiğinde üç hibrit algoritmanın sonuçlarının da hibrit olmayan NSGA-II algoritmasından daha iyi olduğu görülmektedir.



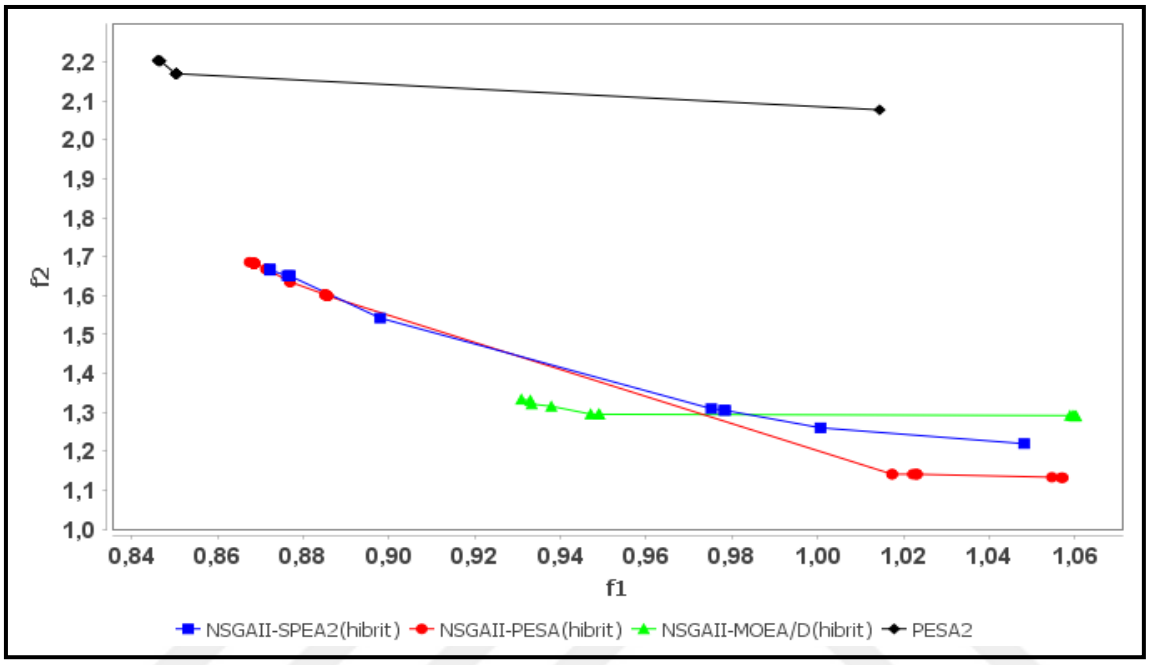
Şekil 4.17. İkinci konfigürasyon için tüm hibrit algoritmalar ile NSGA-II algoritmasıyla elde edilen sonuçlar

Şekil 4.18'de hibrit algoritmalar ile hibrit olmayan SPEA2 algoritmasına dair Pareto eğrilerinin bir karşılaştırılması verilmektedir. Buna göre her bir hibrit algoritma ile elde edilen sonuçların hibrit olmayan SPEA2 algoritmasına göre daha iyi olduğu görülmektedir.



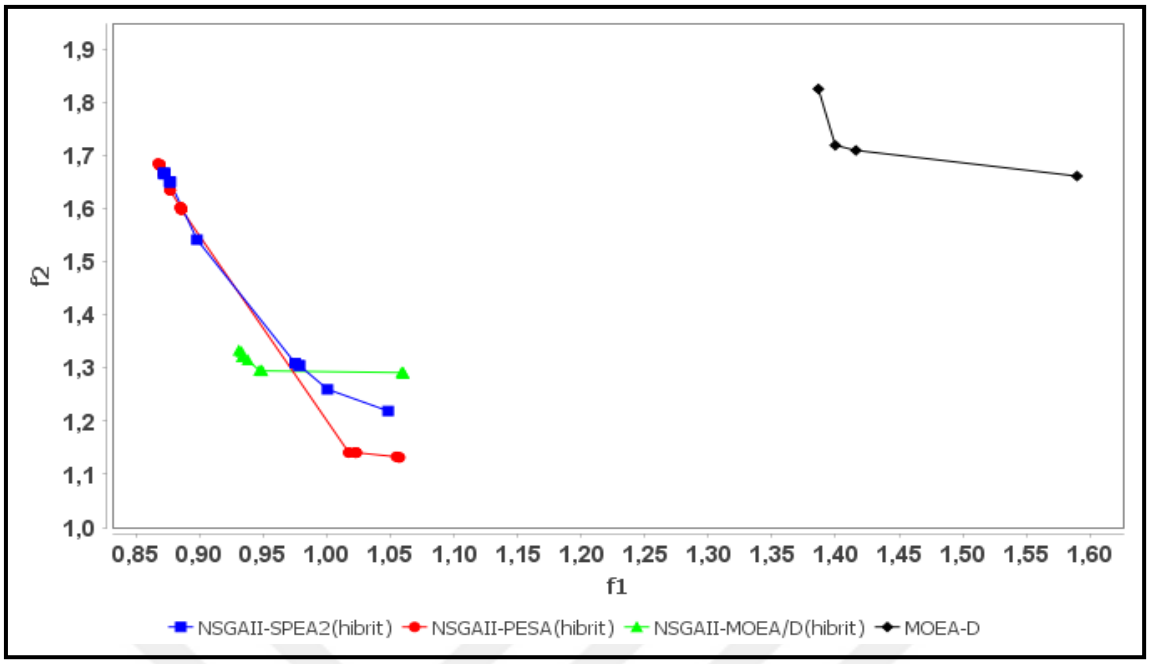
Şekil 4.18. İkinci konfigürasyon için tüm hibrit algoritmalar ile SPEA2 algoritmasıyla elde edilen sonuçlar

Hibrit algoritmaların performansı PESA2 algoritmasıyla karşılaştırılmıştır. Şekil 4.19'da hibrit algoritmalar ile PESA2 algoritmasına ait Pareto-optimal eğrileri gösterilmektedir. Bu eğriler değerlendirildiğinde üç hibrit algoritma ile de PESA2 algoritmasından daha iyi sonuçların elde edildiği görülmüştür.



Şekil 4.19. İkinci konfigürasyon için tüm hibrit algoritmalar ile PESA2 algoritmasıyla elde edilen sonuçlar

İkinci konfigürasyon üzerinde son olarak hibrit algoritmalar ile MOEA/D algoritmasının performansları karşılaştırılmıştır. Şekil 4.20'de hibrit algoritmalar ile MOEA/D algoritması ile elde edilen Pareto-optimal eğrileri incelendiğinde hibrit algoritmaların performans olarak MOEA/D algoritmasına bariz bir şekilde üstünlük kurduğu görülmüştür.

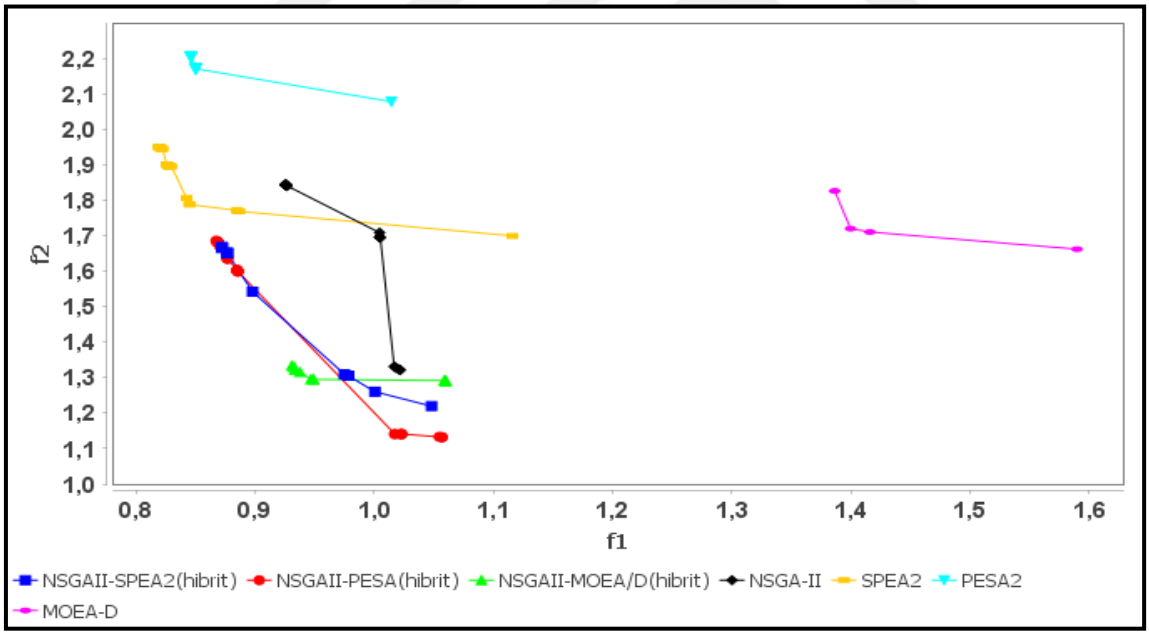


Şekil 4.20. İkinci konfigürasyon için tüm hibrit algoritmalar ile MOEA/D algoritmasıyla elde edilen sonuçlar

Çizelge 4.10'da ikinci konfigürasyon üzerinde uygulanan hibrit algoritmalar ve NSGA-II, SPEA2, PESA2 ve MOEA/D algoritmaları ile elde edilen bazı optimal çözümler gösterilmektedir. Şekil 4.21'de ise ikinci konfigürasyon üzerinde uygulanan ve yukarıda ayrı ayrı değerlendirmesi yapılan tüm hibrit algoritmalar ile NSGA-II, SPEA2, PESA2 ve MOEA/D algoritmalarının Pareto-optimal eğrileri gösterilmektedir. Buna göre, ikinci konfigürasyon üzerinde uygulanan tüm hibrit algoritmaların performansının diğer algoritmaların performanslarına göre daha iyi olduğu görülmüştür. Bu da önerilen hibrit algoritma yöntemlerinin problem üzerinde etkinliğini göstermektedir.

