ÇOK AMAÇLI KARESEL ATAMA PROBLEMLERİ İÇİN MATEMATİKSEL VE SEZGİSEL ÇÖZÜM YAKLAŞIMLARI

Merve ULUEL

YÜKSEK LİSANS TEZİ

Endüstri Mühendisliği Ana Bilim Dalı

Danışman: Yrd. Doç. Dr. Zehra KAMIŞLI ÖZTÜRK

Eskişehir

Anadolu Üniversitesi

Fen Bilimleri Enstitüsü

Haziran, 2016

Bu tez çalışması BAP komisyonunca kabul edilen 1601F041 no.lu proje kapsamında desteklenmiştir.

JÜRİ VE ENSTİTÜ ONAYI

Merve ULUEL'in , "Çok Amaçlı Karesel Atama Problemlerinde Matematiksel ve Sezgisel Çözüm Yaklaşımları" başlıklı tezi, 28.06.2016 tarihinde aşağıdaki jüri tarafından Anadolu Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliği'nin ilgili maddeleri uyarınca, Endüstri Mühendisliği Ana Bilim dalında Yüksek Lisans tezi olarak kabul edilmiştir.

	Unvanı - Adı Soyadı	İmza
Üye (Tez Danışmanı) :	Yrd. Doç. Dr. Zehra KAMIŞLI ÖZTÜRK	…….
Üye :	Yrd. Doç. Dr. Nergiz KASIMBEYLİ	…….....
Üye :	Yrd. Doç. Dr. Tuğba SARAÇ	…….....

…….

Enstitü Müdürü

ÖZET

ÇOK AMAÇLI KARESEL ATAMA PROBLEMLERİ İÇİN MATEMATİKSEL VE SEZGİSEL ÇÖZÜM YAKLAŞIMLARI

Merve ULUEL

Endüstri Mühendisliği Anabilim Dalı

Anadolu Üniversitesi, Fen Bilimleri Enstitüsü, Haziran 2016

Danışman: Yrd. Doç. Dr. Zehra KAMIŞLI ÖZTÜRK

Karesel atama problemleri (QAP) 1957 yılında Koopmans ve Beckman tarafından geliştirilmiştir. Amaç tesis ve lokasyonlar arasındaki atamaların toplam maliyetinin enküçüklenmesinin sağlanmasıdır. QAP'ın matematiksel modelindeki kısıtlar 0-1 tamsayı atama kısıtları iken, amaç fonksiyonu da karesel yapıya sahip olduğu için doğrusal olmayan bir fonksiyondur. Bu iki nedenle, karesel atama problemleri çok zor problem sınıfına (NP-Zor) girmektedir ve 20'den fazla lokasyona sahip problemler için en iyi çözümün bulunması neredeyse imkânsızdır.

2002 yılında ise gerçek hayat problemlerinin çok amaçlı yapıya sahip olması nedeniyle, Çok Amaçlı Karesel Atama Problemi (mQAP) Knowles ve Corne tarafından geliştirilmiştir. Bundan sonra 2005 yılı itibariyle farklı yazarlar tarafından bu konu ile ilgili farklı çözüm yaklaşımları sunan çalışmalar yapılmıştır. Bu çalışmalar sezgisel ve metasezgisel algoritmalar temeline dayanmaktadır.

mQAP da, QAP gibi NP-Zor problem sınıfındadır. Ayrıca çözümü için doğrusal olmayan birden fazla amacın ele alınması gerekmektedir. Bu çalışmada, henüz literatürde nispeten yeni bir konu olan mQAP için bilinen test problemleri ele alınmış ve özellikle doğrusal olmayan yapıdaki çok amaçlı problemlerin çözümünde etkinliği kanıtlanmış Konik Skalerleştirme yöntemi ve çok amaçlı evrimsel algoritmalarından biri olan NSGA-II algoritması ile mQAP problemlerine çözüm aranmıştır. Çalışma sonucunda ayrıca evrimsel algoritma ve skalerleştirme yöntemlerinin güçlü yönleri bir araya getirilerek melez bir algoritma geliştirilmiştir. Elde edilen sonuçlar literatürdeki diğer yöntemlerle elde edilen sonuçlarla karşılaştırılarak performansları incelenmiştir.

Anahtar Sözcükler: Çok amaçlı karesel atama problemleri, Konik skalerleştirme, NSGA-II, Çok amaçlı evrimsel melez algoritma

ABSTRACT

MATHEMATICAL AND HEURISTICS SOLUTION APPROACHES TO THE MULTIOBJECTIVE QUADRATIC ASSIGNMENT PROBLEMS

Merve ULUEL

Department of Industrial Engineering

Anadolu University, Graduate School of Science, June, 2016

Supervisor: Asst. Prof. Dr. Zehra KAMISLI OZTURK

Quadratic Assignment Problems (QAP) were firstly defined by Koopmans and Beckmann in 1957. For this problem, the objective is cost minimization of total assignment between locations and facilities. While the constraints of QAP are 0-1 assignment constraints, the objective function is nonlinear because of the quadratic structure. Based on these two reasons, QAP is classified as NP-Hard. And it is almost impossible to find optimal solutions for number of locations are greater than 20.