Çizelge 4.10. İkinci konfigürasyon için hibrit algoritmalar ile diğer algoritmalardan elde edilen bazı optimal sonuçlar

Tasarım Değişkenleri	*NSGAII-SPEA2*	*NSGAII-PESA2*	*NSGAII-MOEA/D*	*NSGA-II*	*SPEA2*	*PESA2*	*MOEA/D*
a	143,0	111,4	109,0	234,3	119,1	117,3	120,00
b	87,86	100,0	99,99	192,0	81,18	107,1	98,90
c	100,0	100,0	100,0	225,9	100,0	185,3	142,87
e	49,99	115,0	2,74	39,10	25,67	8,35	14,58
f	43,93	9,81	93,14	137,6	88,44	187,2	98,71
l	126,3	100,0	122,0	210,5	135,5	123,9	125,9
δ	2,63	2,26	2,62	2,69	2,98	3,05	2,59
Popülasyon B.	200	200	200	200	200	200	200
Çaprazlama Or.	0,9	0,9	0,9	0,9	0,9	0,9	0,9
Mutasyon Or.	0,1	0,1	0,1	0,1	0,1	0,1	0,1
Jenerasyon Say.	30000	30000	30000	30000	30000	30000	30000
$f_1(x)$	0,978	1,017	0,949	1,018	1,116	1,014	1,386
$f_2(x)$	1,305	1,140	1,294	1,327	1,699	2,077	1,825



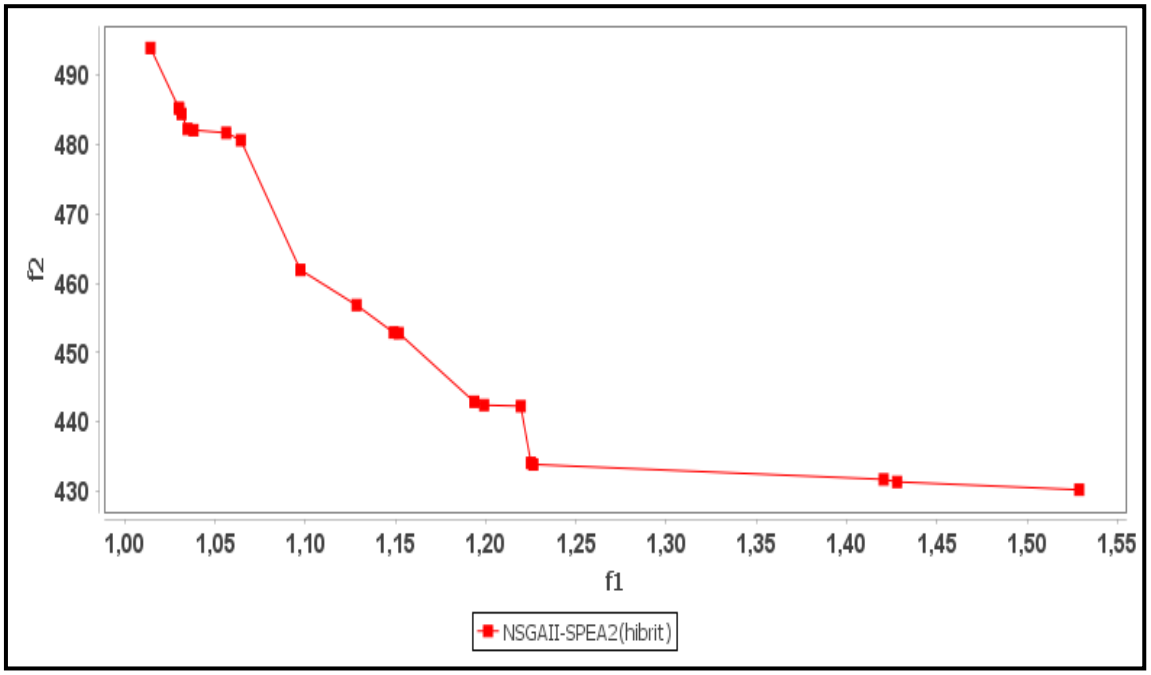
Şekil 4.21. İkinci konfigürasyon için tüm hibrit algoritmalar ile hibrit olmayan algoritmalara dair Pareto-optimal eğrileri

4.1.3. Üçüncü konfigürasyon için yapılan deneysel çalışmalar

Bu bölümde üçüncü konfigürasyon için yapılan deneysel çalışmaların sonuçları sunulmuştur ve elde edilen sonuçlar literatürdeki benzer çalışmalarla karşılaştırılmıştır.

Üçüncü konfigürasyon için ilk olarak hibrit NSGAII-SPEA2 algoritması uygulanmıştır. Hibrit algoritmanın ilk kısmı olan NSGAII için başlangıç popülasyonunun büyüklüğü 200 ve maksimum jenerasyon sayısı ise 30000 olarak belirlenmiştir. Çaprazlama işlemi için tek noktalı çaprazlama yöntemi tercih edilmiştir ve çaprazlama oranı 0,9, dağılım indeksi ise 15,0 olarak belirlenmiştir. Mutasyon işlemi için ise polinom mutasyon yöntemi kullanılmıştır ve mutasyon oranı 1/7, dağılım oranı ise 20,0 alınmıştır. Hibrit algoritmanın ikinci kısmı olan SPEA2 için ise çaprazlama yöntemi olarak tek noktalı çaprazlama yöntemi kullanılıp çaprazlama oranı 0,9, dağılım indeksi 15,0 olarak belirlenmiştir. Mutasyon yöntemi olarak ise polinom mutasyon kullanılmıştır ve mutasyon oranı 0,9, dağılım oranı 20,0 olarak alınmıştır.

Üçüncü konfigürasyon için NSGAII-SPEA2 algoritması ile toplam 20 adet optimal sonuç elde edilmiştir ve bu sonuçlara dair Pareto-optimal eğrisi Şekil 4.22'de görülmektedir.



Şekil 4.22. Üçüncü konfigürasyon için hibrit NSGAII-SPEA2 algoritması ile elde edilen optimizasyon sonuçları

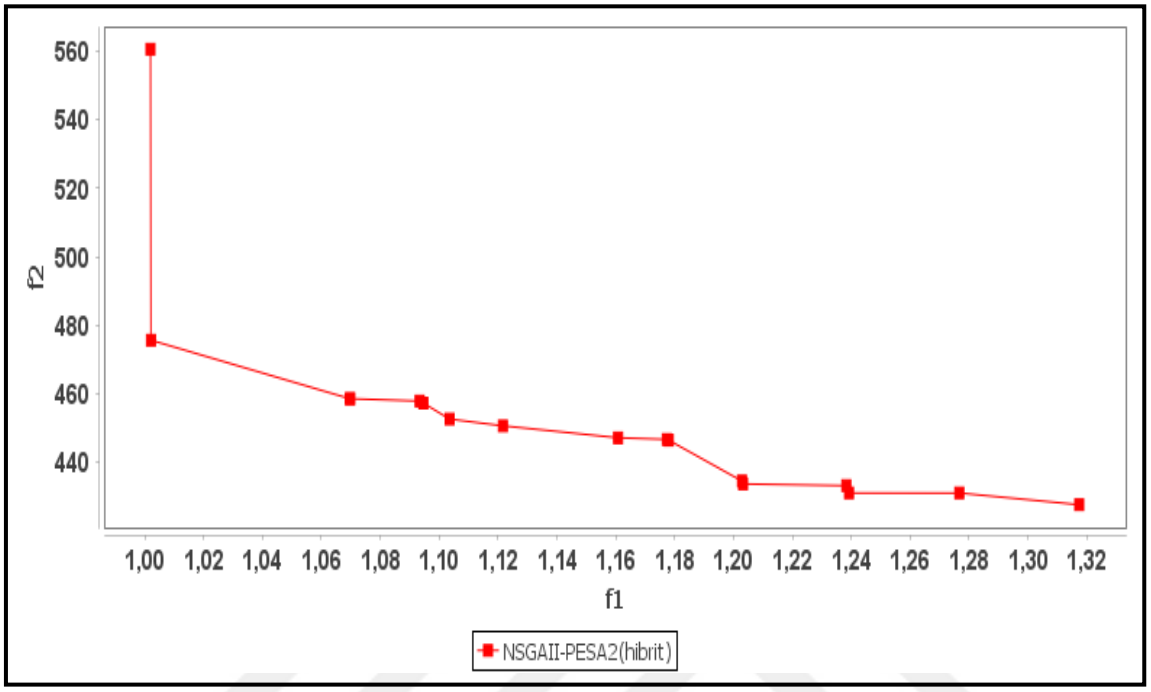
Çizelge 4.11'de ise hibrit NSGAII-SPEA2 ile elde edilen optimal sonuçların literatürdeki benzer çalışmalarla karşılaştırılması gösterilmektedir. Karşılaştırma sonuçlarında amaç fonksiyonlarının aldıkları değerlere bakıldığında hibrit NSGAII-SPEA2 algoritmasıyla elde edilen f_1(x) ve f_2(x) değerlerinin referans çalışma olan Osyzcka ve Krenich'in [3] çalışmasındaki değerlerden bariz bir şekilde iyi olduğu gözlemlenmiştir. Saravanan ve diğerlerinin [4] çalışmasıyla kıyaslandığında ise hibrit algoritma ile elde edilen f_2(x) değeri Saravanan ve diğerlerinin [4] çalışmasındaki f_2(x) değerinden daha iyi iken, f_1(x) değerinin, Saravanan ve diğerlerinin [4] çalışmasındaki f_1(x) değerinden daha düşük olduğu görülmektedir. Bunun sebebi, Saravanan ve diğerlerinin [4] çalışmasında kullanılan farklı konfigürasyonlardır.