Because of the fact that the multiobjective structure of real life problems, Multiobjective Quadratic Assignment Problems (mQAP) are suggested by Knowles and Corne in 2002. After this, as regards 2005, different researchers have studied about this subject based on different solution approaches. These studies are about heuristics and metaheuristic algorithms.

mQAP is also a NP-Hard problem like QAP. In addition, more than one nonlinear objective function have to be considered. In this study, mQAP which is a relatively new subject in the literature will be studied. And we will handle test problems from the literature and seek solutions with conic scalarization method. Conic scalarization method's effectiveness is proved in solution of multiobjective problems with non-linear structure. We also seek solutions with one of the multiobjective evolutionary algorithm NSGA-II. At the end of this study, we propose a hybrid method with evolutionary algorithm and scalarization method. Obtained results were compared with the results from the other methods in the literature and performances were compared.

Keywords: Multiobjective quadratic assignment problem, Conic scalarization, NSGA-II, Multiobjective evolutionary hybrid algorithm

TEŞEKKÜR

Bugüne kadar her zaman yanımda olan sevgili aileme, yüksek lisans eğitimim boyunca akademik çalışmaları ve kişiliği ile kendime örnek aldığım tez danışmanım Sn. Yrd. Doç. Dr. Zehra KAMIŞLI ÖZTÜRK'e, eğitim hayatım boyunca bana bilgi ve tecrübelerini aktaran tüm hocalarıma teşekkür ederim.

Ayrıca tezimi Bilimsel Araştırma Projesi olarak destekleyen Anadolu Üniversitesi Proje Destek Birimi'ne, yüksek lisansıma destek veren işletme müdürüm Sn. Mehmet AKAR'a, işletme müdür yardımcım Sn. Şirin AKÇAM'a ve manevi desteğini her zaman yanımda hissettiğim sevgili arkadaşım Caner SAKA'ya teşekkürlerimi sunarım.

…./…./20….

ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ

Bu tezin bana ait, özgün bir çalışma olduğunu; çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı; bu çalışma kapsamında elde edilemeyen tüm veri ve bilgiler için kaynak gösterdiğimi ve bu kaynaklara kaynakçada yer verdiğimi; bu çalışmanın Anadolu Üniversitesi tarafından kullanılan "bilimsel intihal tespit programı"yla tarandığını ve hiçbir şekilde "intihal içermediğini" beyan ederim. Herhangi bir zamanda, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçlara razı olduğumu bildiririm.

……..

(İmza)

…..…..…..…...……

(Adı-Soyadı)

İÇİNDEKİLER

ÇİZELGELER DİZİNİ

ŞEKİLLER DİZİNİ

ŞEKİLLER DİZİNİ (devam)

ŞEKİLLER DİZİNİ (devam)

1. GİRİŞ

Giriş bölümünde tezin amacı ve tezin kapsamdan bahsedilmiştir.

1.1. Tezin Amacı

Çalışmanın amacı, literatürde son 10 yılda çalışılmaya başlanmış olan mQAP (çok amaçlı karesel atama problemleri) için pareto en iyi çözümlerin farklı yöntemler ile araştırılması ve test problemlerinin bu yöntemler ile çözülmesidir.

NP-Zor sınıfındaki doğrusal olmayan yapıya sahip bu çok amaçlı problemin literatürde çözümü genellikle sezgisel/metasezgisel yöntemler ile araştırılmıştır. Tez kapsamında kullanılan Konik Skalerleştirme yöntemi hiçbir şekilde dışbükeylik ve kümelerin sınırlı olması ile ilgili koşulları talep etmemektedir. Bu nedenle çalışma kapsamında araştırılan bu yaklaşımın literatürdeki mevcut diğer yaklaşımlara göre avantajları araştırılarak önemi ortaya konulmaktadır.

Ayrıca, çözümü araştırılacak test problemleri için boyut büyüdükçe makul sürelerde ve olabildiğince çok pareto çözüm bulmak zorlaşacağından çok amaçlı evrimsel algoritmalardan NSGA-II metasezgiseli ile de pareto çözümler bulunmuştur. Son olarak, hem NSGA-II hem de Konik Skalerleştirme yönteminin güçlü yönlerini bir araya getirerek melez bir algoritma geliştirilmiş ve sonuçlar diğer yöntemlerle karşılaştırılmıştır.

1.2. Tezin Kapsamı

Tezin kapsamında mQAP'ı öneren Knowles ve Corne'un (2002) geliştirdiği test problemleri ele alınmıştır. Problem özellikleri şu şekildedir: 10 lokasyonlu ve 2 amaçlı, 20 lokasyonlu ve 2 amaçlı, son olarak 30 lokasyonlu ve 3 amaçlı problemler ele alınmıştır. Test problemleri sırasıyla ağırlıklandırılmış toplam skalerleştirme yöntemi, konik skalerleştirme yöntemi, NSGA-II metasezgiseli ve geliştirilen melez algoritma ile çözülmüş, sonuçlar karşılaştırılmıştır.

2. LİTERATÜR TARAMASI

Bu bölümde karesel atama problemleri (QAP), çok amaçlı karesel atama problemleri (mQAP) ve çözüm yaklaşımlarından SPEA2, NPGA, PAES, NSGA-II, ağırlıklandırılmış toplam skalerleştirme yöntemi ve konik skalerleştirme yöntemi için yapılan çalışmalar anlatılmaktadır.