Çizelge 4.11. Literatürdeki çalışmaların sonuçları ile hibrit NSGAII-SPEA2 ile elde edilen sonuçların karşılaştırılması (üçüncü konfigürasyon)

Tasarım Değişkenleri	***GA[3]***	***NSGAII[4]***	***NSGAII-SPEA2***
a	135,0	88,1	113,66
b	90,53	52,98	100,74
c	102,2	100,0	100,36
e	0,0	29,95	8,89
f	1,28	69,89	11,11
l	170,57	126,86	118,12
δ	1,8	3,14	1,79
Popülasyon Boyutu	400	100	200
Çaprazlama Oranı	0,6	0,9	0,9
Mutasyon Oranı	0,08	1,0	1/7
Jenerasyon Sayısı	400	150	30000
f_1(x)	1,67	0,04	1,14
f_2(x)	499,81	467,88	452,92

Üçüncü konfigürasyon için ikinci olarak hibrit NSGAII-PESA2 algoritması uygulanmıştır. Bu algoritmada da bir önceki hibrit algoritmadaki gibi NSGAII için başlangıç popülasyonunun değeri 200 ve maksimum jenerasyon sayısı 30000 alınmıştır Çaprazlama yöntemi olarak tek noktalı çaprazlama yöntemi kullanılmıştır ve çaprazlama oranı 0,9, dağılım indeksi ise 15,0 olarak alınmıştır. Mutasyon yöntemi olarak ise polinom mutasyon yöntemi uygulanmıştır ve mutasyon oranı 1/7, dağılım oranı ise 20,0 olarak belirlenmiştir. Hibrit algoritmanın ikinci kısmını oluşturan PESA2 algoritması için ise çaprazlama yöntemi olarak tek noktalı çaprazlama yöntemi seçilip çaprazlama oranı 0,9, dağılım indeksi ise 15,0 olarak alınmıştır. Mutasyon yöntemi olarak da polinom mutasyon yöntemi kullanılmıştır ve mutasyon oranı 1,0, dağılım oranı ise 20,0 olarak belirlenmiştir.

Yapılan optimizasyon çalışması sonucunda 17 adet optimal sonuç elde edilmiştir ve elde edilen sonuçlara dair Pareto-optimal eğrisi Şekil 4.23'te görülmektedir.



Şekil 4.23. Üçüncü konfigürasyon için hibrit NSGAII-PESA2 algoritması ile elde edilen optimizasyon sonuçları

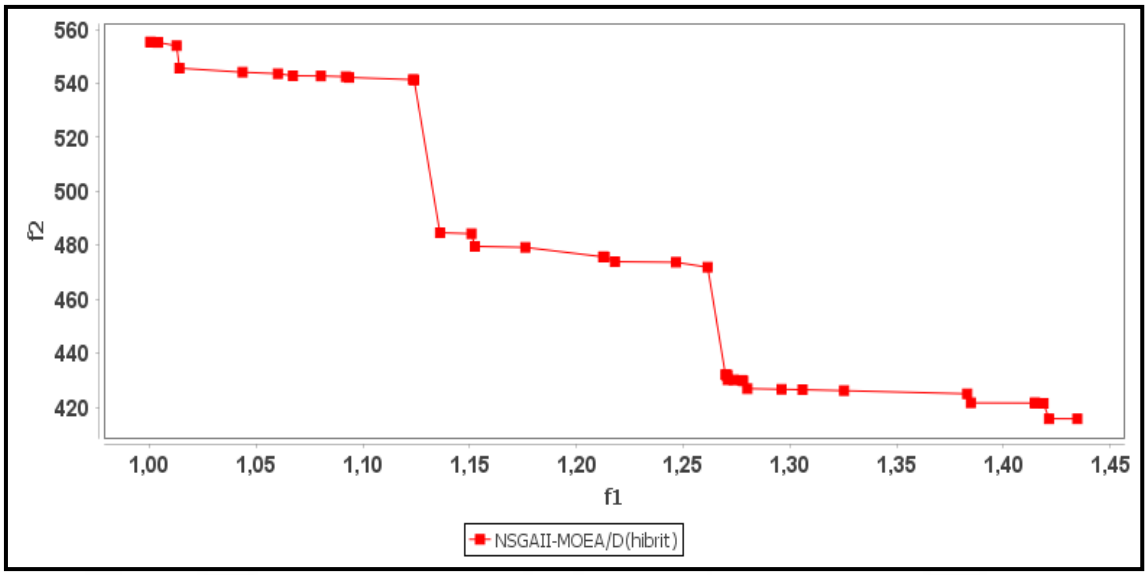
Çizelge 4.12'de de hibrit NSGAII-PESA2 algoritması ile elde edilen optimal değerler, literatürdeki benzer çalışmaların sonuçları ile karşılaştırılmıştır. Elde edilen optimizasyon sonuçları amaç fonksiyonları bazında değerlendirildiğinde elde edilen $f_1(x)$ ve $f2(x)$ değerleri, referans çalışma olan Osyczka ve Krenich'in [3] çalışmasındaki değerlerden açık bir şekilde üstündür. Saravanan ve diğerlerinin [4] çalışmasıyla kıyaslandığında ise $f_2(x)$ değerinin yine daha iyi olduğu gözlemlenirken $f_1(x)$ değerinin daha düşük olduğu görülmüştür. Bunun sebebi de bir önceki hibrit algoritma sonuçları içerisinde açıklanan Saravanan ve arkadaşlarının [4] çalışmasındaki farklı konfigürasyonlardır.

Çizelge 4.12. Literatürdeki çalışmaların sonuçları ile hibrit NSGAII-PESA2 ile elde edilen sonuçların karşılaştırılması (üçüncü konfigürasyon)

Tasarım Değişkenleri	***GA[3]***	***NSGAII[4]***	***NSGAII-PESA2***
a	135,0	88,1	114,10
b	90,53	52,98	108,12
c	102,2	100,0	100,00
e	0,0	29,95	1,38
f	1,28	69,89	10,00
l	170,57	126,86	124,93
δ	1,8	3,14	1,72
Popülasyon Boyutu	400	100	200
Çaprazlama Oranı	0,6	0,9	0,9
Mutasyon Oranı	0,08	1,0	1/7
Jenerasyon Sayısı	400	150	30000
$f_1(x)$	1,67	0,04	1,06
$f_2(x)$	499,81	467,88	458,54

Üçüncü konfigürasyon için son olarak da hibrit NSGAII-MOEA/D algoritması uygulanmıştır. Bunun için hibrit algoritmanın başlangıcını oluşturan NSGAII için başlangıç popülasyonunun büyüklüğü 200 ve maksimum jenerasyon sayısı 30000 olarak belirlenmiştir. Çaprazlama yöntemi olarak tek noktalı çaprazlama yöntemi tercih edilip çaprazlama oranı 0,9, dağılım indeksi ise 15,0 olarak alınmıştır. Mutasyon yöntemi olarak da polinom mutasyon yöntemi kullanılmıştır ve mutasyon oranı 1/7, dağılım oranı da 20,0 olarak belirlenmiştir. Hibrit algoritmanın ikinci kısmını oluşturan MOEA/D algoritması için de çaprazlama yöntemi olarak aynı şekilde tek noktalı çaprazlama yöntemi kullanılıp çaprazlama oranı 0,9, dağılım indeksi 15,0 olarak belirlenmiştir. Mutasyon yöntemi olarak da polinom mutasyon yöntemi tercih edilmiş olup mutasyon oranı 1,0, dağılım oranı da 20,0 olarak belirlenmiştir.

Yapılan optimizasyon çalışması sonucunda toplam 44 adet optimal çözüm elde edilmiştir. Elde edilen bu çözümlere dair Pareto-optimal eğrisi Şekil 4.24'te gösterilmektedir.



Şekil 4.24. Üçüncü konfigürasyon için hibrit NSGAII-MOEA/D algoritması ile elde edilen optimizasyon sonuçları

Çizelge 4.13'te ise elde edilen sonuçlar, literatürdeki benzer çalışmaların sonuçları ile kıyaslanmıştır. Elde edilen sonuçlarda amaç fonksiyonlarının aldığı değerler karşılaştırıldığında hibrit NSGAII-MOEA/D algoritması ile elde edilen $f_1(x)$ ve $f_2(x)$ değerlerinin, referans çalışma olan Osyczka ve Krenich'in [3] çalışmasındaki değerlerden daha iyi olduğu gözlemlenmiştir. Saravanan ve diğerlerinin [4] çalışması ile kıyaslandığında ise hibrit algoritma ile elde edilen $f_2(x)$ değerinin daha iyi olduğu gözlemlenirken $f_1(x)$ değerinin daha düşük çıktığı görülmektedir.