2.1. Karesel Atama Problemleri

Karesel Atama Problemleri (QAP) NP-Zor problem sınıfına giren en zor problemlerden biridir. Karesel atama problemlerinde lokasyon ve tesisler bulunmaktadır. Problem her bir tesisin bir lokasyona ve her lokasyonun bir tesise atanması üzerine kurulmaktadır. Lokasyonlar arasında mesafe matrisleri ve tesisler arasında akış matrisleri bulunmaktadır. Amaç maliyetin enküçüklenmesidir.

QAP ilk olarak 1957 yılında Koopmans ve Beckmann tarafından önerilmiştir. Geliştirilen matematiksel model ekonomik aktiviteler ile ilgilidir. Daha sonra 1961 yılında Steinberg backboard kablolamada bileşenler arasındaki bağlantı sayısını en küçüklemek için problemi QAP olarak modellemiştir. 1972 ve 1980 yıllarında QAP konusunda çalışmalar yapan Heffley, QAP'ı ekonomik problemlere uygulamıştır. 1974 yılında Francis ve White servis müşterilerinde yeni bir tesise atama için bir karar verme sistemi geliştirmişlerdir. Bunların yanı sıra tesis yerleşim problemleri QAP için en popüler çalışmalardandır. 1972 yılında Dickey ve Hopkins, üniversite kampüsünde binaların yerleşimi için QAP uygulamışlardır. Daha sonra hastane ve park yerleşimi gibi tesis yerleşim konularında birçok çalışmalar yapılmıştır.

Problemde karar değişkeni x_{ij} olmak üzere iki atama kısıtı bulunmaktadır. Birincisi her tesisin bir lokasyona atanması, ikincisi de her lokasyonun bir tesise atanmasıdır. Amaç lokasyon ve tesislere göre akışın ve uzaklığın enküçüklenmesidir.

Klasik bir karesel atama probleminin modeli aşağıdaki gibidir:

$$x_{ij} = \begin{cases} 1, & i \text{ tesisi } j \text{ lokasyonuna atanırsa} \\ 0, & \text{diğer durumda} \end{cases}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad 1 \leq j \leq n, \tag{2.1}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad 1 \le i \le n, \tag{2.2}$$

$$x_{ij} \in \{0,1\} \qquad 1 \le i,j \le n \tag{2.3}$$

kısıtları altında,

$$enk \sum_{i,j=1}^{n} \sum_{k,p=1}^{n} b_{ij}\, a_{kp} x_{ik} x_{jp} \tag{2.4}$$

Modelde b_{ij} *i. ve j.* tesisler arasındaki akışı, a_{kp} *k.* ve *p.* lokasyonlar arasındaki uzaklığı göstermektedir.

2.2. Çok Amaçlı Karesel Atama Problemleri

Gerçek hayat problemlerinde birden fazla amaç olan durumlarla sık sık karşılaşıldığından QAP bu tür problemleri çözmek için yetersiz kalmıştır ve bunun sonucunda 2002 yılında Knowles ve Corne tarafından çok amaçlı karesel atama problemi (mQAP) geliştirilmiştir. Bu çalışmada mQAP matematiksel olarak ilk defa formüle edilmiş, çözüm için yerel arama temelli sezgisel bir algoritma geliştirilmiştir.

mQAP Knowles ve Corne tarafından aşağıdaki gibi formüle edilmiştir.

$$C^k(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}\, b^k_{\pi_i \pi_j} \quad , k \in 1,\ldots,m \tag{2.5}$$

$$enk\ C(\pi) = \{ C^1(\pi), C^2(\pi), \ldots. C^m(\pi) \} \tag{2.6}$$

Modelde n tesis/lokasyon sayısı, a_{ij} i ve j lokasyonları arasındaki mesafe, b^k_{ij} i tesisinden j tesisine olan k. akıştır. π_i, $\pi \in P(n)$ permutasyonunda *i.* tesisin lokasyonunu verir. Son olarak enküçüklemenin karşılığı Pareto yüzeyi elde etmektir. mQAP'ta akış sayısı aynı zamanda amaç sayısını vermektedir. Knowles ve Corne tarafından yapılan bu çalışmada elde edilen Pareto çözümlerin çeşitliliğini belirlemek amacıyla 1999 yılında Bachelet tarafından geliştirilen diameter ve entropy metrikleri kullanılmıştır.

2003 yılında yine Knowles ve Corne tarafından yapılan çalışmada ise amaç sayısının 2 ve 3 olduğu test problemleri ele alınmıştır. Yapılan çalışmada mQAP için test problemleri geliştiren bir program yazılmıştır. Problemleri *real-like* ve *uniform* olarak ikiye ayırmışlardır. Problemler lokasyon ve amaç sayısına göre kodlanmıştır. Örneğin, KC10-2fl-2rl olarak kodlanan problemde 10 lokasyon ve 2 amaç vardır ve rl, real like ifadesi için kullanılmaktadır.

Acan ve Unveren (2005) yaptıkları çalışmada keşif yetenekleri zayıflamadan gelecek vaat eden çözümlerin etrafında daha fazla yoğunlaşılmasını sağlayan çok amaçlı bir evrimsel algoritma geliştirmişlerdir. Acan ve Ünveren algoritmalarında Knowles ve

Corne (2003) tarafından geliştirilen test problemlerini kullanmışlardır ve yeni bir algoritma önermişlerdir. Önerilen segment tabanlı harici bellek stratejisine göre elde edilen sonuçlar, algoritmanın performansının literatürde yer alan diğer algoritmalarla rekabet edebilir düzeyde olduğunu göstermektedir.

Day ve Lamonet (2005) çalışmalarında bir başka çok amaçlı evrimsel algoritma olan MOMGA-II ve MOMGA-IIa'yı karşılaştırmışlardır. 3 farklı boyuta sahip olan (10, 20 ve 30) ve 2 ve 3 amaçlı 23 farklı test problemi çözülmüştür. Day ve Lamonet de Acan ve Ünveren (2005) gibi Knowles ve Corne tarafından geliştirilen test problemlerini ele almıştır. Çalışma sonucunda MOMGA-IIa'nın problem örnek boyutlarının 20'den küçük olduğu tüm pareto çözüm noktalarının elde edildiğini göstermişlerdir.

Ibáñez vd. tarafından 2005 yılında yapılan çalışmada karınca kolonisi algoritmasının ve çok amaçlı kombinatoryal eniyileme problemlerinin çözümü için evrimsel bir algoritma (SPEA2)'nın iteratif bir gelişme algoritması ve Gürbüz Tabu Algoritması ile melezlenmiş türevleri sunulmuştur. Çalışmada iki amaçlı problemler ele alınmıştır. Rassal yerel arama algoritmaları araştırılmış ve farklı skalerleştirme modelleri için tekrarlayan uygulamalarla karşılaştırılmıştır.

Garrett ve Dasgupta (2006) çok amaçlı bir bölge içinde yaygın olarak kullanılan bazı statik ve dinamik analiz tekniklerini genişletmiş ve yaygın çalışılmış çok amaçlı karesel atama problemi yapısı için analiz etmişlerdir. Statik landscape özellikleri kullanışlı bir anlayış için büyük bir anlaşma sağlamaktadır. Ayrıca çalışmada basit ardışık yerel arama algoritmasına karşı elde edilen melez çok amaçlı evrimsel algoritmaların güçlü yönleri gösterilmektedir.

2006 yılında Paquete ve Stützle yaptıkları çalışmalarında akış matrisi arasındaki bağlantıların farklı dereceleri ile iki amaçlı karesel atama problemi için iki rassal yerel arama algoritmasının performansına yer vermişlerdir. Deneysel sonuçlar göstermektedir ki, çözüm kalitesi ve hesaplama süresine göre algoritmanın performansı kuvvetle akış matrisleri arasındaki bağıntıya bağlıdır.

Gutiérrez ve Brizuela (2007) karesel atama problemi üzerinde Winners algoritması ile Go'nun çok amaçlı bir sürümünü incelemişlerdir. Tasarım parametresi olarak rassal yürüyüş uzunluğu ve parçacıkların sayısı analiz edilmiştir. Sonuç olarak parçacık başına ortalama rassal yürüyüş çabası ile algoritma tarafından oluşturulan ortalama çözüm kalitesi arasında bariz bir ilişki bulmuşlardır.

Ammar (2008) yaptığı çalışmada bulanık değişkenler olan amaçlarda, kısıtlarda ve karar vektöründe bulanık rassal katsayılar matrisi olan çok amaçlı bir karesel atama

problemini ele almıştır. Göreceli skaler bulanık karesel programlamanın çok amaçlı bulanık karesel programlamanın etkin çözümleri gösterilmiştir. Çalışmada bulanık portföy problemi üzerinde durulmuş ve çözüm önerilmiştir. Daha sonra önerilen çözümleri desteklemek için sayısal örneklere yer verilmiştir.

Zhao vd. (2008) çalışmalarında çok amaçlı karesel atama problemlerini çözmek için bulanık parçacık sürü algoritmasını önermişlerdir. Sürüdeki parçacıklar ve problem uzayı arasında etkin bir şekilde yeni bir eşleme önerilmiştir. Önerilen yaklaşımın performansı değerlendirilmiştir. Deneysel sonuçlar önerilen yaklaşımın mQAP'ın etkin bir şekilde çözülmesi için uygulanabilir olduğunu göstermektedir.

Ammar (2009) yaptığı çalışmada amaçta bulanık rassal katsayılar matrisi olan ve karar vektörü bulanık olan çok amaçlı karesel programlama problemi için yalancı değişkenler önermiştir. Bulanık karesel çok amaçlı programlama problemlerinin etkin çözümleri, göreceli skaler bulanık karesel programlamanın en iyi çözüm serilerinde yeniden çözülmüştür.

Garrett ve Dasgupta (2009) mQAP için çok amaçlı memetik algoritmaların yapımı için stratejilerin bir dizisinde deneysel bir karşılaştırma sağlamışlardır. Ayrıca farklı problem örneklerinin uyum landscape özelliklerinin analizinden kazanılan görüşleri kullanarak bu sonuçların daha ilkeli analizi sağlanmıştır.

Li ve Silva (2009) çalışmalarında mQAP'ta pareto-en iyi yüzeye yaklaşmak için mGRASP/MH olarak adlandırılan elitist Açgözlü Rassal Uyarlanabilir Arama İşlemi (GRASP) metasezgisel algoritmayı önermişlerdir. Yapılan deneysel çalışmalar mQAP örneklerinin karşılaştırmalı olarak çözüldüğünde mGRASP/MH'nin diğer son gelişmeleri yansıtan birçok çok amaçlı sezgisel algoritmalarla benzer ya da daha iyi olduğunu göstermiştir.