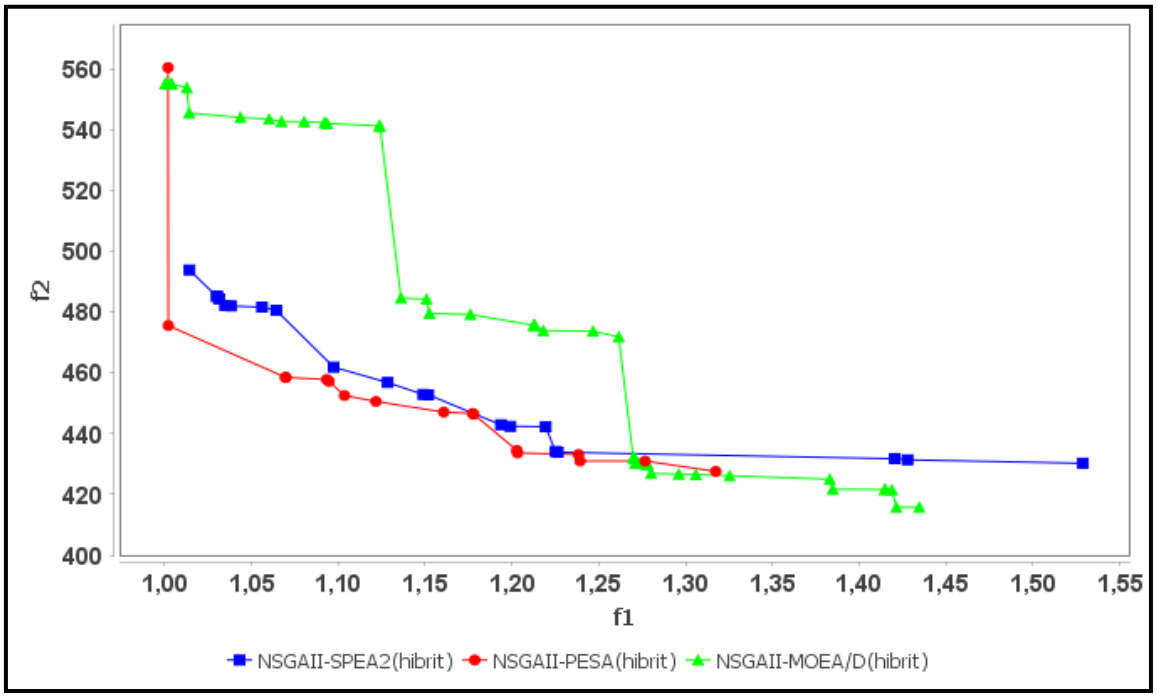
Çizelge 4.13. Literatürdeki çalışmaların sonuçları ile hibrit NSGAII-MOEA/D ile elde edilen sonuçların karşılaştırılması (üçüncü konfigürasyon)

Tasarım Değişkenleri	***GA[3]***	***NSGAII[4]***	***NSGAII-MOEA/D***
a	135,0	88,1	102,5
b	90,53	52,98	90,75
c	102,2	100,0	100,27
e	0,0	29,95	8,91
f	1,28	69,89	18,40
l	170,57	126,86	106,07
δ	1,8	3,14	1,9
Popülasyon Boyutu	400	100	200
Çaprazlama Oranı	0,6	0,9	0,9
Mutasyon Oranı	0,08	1,0	1/7
Jenerasyon Sayısı	400	150	30000
$f_1(x)$	1,67	0,04	1,28
$f_2(x)$	499,81	467,88	426,94

Üçüncü konfigürasyon için uygulanan üç hibrit algoritmanın sonuçlarının karşılaştırılması ise Çizelge 4.14'te gösterilmektedir. Şekil 4.25'te her üç hibrit algoritma ile edilen pareto-optimal eğrileri görülmektedir. Pareto eğrileri karşılaştırıldığında üçüncü konfigürasyon için en iyi sonuçların hibrit NSGAII-PESA2 algoritması ile elde edildiği görülmektedir.

Çizelge 4.14. Üçüncü konfigürasyon için hibrit algoritma sonuçlarının karşılaştırılması

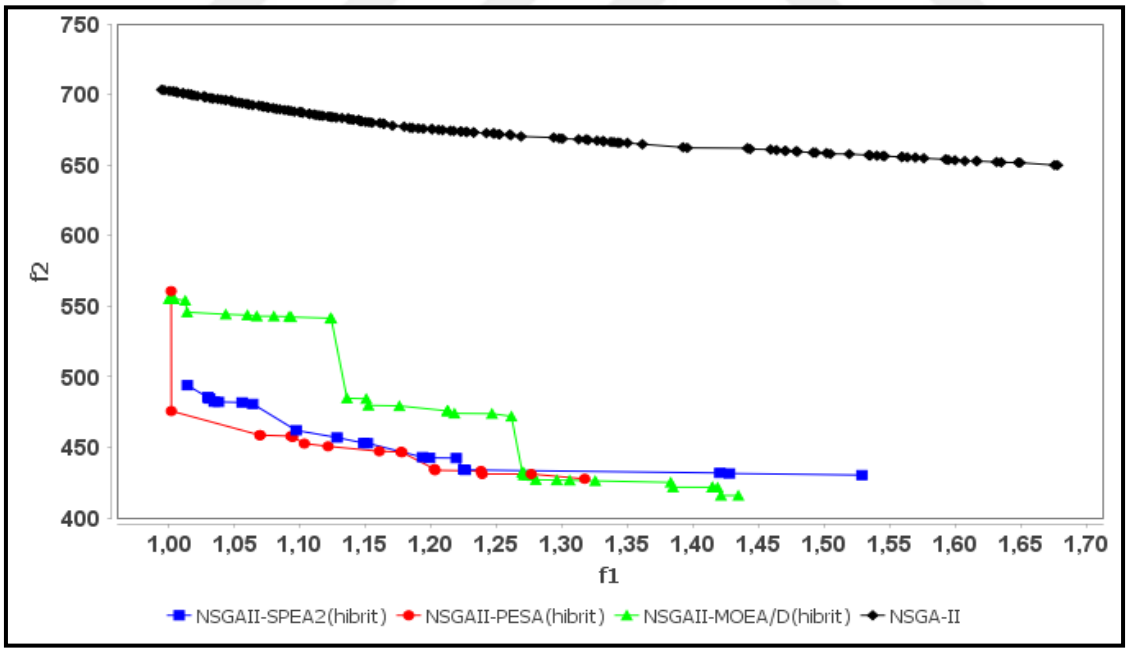
Tasarım Değişkenleri	***NSGAII-SPEA2***	***NSGAII-PESA2***	***NSGAII-MOEA/D***
a	113,66	114,10	102,5
b	100,74	108,12	90,75
c	100,36	100,00	100,27
e	8,89	1,38	8,91
f	11,11	10,00	18,40
l	118,12	124,93	106,07
δ	1,79	1,72	1,9
Popülasyon Boyutu	200	200	200
Çaprazlama Oranı	0,9	0,9	0,9
Mutasyon Oranı	1/7	1/7	1/7
Jenerasyon Sayısı	30000	30000	30000
$f_1(x)$	1,14	1,06	1,28
$f_2(x)$	452,92	458,54	426,94



Şekil 4.25. Üçüncü konfigürasyon için tüm hibrit algoritmaların Pareto-optimal eğrileri

Üçüncü konfigürasyon üzerinde hibrit algoritmaların performanslarını daha iyi değerlendirmek amacıyla çok amaçlı optimizasyon algoritmalarından NSGA-II, SPEA2, PESA2 ve MOEA/D algoritmaları üçüncü konfigürasyon üzerinde hibrit algoritmalar ile eşit şartlarda uygulanmıştır. Bu algoritmalarda çaprazlama işlemi için çaprazlama oranı 0,9, dağılım indeksi ise 15,0 kullanılmıştır. Mutasyon işlemi için ise mutasyon oranı 1/7, dağılım oranı ise 20,0 parametreleri kullanılmıştır. Söz konusu algoritmalar bu parametre değerleriyle hibrit algoritmalarda olduğu gibi 30000 jenerasyon ile üçüncü konfigürasyon üzerinde çalıştırılmıştır.