Özkale ve Fığlalı (2013) yaptıkları çalışmada çok amaçlı problemleri çözmek için karınca koloni eniyileme metasezgiseline dayalı bazı algoritmaları anlatılmış ve hem iki amaçlı karesel atama problemi (biQAP) örneklerini çözmek için hem de bu algoritmaların performansını değerlendirmek için programlanmıştır. Gürbüz parametre kümeleri hesaplanmış ve literatürdeki iki amaçlı problemler, bu parametreler kullanılarak çözülmüştür. Değerlendirme adımında ise hiyerarşik olmayan yapıda çoklu önem testi kullanılmış ve bu test için gerekli olan performans metriği tanıtılmıştır.

2014 yılında Almeida vd. tarafından yapılan çalışmada iki amaç bulunmaktadır. İlk amaç, mQAP'la başa çıkan transgenetik algoritmaların ve çok amaçlı evrimsel eniyileme sistemini birleştiren melez algoritmaların incelenmesi ve ikinci olarak da mQAP ile başa

çıkmak için ayrışmaya dayalı olanlarla Pareto hakimiyetine dayalı çok amaçlı evrimsel eniyileme algoritmalarının yeteneğini karşılaştırmaktır. Bu kapsamda NSTA (TA + NSGA-II) ve MOTA/D (TA + MOEA/D) olmak üzere iki melez algoritma önerilmiştir. Önerilen algoritmalar 126 test probleminde NSGA-II ve MOEA/D ile karşılaştırılmıştır. Sonuçlar MOTA/D'nin üstünlüğünü göstermektedir.

Drugan tarafından 2014 yılında yapılan çalışmada bilinen yerel en iyi ile küçük boyutlu mQAP birleştirilerek geniş mQAP durumları üretilmiştir ve bileşik mQAP olarak adlandırılmıştır. Çalışmanın sonucunda sayısal deneyler kullanılarak çok amaçlı metasezgiseller için bileşik mQAP'ın zor olduğu gösterilmiştir.

2014 yılında Liefooghe, Verel ve Hao tarafından yapılan çalışmada elitist evrimsel çok amaçlı eniyileme algoritması ve bir kazanım skalerleştirme fonksiyonu kullanarak state-of-the-art tek amaçlı tabu arama işlemini birleştiren bir melez sezgisel önerilmiştir.

Literatür taramasında mQAP için farklı çözüm yöntemlerinin geliştirildiği, melez algoritmaların da üzerinde çalışıldığı bir konu olduğu anlaşılmıştır. Ele alınan farklı problemler hem sezgisel hem de matematiksel yöntemlerle çözülmüştür. Gerçek hayat problemlerine de uygulanabilir olduğundan tercih edilmektedir.

2.3. Çok Amaçlı Problemlerde Çözüm Yaklaşımları

Bu bölümde literatürde yer alan ve çok amaçlı problemleri çözmek için geliştirilen çok amaçlı evrimsel algoritmalar hakkında bilgi verilecektir.

2.3.1. SPEA2 (Güçlü Pareto Evrimsel Algoritma-2)

SPEA2, 2002 yılında Zitzler ve diğerleri tarafından geliştirilmiştir. 1999 yılında Zitzler ve Thiele tarafından önerilen SPEA'nın diğer çok amaçlı evrimsel algoritmalarla karşılaştırıldığında çok iyi sonuçlar vermesi üzerine daha da geliştirilerek SPEA2 algoritması önerilmiştir. Yapılan çalışmada ayrıca SPEA2 algoritması SPEA, PESA ve NSGA-II ile karşılaştırılmıştır.

SPEA olağan popülasyon ve kayıtları (dış küme) kullanır. Başlangıç bir çözümle başlar ve boş kayıtlar her bir iterasyon ile doldurulur. İlk olarak tüm baskın olmayan popülasyon üyeleri kayıtlara kopyalanır. Herhangi bir baskın olmayan birey veya çiftler bu güncelleme işlemi sırasında kayıtlardan kaldırılır. Güncellenen kayıtların boyutu önceden belirlenmiş limitleri aşarsa, bundan başka kayıt üyeleri baskın olmayan

özellikleri koruyan bir kümeleme tekniği ile silinir. Bundan sonra da uyum değerleri hem kayıtlara hem de popülasyon üyelerine atanır.

SPEA2 algoritması adı geçen problemlerin üstesinden gelmek için önerilmiştir. Algoritması şu şekildedir:

Girdiler:

N (popülasyon büyüklüğü)

$\overline{N}$ (kayıtların büyüklüğü)

T (jenerasyonun en büyük sayısı)

Çıktı:

A (baskın olmayan küme)

Adım 1: Başlangıç durumu: Başlangıç popülasyonu P_0 türetilir ve boş bir kayıt oluşturulur. $\bar{P}_0 \neq 0$. t=0 olarak set edilir.

Adım 2: Uyum Ataması: P_t ve $\bar{P}_t$'deki bireylerin uyum değerleri hesaplanır.

Adım 3: Çevre seçimi: P_t ve $\bar{P}_t$'deki tüm baskın olmayan bireyler $\bar{P}_{t+1}$'e kopyalanır. $\bar{P}_{t+1}$ boyutu $\overline{N}$'yi aşarsa, kesme operatörü vasıtasıyla $\bar{P}_{t+1}$ azaltılır. Diğer durumda $\bar{P}_{t+1}$, N'den daha küçükse, o zaman $\bar{P}_{t+1}$, P_t ve $\bar{P}_t$'deki baskın bireylerle doldurulur.