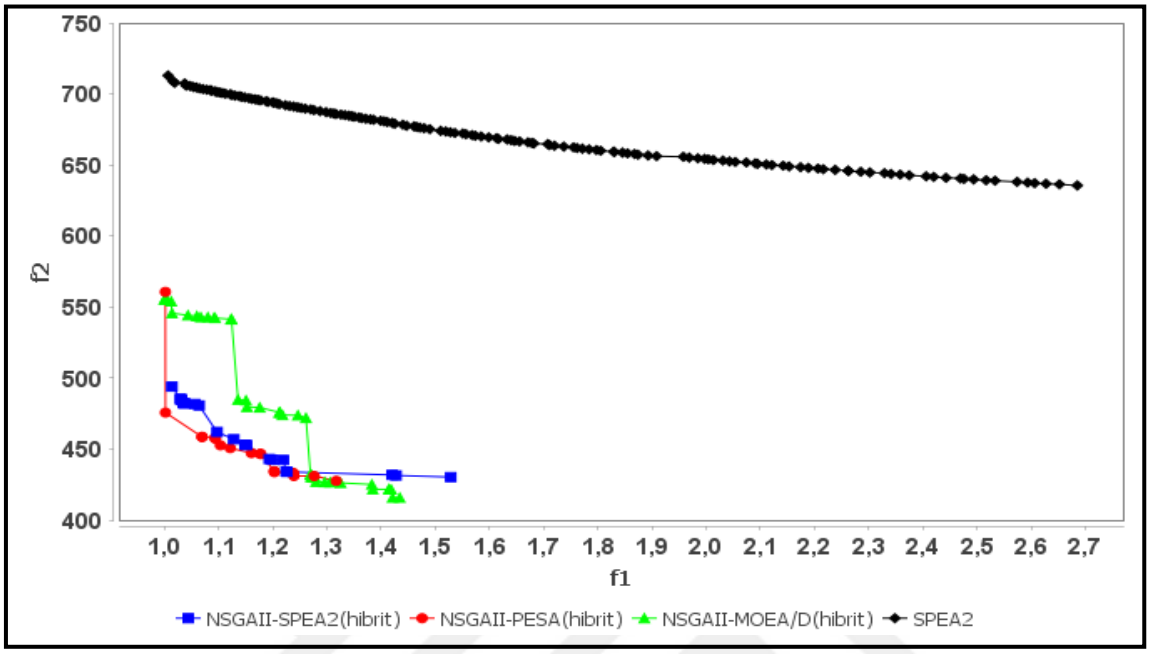
İlk olarak hibrit algoritmalar ile NSGA-II algoritmasının performansı karşılaştırılmıştır. Şekil 4.26'da hibrit algoritmalar ile NSGA-II algoritmasına dair Pareto-optimal eğrileri gösterilmektedir. NSGA-II algoritması, hibrit algoritmalar ile eşit koşullarda çalıştırıldığında 150 adet pareto-optimal çözüm elde edilmiştir. Pareto-optimal eğrileri değerlendirildiğinde her bir hibrit algoritma ile elde edilen sonuçların NSGA-II algoritmasına göre daha iyi olduğu görülmektedir.



Şekil 4.26. Üçüncü konfigürasyon için tüm hibrit algoritmalar ile NSGA-II algoritmasıyla elde edilen sonuçlar

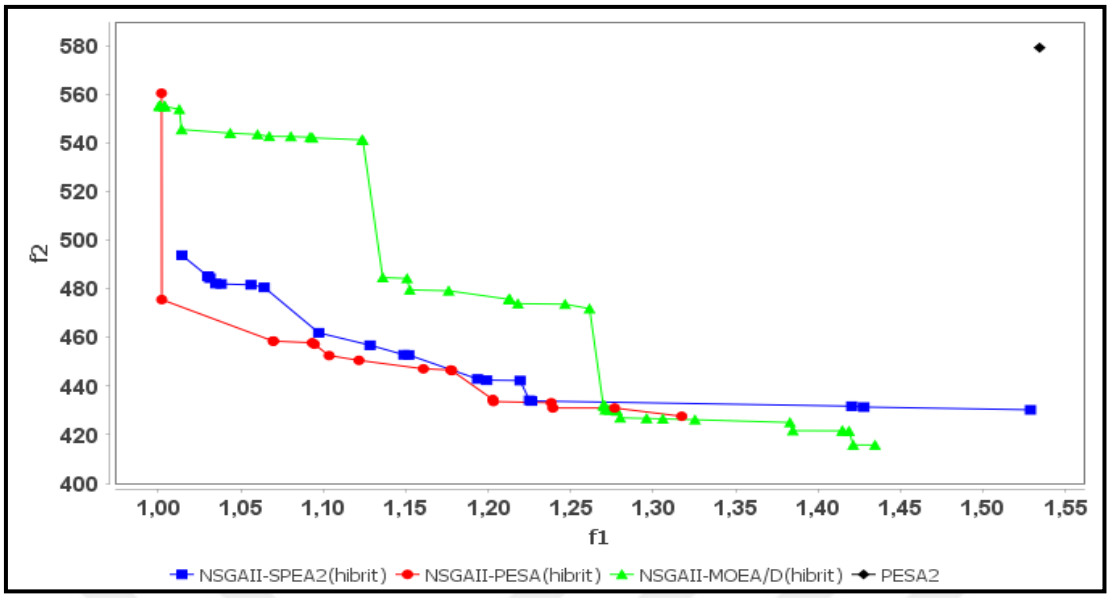
İkinci karşılaştırma olarak hibrit algoritmalar ile SPEA2 algoritmasının performansı değerlendirilmiştir. SPEA2 algoritması, hibrit algoritmalar ile eşit koşullarda çalıştırıldığında 150 adet pareto-optimal çözüm elde edilmiştir. Şekil 4.27'de hibrit

algoritmalar ile SPEA2 algoritmasına ait Pareto-optimal eğrileri gösterilmektedir. Bu eğriler incelendiğinde tüm hibrit algoritmaların performansının SPEA2 algoritmasının performansına göre daha iyi olduğu görülmüştür.



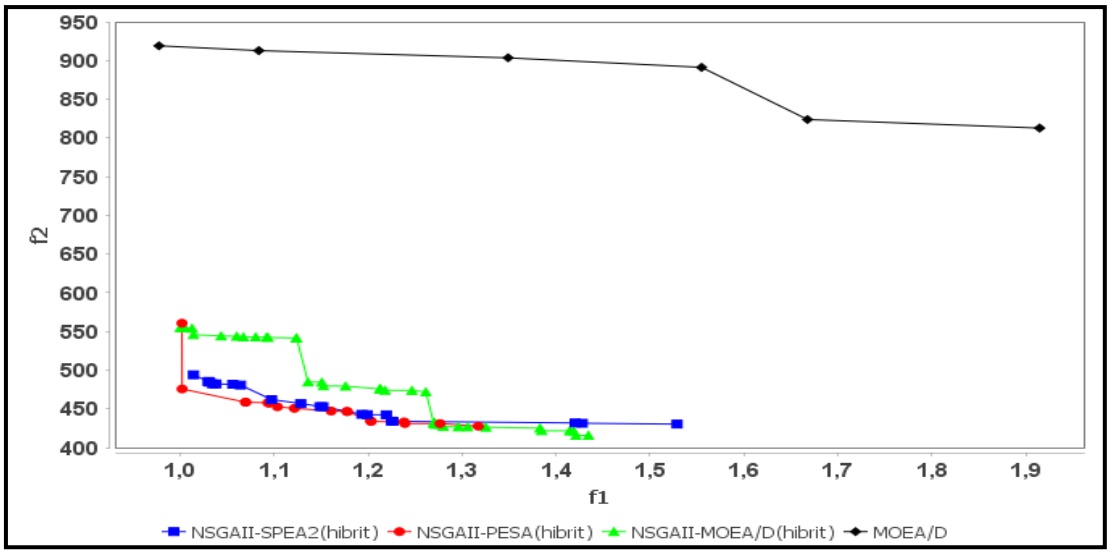
Şekil 4.27. Üçüncü konfigürasyon için tüm hibrit algoritmalar ile SPEA2 algoritmasıyla elde edilen sonuçlar

Üçüncü olarak hibrit algoritmalar ile PESA2 algoritmasının performansları değerlendirilmiştir. PESA2 algoritması, hibrit algoritmalar ile eşit koşullarda çalıştırıldığında 1 adet pareto-optimal çözüm elde edilmiştir. Şekil 4.28'de hibrit algoritmalar ile PESA2 algoritmasının Pareto-optimal eğrileri gösterilmektedir. Buna göre hibrit algoritmaların, PESA2 algoritmasına göre bariz bir şekilde daha iyi sonuçlar verdiği görülmektedir.



Şekil 4.28. Üçüncü konfigürasyon için tüm hibrit algoritmalar ile PESA2 algoritmasıyla elde edilen sonuçlar

Üçüncü konfigürasyon üzerinde son olarak hibrit algoritmalar ile MOEA/D algoritmasının performansı karşılaştırılmıştır. MOEA/D algoritması, hibrit algoritmalar ile eşit koşullarda çalıştırıldığında 6 adet pareto-optimal çözüm elde edilmiştir. Şekil 4.29'da hibrit algoritmalar ile MOEA/D algoritmasına ait Pareto-optimal eğrileri gösterilmektedir. Bu eğriler değerlendirildiğinde hibrit algoritmalar ile MOEA/D algoritmasına göre daha iyi sonuçlar elde edildiği gözlemlenmiştir.