Adım 4: Bitiş: $t \geq T$ veya diğer durdurma kriterleri sağlanırsa, $\bar{P}_{t+1}$'de baskın olmayan bireyler tarafından sunulan karar vektörleri kümesine A koyulur.

Adım 5: Çaprazlama Seçimi: Çaprazlama havuzunu doldurmak için $\bar{P}_{t+1}$ üzerinde değiştirme ile ikili turnuva seçimi gerçekleştirilir.

Adım 6: Varyasyon: Çaprazlama havuzuna yeniden kombinasyon ve mutasyon operatörleri uygulanır ve elde edilen popülasyona $\bar{P}_{t+1}$ konulur.

Çalışma sonucunda farklı problemlerde SPEA2'nin NSGA-II ile benzer olduğu görülmüştür. Bazı durumlarda özellikle amaç sayısı arttığından SPEA2 noktaların daha iyi dağılımı sağlarken, NSGA-II geniş yayılıma ve dolayısıyla performans ölçümünde daha iyi bir değere ulaşmıştır. PESA ise ancak bazı test fonksiyonlarında dış çözümleri tutmak için bazı zorluklara sahip olma eğilimindedir. Zitzler ve diğerleri (2002) tarafından yapılan çalışmada üç algoritmanın karşılaştırmasından elde edilen sonuçlar şu şekildedir:

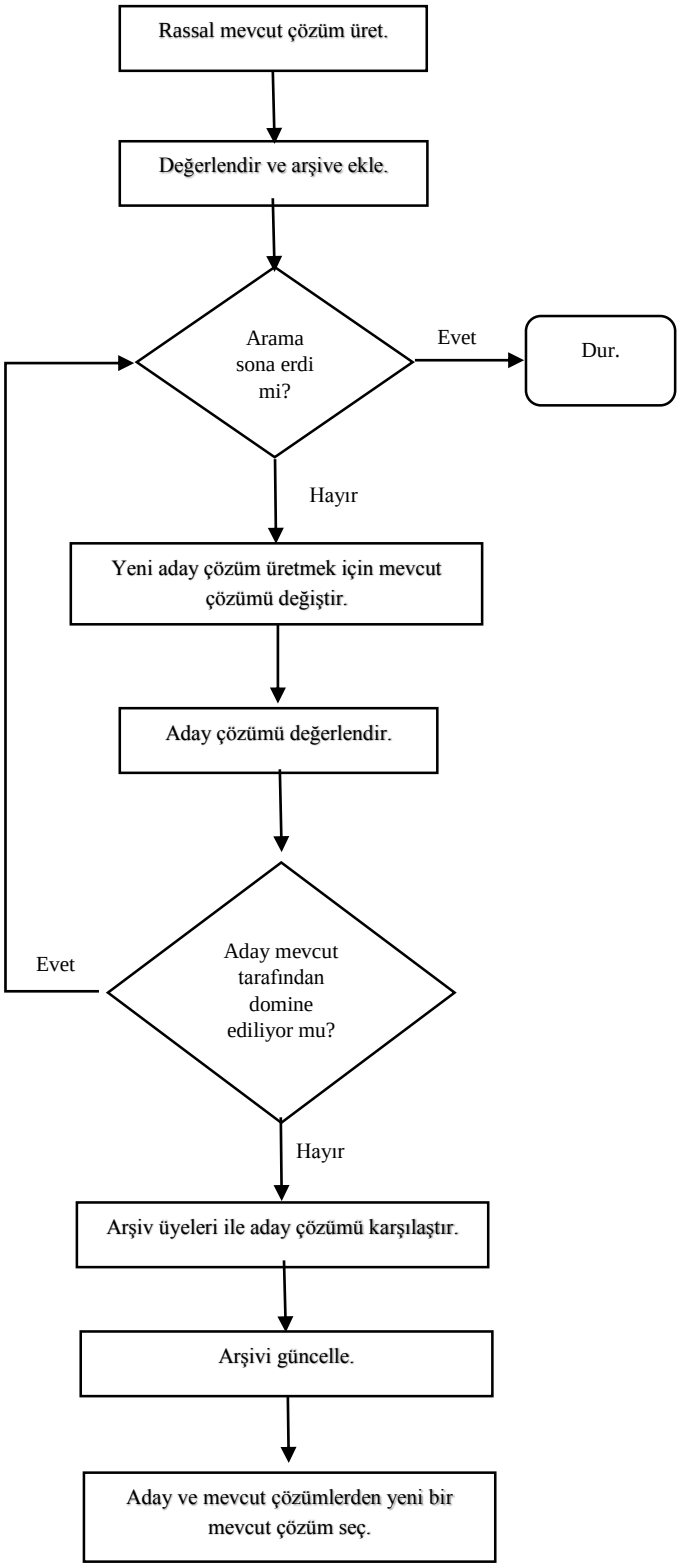
- SPEA2 tüm problemlerde öncül SPEA'dan daha iyi bir performans göstermiştir.

- PESA muhtemelen daha yüksek bir seçkincilik yoğunluğuna göre en hızlı yakınsamaya sahiptir. Ama her zaman sınır çözümler tutamadığı için bazı problemlerde zorlukları vardır.
- SPEA2 ve NSGA-II tüm problemlerde en iyi performansı göstermiştir.
- Daha yüksek boyutlu amaç uzayında SPEA2'nin PESA ve NSGA-II üzerinde avantajlarının olduğu görülmüştür.

2.3.2. PAES (Pareto Arşivlenen Evrim Stratejisi)

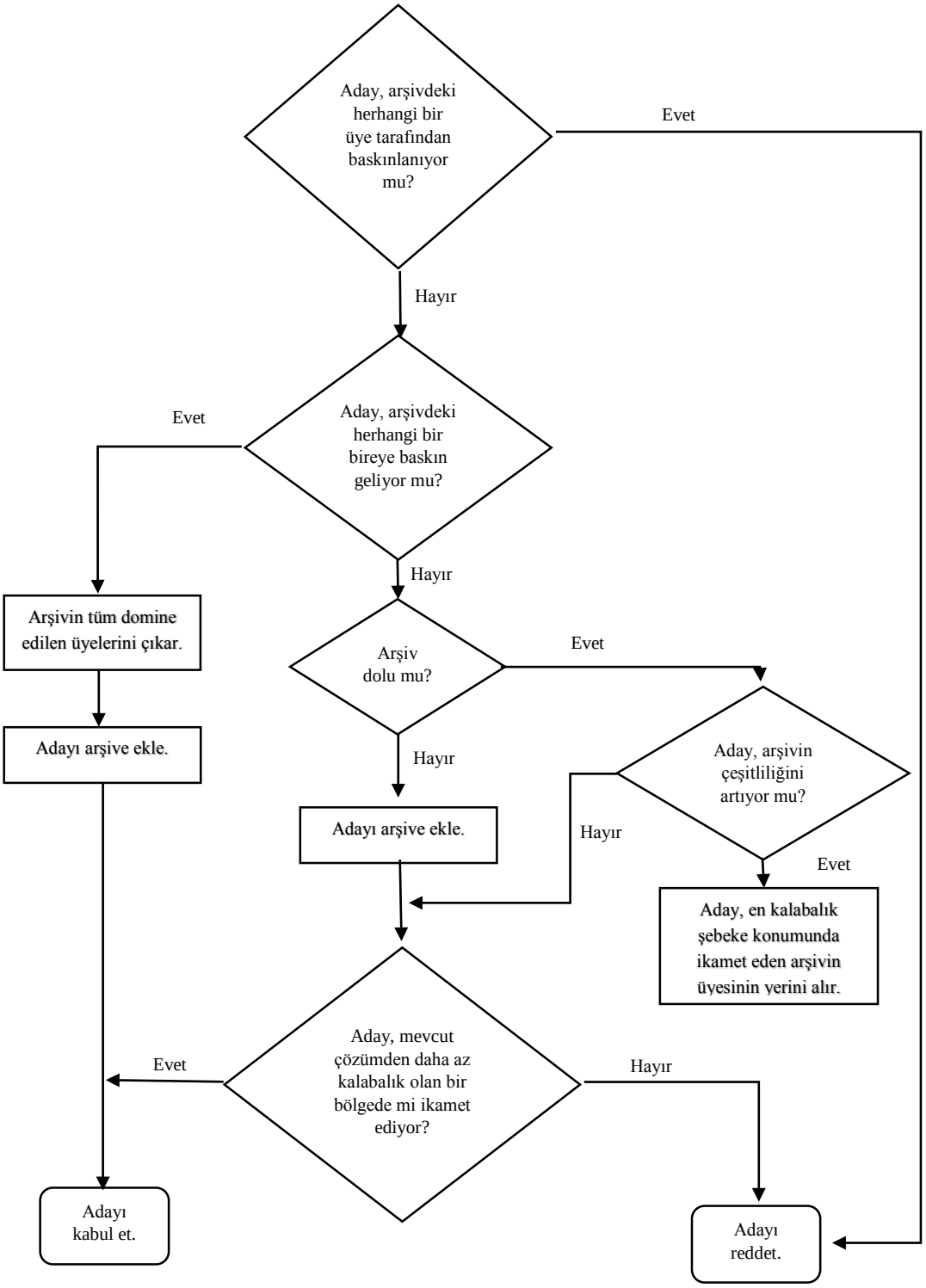
1999 yılında Knowles ve Corne tarafından pareto çok amaçlı eniyileme için yeni bir algoritma olan PAES geliştirilmiştir. PAES'in geliştirilmesinde iki ana amaç vardır. Bunlardan ilki algoritmanın kesinlikle yerel arama ile sınırlandırılmasıdır. Örneğin, algoritma sadece küçük değişimlere uğrayan operatörleri kullanmalı ve mevcut çözümden civardaki komşuya geçmelidir. Bu yönüyle PAES, NPGA, NSGA ve VEGA başta olmak üzere birçok çok amaçlı genetik algoritmadan oldukça farklı olmaktadır. İkinci amaç ise, algoritmanın gerçek Pareto en iyisi olması ve eşit değer sahip olan ve birbirine baskın olmayan tüm çözümleri iyileştirmesidir. Ancak aynı zamanda bu iki amaca birden erişilmesi tartışmalıdır. Bu nedenle vakaların bir çoğunda, iki çözüm karşılaştırıldığında biri diğerine baskın olmayacaktır. PAES'te bu problemin üstesinden daha önce bulunan baskın olmayan çözümlerin arşivlenmesinin sürdürülmesi ile gelinmektedir. Aslında, üç ayrımı kapsayacak şekilde PAES'in incelenmesi yol gösterici olacaktır. Bu üç ayrım şu şekilde incelenebilir: aday çözüm, aday kabul çözüm fonksiyonu ve baskın olmayan çözümler arşivi. Bu şekilde bakıldığında PAES bir çok amaçlı yerel arama prosedürüne en basit şekilde anlaşılması zor olan yaklaşımı sunmaktadır.