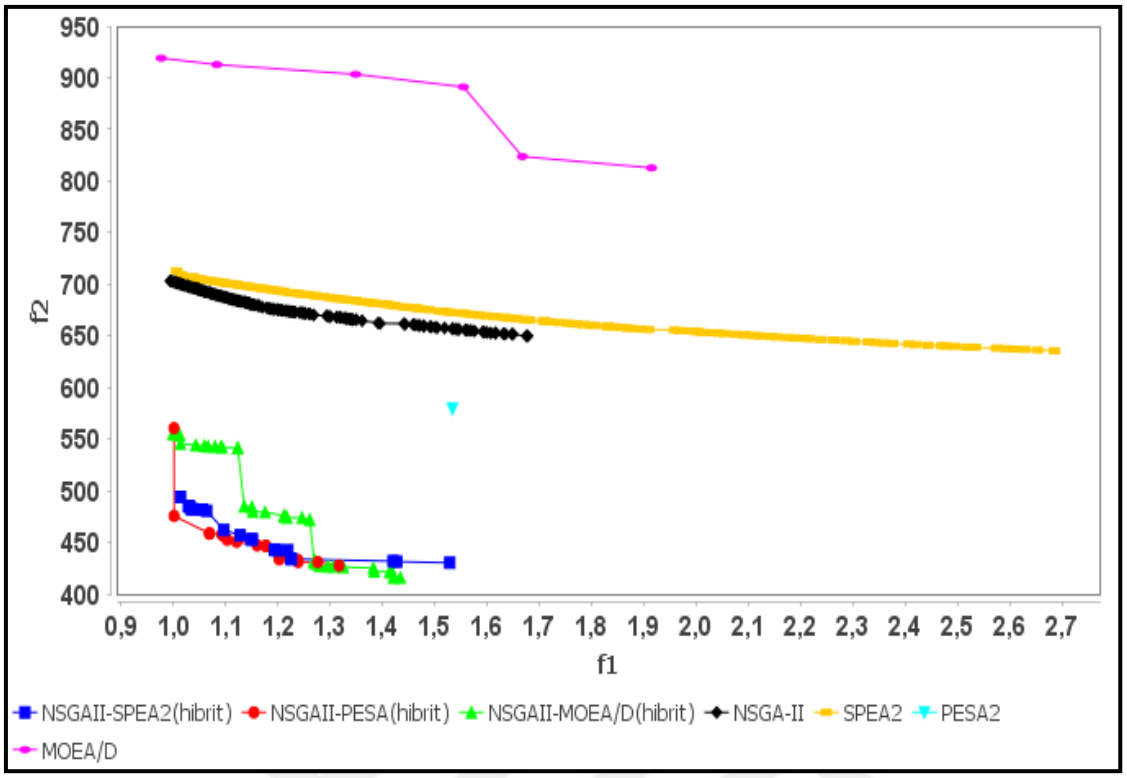


Şekil 4.29. Üçüncü konfigürasyon için tüm hibrit algoritmalar ile MOEA/D algoritmasıyla elde edilen sonuçlar

Çizelge 4.15'te üçüncü konfigürasyon üzerinde uygulanan hibrit algoritmalar ile NSGA-II, SPEA2, PESA2 ve MOEA/D algoritmalarından elde edilen bazı optimal sonuçlar gösterilmektedir. Şekil 4.30'da da yukarıda ayrı ayrı değerlendirilmesi yapılan hibrit algoritmalar ile NSGA-II, SPEA2, PESA2 ve MOEA/D algoritmalarına dair Pareto-optimal eğrileri gösterilmektedir. Yapılan tüm bu değerlendirmeler sonucunda üçüncü konfigürasyon üzerinde önerilen hibrit algoritmalar ile NSGA-II, SPEA2, PESA2 ve MOEA/D algoritmalarından daha iyi sonuçlar elde edildiği görülmüştür. Bu durum, önerilen hibrit yöntemlerin üçüncü konfigürasyon üzerindeki etkinliğini göstermektedir.

Çizelge 4.15. Üçüncü konfigürasyon için hibrit algoritmalar ile diğer algoritmalar ile elde edilen bazı optimal çözümler

Tasarım Değişkenleri	***NSGAII-SPEA2***	***NSGAII-PESA2***	***NSGAII-MOEA/D***	***NSGA-II***	***SPEA2***	***PESA2***	***MOEA/D***
a	113,66	114,1	102,5	183,1	196,3	142,1	139,5
b	100,74	108,1	90,75	141,9	130,0	102,1	103,0
c	100,36	100,0	100,2	114,2	100,0	101,1	262,6
e	8,89	1,38	8,91	6,83	21,37	12,96	27,05
f	11,11	10,00	18,40	14,84	18,00	21,41	250,0
l	118,12	124,9	106,0	218,8	230,7	199,5	130,4
δ	1,79	1,72	1,9	1,90	2,08	2,29	3,14
Popülasyon Boy.	200	200	200	200	200	200	200
Çaprazlama Or.	0,9	0,9	0,9	0,9	0,9	0,9	0,9
Mutasyon Or.	1/7	1/7	1/7	1/7	1/7	1/7	1/7
Jenerasyon Say.	30000	30000	30000	30000	30000	30000	30000
$f_1(x)$	1,14	1,06	1,28	1,15	1,16	1,53	1,08
$f_2(x)$	452,92	458,5	426,9	679,9	696,4	579,3	912,7



Şekil 4.30. Üçüncü konfigürasyon için tüm hibrit algoritmalar ile hibrit olmayan algoritmalara dair Pareto-optimal eğrileri

Tüm konfigürasyonlar üzerinde yapılan optimizasyon çalışmaları değerlendirildiğinde robot tutucu tasarım problemi için en uygun konfigürasyonun birinci konfigürasyon olduğu sonucuna varılmıştır. Bunun nedeni, uygulanan üç hibrit algoritma için de daha fazla sayıda pareto-optimal sonuç elde edilebilmesi ve dolayısıyla bilinen en iyi tasarım değerlerine ulaşmada birinci konfigürasyonun diğer konfigürasyonlara göre daha verimli bir konfigürasyon olmasıdır.

4.2. Performans Değerlendirmesi

Bu bölümde robot tutucuların tasarım problemini çözmek amacıyla kullanılan hibrit NSGAII-SPEA2, NSGAII-PESA2 ve NSGAII-MOEAD algoritmaları performans bakımından karşılaştırılmıştır. Her bir konfigürasyon için çalışmanın sonuçları, çok amaçlı optimizasyon yöntemlerinin değerlendirilmesinde sıklıkla kullanılan hypervolume ve inverted generational distance metriklerine göre değerlendirilmiştir.

4.2.1. Birinci konfigürasyon için performans değerlendirilmesi

Birinci konfigürasyon için uygulanan üç hibrit algoritma hypervolume ve inverted generational distance metrikleri ile çalışma zamanı üzerinden değerlendirilmiştir. Bunun için her bir algoritma 30000 iterasyon ile toplam 30 kez çalıştırılmıştır.

Çizelge 4.16'da hypervolume metriğine göre üç hibrit algoritmanın en küçük, en büyük, ortalama, medyan ve standart sapma değerleri gösterilmektedir. Bu sonuçlar değerlendirildiğinde standart sapması en küçük olan hibrit algoritmanın 0,073 değeriyle NSGAII-MOEA/D algoritması olduğu görülmektedir. Bu durum, NSGAII-MOEA/D algoritması ile elde edilen çözümlerin daha düzgün ve dengeli bir dağılım gösterdiğini belirtmektedir.

Çizelge 4.16. Hypervolume metriğine göre istatistiksel sonuçlar (birinci konfigürasyon için)

	NSGAII-SPEA2	***NSGAII-PESA2***	***NSGAII-MOEA/D***
Min	0,018	0,344	0,576
Max	0,630	0,942	0,893
Median	0,408	0,792	0,785
Mean	0,452	0,781	0,784
Standart Deviation	0,151	0,118	0,073

Çizelge 4.17'de ise inverted generational distance metriğine göre bir performans değerlendirilmesi yapılmıştır. Buna göre standart sapma değerlerine bakıldığında NSGAII-MOEA/D algoritmasının standart sapma değerinin diğer iki algoritmaya oranla daha düşük olduğu gözlemlenmektedir. Dolayısıyla inverted generational distance metriğine göre de hibrit NSGAII-MOEA/D algoritmasıyla elde edilen veriler daha düzenli bir dağılım göstermektedir.

Çizelge 4.17. Inverted generational distance metriğine göre istatistiksel sonuçlar (birinci konfigürasyon için)

	NSGAII-SPEA2	***NSGAII-PESA2***	***NSGAII-MOEA/D***
Min	0,108	0,032	0,071
Max	0,587	0,376	0,241
Median	0,254	0,151	0,144
Mean	0,268	0,174	0,146
Standart Deviation	0,110	0,072	0,032

Çizelge 4.18'te üç hibrit algoritma çalışma zamanları açısından değerlendirilmiştir. Buna göre hibrit NSGAII-PESA2 algoritmasının diğer iki hibrit algoritmaya göre çalışma zamanının daha kısa olduğu görülmektedir.

Çizelge 4.18. Çalışma zamanına göre istatistiksel sonuçlar (birinci konfigürasyon için)

	NSGAII-SPEA2(s)	***NSGAII-PESA2(s)***	***NSGAII-MOEA/D(s)***
Min	6,99	3,571	9,734
Max	18,298	12,584	17,563
Median	11,571	7,878	8,641
Mean	12,292	8,033	9,734
Standart Deviation	3,389	2,341	0,032

4.2.2. İkinci konfigürasyon için performans değerlendirilmesi

İkinci konfigürasyon için uygulanan üç hibrit algoritmanın performansı hypervolume ve inverted generational distance metrikleri ile çalışma zamanı üzerinden değerlendirilmiştir. Bunun için her bir algoritma 30000 iterasyon ile toplam 30 kez çalıştırılmıştır.

Çizelge 4.19'da hypervolume metriğine göre üç hibrit algoritmanın performansı karşılaştırılmıştır. Bu karşılaştırmaya göre üç hibrit algoritmanın ikinci konfigürasyon üzerindeki performans dağılımlarının birbirine çok yakın olduğu görülmektedir. Hibrit NSGAII-MOEA/D algoritması, diğer iki hibrit algoritmaya göre ortalamaya daha yakın bir dağılım göstermiştir. Standart sapmalar açısından bakıldığında ise NSGAII-SPEA2 ve NSGAII-PESA2 algoritmalarından elde edilen çözümlerin NSGAII-MOEAD algoritmasından elde edilen çözümlere göre bir nebze daha düzenli bir dağılımı olduğu söylenebilmektedir.

Çizelge 4.19. Hypervolume metriğine göre istatistiksel sonuçlar (ikinci konfigürasyon için)

	NSGAII-SPEA2	***NSGAII-PESA2***	***NSGAII-MOEA/D***
Min	0,0	0,0	0,0
Max	0,424	0,358	0,539
Median	0,208	0,103	0,127
Mean	0,209	0,131	0,123
Standart Deviation	0,120	0,120	0,125

Çizelge 4.20'de ise ikinci konfigürasyon için üç hibrit algoritmanın performans değerlendirilmesi inverted generational distance metriğine göre yapılmıştır. Bu değerlendirmeye bakıldığında ise standart sapmaların aldıkları değerlere göre hibrit NSGAII-MOEA/D algoritması diğer iki hibrit algoritmaya göre daha düşük bir değer aldığı görülmektedir. Dolayısıyla hibrit NSGAII-MOEA/D algoritması ile elde edilen çözümler daha düzenli ve daha az riskli bir dağılım göstermektedir.

Çizelge 4.20. Inverted generational distance metriğine göre istatistiksel sonuçlar (ikinci konfigürasyon için)

	NSGAII-SPEA2	***NSGAII-PESA2***	***NSGAII-MOEA/D***
Min	0,078	0,209	0,134
Max	0,543	0,901	0,603
Median	0,210	0,495	0,294
Mean	0,252	0,507	0,317
Standart Deviation	0,136	0,151	0,102

Çizelge 4.21'de ise ikinci konfigürasyon için üç hibrit algoritmanın performans değerlendirilmesi çalışma zamanına göre yapılmıştır. Elde edilen sonuçlara göre hibrit NSGAII-SPEA2 algoritmasının en hızlı çalışan algoritma olduğu görülmektedir.

Çizelge 4.21. Çalışma zamanına göre istatistiksel sonuçlar (ikinci konfigürasyon için)

	NSGAII-SPEA2(s)	***NSGAII-PESA2(s)***	***NSGAII-MOEA/D(s)***
Min	7,10	12,191	13,898
Max	24,701	22,811	21,839
Median	15,807	17,077	17,619
Mean	15,803	17,623	17,514
Standart Deviation	2,823	2,935	2,411

4.2.3. Üçüncü konfigürasyon için performans değerlendirilmesi

Üçüncü konfigürasyon için uygulanan üç hibrit algoritmanın performansı hypervolume ve inverted generational distance metrikleri ile çalışma zamanına göre değerlendirilmiştir. Bunun için her bir algoritma 30000 iterasyonla birlikte toplam 30 kez çalıştırılmıştır.

İlk olarak üç hibrit algoritmanın üçüncü konfigürasyon üzerindeki performansı hypervolume metriği üzerinden değerlendirilmiştir (Çizelge 4.22). Buna göre, hibrit NSGAII-PESA2 algoritması standart sapma değerleri açısından daha düşük bir değer almaktadır. Dolayısıyla hibrit NSGAII-PESA2 algoritması ile elde edilen çözümler daha düzenli bir sağılım göstermektedir. Bunun dışında standart sapma ile ortalamalar arasındaki farka bakıldığında NSGAII-MOEAD algoritması ile elde edilen verilerin diğer iki algoritma ile elde edilen çözümlere kıyasla ortalamaya daha yakın bir dağılım gösterdikleri gözlemlenmiştir.

Çizelge 4.22. Hypervolume metriğine göre istatistiksel sonuçlar (üçüncü konfigürasyon için)

	NSGAII-SPEA2	***NSGAII-PESA2***	***NSGAII-MOEA/D***
Min	0,030	0,193	0,0
Max	0,481	0,419	0,508
Median	0,346	0,312	0,183
Mean	0,324	0,313	0,180
Standart Deviation	0,078	0,053	0,121

Üçüncü konfigürasyon üzerinde uygulanan üç hibrit algoritmanın performansı inverted generational distance metriğine göre değerlendirilmiştir (Çizelge 4.23). Bu değerlendirmeye göre, hypervolume metriğine göre olan değerlendirmede olduğu gibi hibrit NSGAII-PESA2 algoritması standart sapma değeri bakımından en düşük değeri almıştır ve bundan dolayı bu algoritma ile elde edilen verilerin daha düzenli bir dağılım gösterdiği söylenebilmektedir. Standart sapma ile ortalama arasındaki farklara bakıldığında ise hibrit NSGAII-SPEA2 algoritması ile elde edilen çözümlerin ortalamaya daha yakın bir dağılım gösterdiği gözlemlenmiştir.

Çizelge 4.23. Inverted generational distance metriğine göre istatistiksel sonuçlar (üçüncü konfigürasyon için)

	NSGAII-SPEA2	***NSGAII-PESA2***	***NSGAII-MOEA/D***
Min	0,101	0,197	0,239
Max	0,559	0,458	1,032
Median	0,161	0,280	0,399
Mean	0,084	0,285	0,459
Standart Deviation	0,188	0,055	0,200

Çizelge 4.24‘de ise üç hibrit algoritmanın çalışma zamanları açısından karşılaştırılması yapılmıştır. Elde edilen sonuçlar değerlendirildiğinde hibrit NSGAII-MOEAD algoritmasının diğer iki algoritmaya göre daha hızlı sonuç verdiği görülmektedir.

Çizelge 4.24. Çalışma zamanına göre istatistiksel sonuçlar (üçüncü konfigürasyon için)

	NSGAII-SPEA2(s)	***NSGAII-PESA2(s)***	***NSGAII-MOEA/D(s)***
Min	15,232	14,655	13,489
Max	27,155	22,425	21,319
Median	18,415	17,914	17,514
Mean	18,786	18,095	17,758
Standart Deviation	2,923	2,176	1,764

5. SONUÇ

Bu çalışmada robot tutucuların çok amaçlı optimizasyonu için çok amaçlı hibrit bir yöntem önerilmiştir. Önerilen hibrit yöntem içerisinde üç adet hibrit algoritma yer almaktadır. Bu algoritmalar; hibrit NSGAII-SPEA2, hibrit NSGAII-PESA2 ve hibrit NSGAII-MOEA/D algoritmalarıdır. Çalışmanın amacı önerilen hibrit optimizasyon yöntemi ile robot tutucu uçları arasındaki belli bir mesafe için tutucu tarafından uygulanan kuvvetlerin farkını, tutucu işleticisinin uyguladığı kuvvet ile tutucu uçlarının uyguladığı kuvvetin oranını, tutucu işleticisi ile tutucu uçları arasındaki kaydırma iletim oranını ve tutucuların tüm elemanlarının uzunluklarını minumuma indirmektir. Bu amaç doğrultusunda, üç farklı robot tutucu konfigürasyonu üzerinde önerilen hibrit optimizasyon yöntemi uygulanmıştır. Elde edilen sonuçlar önceki çalışmalarda probleme uygulanan farklı yöntemlerin sonuçlarıyla ve bazı çok amaçlı optimizasyon algoritmalarının problem üzerinde hibrit yöntemlerle eşit parametre değerleriyle çalıştırılmasıyla elde edilen sonuçlarla karşılaştırılmıştır. Optimizasyon sonuçları değerlendirildiğinde, önerilen çok amaçlı hibrit yöntem rakiplerine bariz bir üstünlük sağlamaktadır.

Bu tez çalışmasında sunulan robot tutucu tasarım probleminin çözümünde ilk kez hibrit çok amaçlı optimizasyon yöntemi uygulanmıştır. Bundan dolayı bu tez çalışması kapsamında geliştirilen bu yöntem, gelecekte robot tutucu tasarımı ile ilgili optimizasyon problemlerinin çözümünde iyi bir çözüm yöntemi olarak düşünülebilir.

KAYNAKLAR

1. Kamran, I. (2013). *Fundamental Engineering Optimization Methods Second Edition.*

2. Osyczka, A. (2002). Evolutionary algorithms for single and multicriteria design optimization. *Heidelberg: Physica-Verlag*.

3. Osyczka, A. and Krenich, S. (2003, November). *New methods for multicriteria design optimization using evolutionary algorithms*, in: Workshop: Optimal Design Laboratories de Me´ canique des Solides Ecole Polytechnique Palaiseau, France, 26–28.

4. Saravanan, R., Ramabalan, S., Godwin Raja Ebenezer, N. and Dharmaraja, C. (2009). Evolutionary Multi Criteria Design Optimization of Robot Grippers. *Applied Soft Computing,* 9(1) , 159-72.

5. Datta, R., and Deb, K. (2011). Multi-Objective Design and Analysis of Robot Gripper Configurations Using an Evolutionary-Classical Approach. *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation -GECCO '11*, 1843. Dublin, Ireland: ACM Press.

6. Monkman, G. J., Hesse, S., Steinmann, R. and Schunk, H. (2006). *Robot Grippers*. Wiley.

7. Chenl, J., Deng, H., Chai, W., Xiong, J. and Xia, Z. (2018). Manipulation Task Simulation of a Soft Pneumatic Gripper Using ROS and Gazebo. *2018 IEEE International Conference on Real-time Computing and Robotics(RCAR),* 378-83.

8. Bock, T., Herbst, J., Balaguer, C. and Abderrahim, M. (2000). Design of a Gripping System for the Automated Assembly of Large Building Modules. Taipei, Taiwan.

9. Pham, D. T., and S. H. Yeo. (1998). A knowledge-based system for robot gripper selection: criteria for choosing grippers and surfaces for gripping. *International Journal of Machine Tools and Manufacture,* 28(4), 301-13.

10. İnternet: Magnetic gripper | Goudsmit Magnetics, URL: https://www.goudsmitmagnets.com/industrial-magnetic-systems/magnetic-handling/robot-end-of-arm-tooling/magnetic-grippers . Son Erişim Tarihi: 05 Ekim 2019.

11. Bolmsjö, G. (2006). *Industriell robotteknik,* Studentlitteratur AB, Lund.

12. İnternet: Piab Broadens Scope of Kenos KVG Vacuum Grippers-The Robot Report, URL: https://www.therobotreport.com/piab-kenos-kvg-vacuum-grippers/ Son Erişim Tarihi: 05 Ekim 2019

13. İnternet: Pneumatic Gripper | AGW-500-1 | Clamp | Parallel Jaws. American, URL: http://www.agi-automation.com/product/agw-500-1-sealed-pneumatic-gripper/ . Son Erişim Tarihi: 05 Ekim 2019

14. Fraś, J., Maciaś, M., Czubaczyński, F., Sałek, P. and Główka, J. (2017). Soft Flexible Gripper Design, Characterization and Application. *In: Szewczyk R., Kaliczyńska M. (eds) Recent Advances in Systems, Control and Information Technology. SCIT 2016. Advances in Intelligent Systems and Computing*, vol 543. Springer, Cham.

15. Cutkosky, M.R. (1989). On Grasp Choice, Grasp Models, and the Design of Hands for Manufacturing Tasks. *IEEE Trans. Robot. Automat*, 5(3), 269–279.

16. Zitar, R.A. (2005). Optimum gripper using ant colony intelligence. *Industrial Robot:An International Journal*, 32 (1), 17-23.

17. Lanni, C. and Ceccarelli, M. (2009). An Optimization Problem Algorithm for Kinematic Design of Mechanisms for Two-Finger Grippers . *The Open Mechanical Engineering Journal*, 3(1), 49-62.

18. Li,Q., Qin, Q., Zhang, S.W., and Deng, H. (2010). Optimal design for heavy forging robot grippers. *Appl. Mech. Mater*, 44–47, 743–747.

19. Rao, R.V., Savsani, V.J. and Vakhaira, D.P. (2011). Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer Aided Design*, 43,303-315.

20. Pahwa, T., Gupta, S., Bansal, V., Prasad, B. and Kumar, D. (2012). *Analysis & Design Optimization of laterally driven Poly-Silicon Electro-thermal Micro-gripper for Micro-objects Manipulation*. 2012 COMSOL Conference, Bangalore.

21. Dao, T.P. and Huang, S.C. (2013). An Optimal Study of a Gripper Compliant Mechanism Based on Fuzzy-Taguchi Method. *Applied Mechanics and Materials*, 418, 141-44.

22. Ciocarlie, M., Hicks, F.M., and Stanford, S. (2013). Kinetic and dimensional optimization for a tendon-driven gripper. In *IEEE International Conference on Robotics and Automation*, 2751-58.

23. Avder, A., Şahin, İ., and Dörterler, M. (2019, June). Multi-objective Design Optimization of the Robot Grippers with SPEA2, *International Journal of Intelligent Systems and Applications in Engineering*, 7(2),83-87.

24. Laumanns, M., Thiele, L., Deb, K. and Zitzler, E. (2002). Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary computation*, 10(3), 263-282.

25. Aravelli and Aparna. (2014). Multi-objective Design Optimization of Engineering Systems: Uncertainty Approach and Practical Applications. *Open Access Dissertations.* 1156.

26. Zitzler, E. (1999). *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. Doctoral Dissertation, Swiss Federal Institute of Technology, Zurich.

27. Deb, K. (2001). *Multi-Objective Optimization using Evolutionary Algorithms.* Wiley-Interscience Series in Systems and Optimization, 954, Chiche

28. Kaya S. and Fığlalı N. (2017). Çok Amaçlı Optimizasyon Problemlerinde Pareto Optimal Kullanımı. *Social Sciences Research Journal*, 5(2), ISSN:2147-5237.

29. Ergül, E. (2010). *Çok Amaçlı Genetik Algoritmalar: Temelleri ve Uygulamaları*, Doktora Tezi, Samsun Ondokuz Mayıs Üniversitesi, Fen Bilimleri Enstitüsü, Samsun, 58-59.

30. Fonseca, C.,M. and Fleming, P.J. (1993). Genetic algorithms for multiobjective optimization:Formulation, discussion and generalization. *In S.Forrest(Ed.),Proceedings of the Fifth International Conference on Genetic Algorithms*, San Mateo, California, 416-423.

31. Horn, J. and Nafpliotis, N. (1993). Multiobjective optimization using the niched pareto genetic algorithm. IlliGAL Report 93005, Illinois Genetic Algorithms Laboratory, University of Illinois, Urbana, Champaign.

32. Hajela, P. and Lin, C.Y. (1992). Genetic search strategies in multicriterion optimal design. *Structural Optimization*, 4, 99-107.

33. Schaffer, J. D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. In J. J. Grefenstette (Ed.), *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, Pittsburgh, PA, 93–100.

34. Srinivas, N. and Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation, 2*(3), 221-248.

35. Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T. (2002). A Fast and Elitist Multiobjective Genetic Algorithm:NSGA-II. *IEEE Trans. Evol. Comput*., 6 (2), 182-197.

36. Zitzler, E. and L. Thiele (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*.

37. Deb, K. (2004). Multi-Objective Optimization Using Evolutionary Algorithms. John-Wiley and Sons, 497 p., New York.

38. Srinivas, N., and Deb, K. (1995). Multiobjective optimization using nondominated sorting in genetic algorithms, *Evol. Comput.*, 2 (3), 221-248.

39. Zhang, Q., and Li, H .(2007). MOEA/D:A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation*, Cilt 11, 712-731.

40. Ho-huu, V., Hartjesi S.,Visser, H.G and Curran, R. (2017). An Efficient Application of the MOEA/D Algorithm for Designing Noise Abatement Departure Trajectories. *Aerospace*, 4, 54.

41. Zitzler, E., Laumanns, M. and Thiele, L. (2001). *SPEA2:Improving the Strength Pareto Evolutionary Algorithm*. TIK-Report 103.

42. Corne, D.W., Knowles, J.D. and Oates, M.J. (2001). The Pareto Envelope-based Algorithm for Multiobjective Optimization. *Lecture Notes In Computer Science*; Vol. 1917 Proceedings of the 6th International Conference on Parallel Problem Solving from Nature.

43. Sağ T. (2008). *Çok Kriterli Optimizasyon için Genetik Algoritma Yaklaşımları*, Yüksek Lisans Tezi, Selçuk Üniversitesi Bilgisayar Mühendisliği Anabilim Dalı, Konya, 51-53.

44. İnternet: Hadka, D. (2016), *MOEA Framework*. URL: https://usermanual.wiki/Pdf/BeginnersGuidePreview.697890623.pdf, Son Erişim Tarihi: 11.04.2019.

45. Hadka, D. (2016). *Beginner's Guide to the MOEA Framework*. 2.8 edition. CreateSpace Independent Publishing Platform.

46. Laumanns, M., Zitzler, E. and Thiele, L. (2000). A Unified Model for Multi-Objective Evolutionary Algorithms with Elitism. *In 2000 Congress on Evolutionary Computation*, 1, 46–53, Piscataway, New Jersey, IEEE Service Center.

47. Bradstreet, L. (2011). *The hypervolume indicator for multi-objective optimisation: calculation and use*. Department of Computer Science & Software Engineering for the degree of Doctor of Philosophy of The University of Western Australia.

48. While, L., Hingston, P., Barone, L. and Huband, S. (2006). A Faster Algorithm for Calculating Hypervolume. *IEEE Transactions on Evolutionary Computation*, 10(1), 29–38.

49. Naujoks, B., Beume, N. and Emmerich, M. (2005). Multi-objective Optimization Using S-metric Selection: Application to Three-dimensional Solution Spaces. *In 2005 IEEE Congress on Evolutionary Computation (CEC'2005),* 2, 1282–1289.

50. Ishibuchi, H., Masuda, H., Tanigaki, Y. and Nojima, Y. (2014). *Difficulties in Specifying Reference Points to Calculate the Inverted Generational Distance for Many-Objective Optimization Problems*. Department of Computer Science and Intelligent Systems Graduate School of Engineering, Osaka Prefecture University Sakai, Osaka 599-8531, Japan 2014 IEEE.

51. Mashwani, W.K, Salhi, A., Jan, M.A., Khanum, R.A. and Sulaiman, M. (2015). Impact Analysis of Crossovers in Multiobjective Evolutionary Algorithm, *Science International*, 27 (6), 4943 - 4956.

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, Adı	:	AVDER, Ayşenur
Uyruğu	:	T.C.
Doğum Tarihi ve Yeri	:	26/04/1993 İstanbul
Medeni Hali	:	Bekâr
Telefon	:	0 (505) 835 76 44
E-posta	:	aysenuravder@gmail.com

Eğitim	**Yer**	**Mezuniyet Yılı**
Yüksek lisans	Gazi Üniversitesi / Bilişim Sistemleri	Devam ediyor
Lisans	Selçuk Üniversitesi / Bilgisayar Mühendisliği	2016
Lise	Ankara Cumhuriyet Anadolu Lisesi	2011

İş Deneyimi

Yıl	**Yer**	**Görev**
2018-2018	Parantez Teknoloji	Bilgisayar Mühendisi

Yabancı Dil

İngilizce

Yayınlar

Avder, A., Şahin, İ., and Dörterler, M. (2019, June). Multi-objective Design Optimization of the Robot Grippers with SPEA2, *International Journal of Intelligent Systems and Applications in Engineering (IJISAE)*, 7(2), 83-87.

Hobiler

Kitap okumak, film ve tiyatro izlemek, müzik dinlemek, futbol.

GAZİLİ OLMAK AYRICALIKTIR...