PAES'in yapısı Şekil 2.1'de gösterilmektedir.



Şekil 2.1. *PAES Prosedürü*
Kaynak: *Knowles, Corne, 1999*

PAES'in arşivleme ve kabul mantığı Şekil 2.2'de gösterilmektedir.

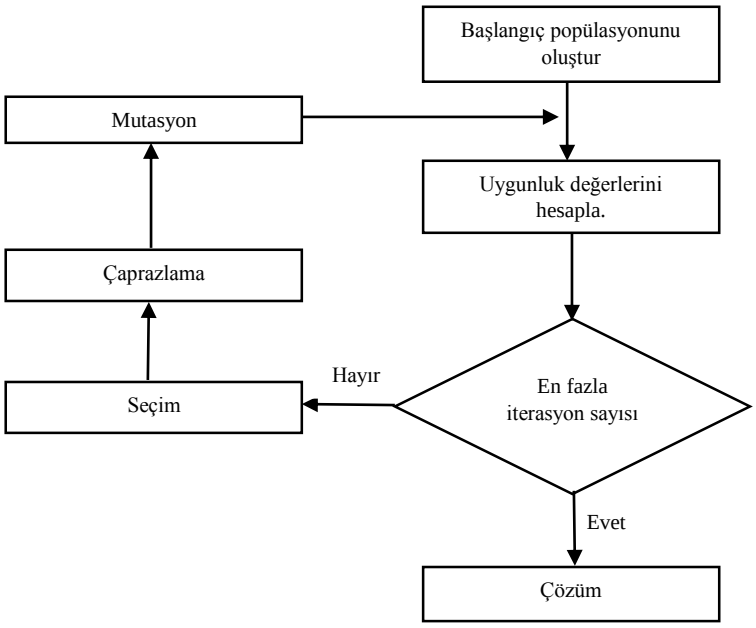


Şekil 2.2. *PAES Arşivleme ve Kabul Mantığı Yapısı*
Kaynak: *Knowles, Corne, 1999*

Çalışmada test problemleri ele alınmış ve PAES, içe yerleştirilmiş pareto evrimsel algoritma (NPEA) ile karşılaştırılmıştır. Sonuçta PAES'in daha iyi sonuçlar verdiği görülmektedir.

2.3.3. Genetik algoritmalar

Genetik algoritmaların ilkeleri ilk defa 1975 yılında Michigan Üniversitesi'nde John Holland tarafından geliştirilmiştir. Holland 1975 yılında yaptığı çalışmada evrim yasalarını genetik algoritmalar içinde optimizasyon problemleri için kullanmıştır. Genetik algoritmanın genel yapısı şu şekildedir:



Şekil 2.3. *Genetik algoritma yapısı*
Kaynak: *Holland, 1975*

Genetik algoritmanın yapısında geçen kavramların tanımları aşağıda gösterilmektedir:

- Seçim: Mevcuttaki bireyi genetik yapısında herhangi bir değişiklik yapmadan bir sonraki nesle kopyalar.
- Çaprazlama: İki bireyin yapılarını rassal olarak birleştirir ve yeni nesiller oluşturur.
- Mutasyon: Var olan bireyin genlerinden biri ya da birkaç tanesinin yerleri değiştirilerek oluşturulur.

Klasik genetik algoritmada yapılan bir takım değişikler sonucunda yeni algoritmalar geliştirilmiştir. Bunun sonucunda geliştirilen NPGA ve NSGA-II algoritması izleyen bölümlerde anlatılmıştır.

2.3.3.1. NPGA (İçe Yerleştirilmiş Pareto Genetik Algoritma)

Horn vd. (1994) tarafından pareto en iyi çözümleri bulmak için NPGA algoritması geliştirilmiştir. NPGA'nın özellikleri genetik algoritma için seçim uygulamasını yerelleştirmektedir. Genetik algoritmanın en yaygın uygulanan seçim tekniği turnuva seçimidir. Turnuva seçiminde bireyler mevcut popülasyondan rassal olarak seçilir ve en iyi alt küme bir sonraki nesle aktarılır. Turnuva seçiminde sadece bir tek cevap istendiği varsayılır. Nesillerin belli bir sayısından sonra popülasyon homojen olarak bir araya gelecektir. Çalışmada yakınsamayı önlemek ve çoklu Pareto optimum çözümleri korumak için turnuva seçimi iki yolla değiştirilmiştir. İlk olarak Pareto hakimiyeti turnuvası eklenmiştir. İkinci olarak da dominant olmayan bir turnuva olduğunda, kazananı belirlemek için paylaşım uygulanmaktadır.

Pareto hâkimiyeti turnuvasında ilk olarak, küçük yerel hakimiyet kriterleri kullanılmış ancak yetersiz hakimiyet baskısının üretildiği görülmüştür. Sonraki nesillerde çok fazla egemen birey vardır ve iki örnek büyüklüğü bireyin gerçek "hâkimiyet sıralamasını" tahmin etmek çok küçük bir ihtimal olarak görünmektedir. Seçim için populasyondan iki aday rassal olarak seçilir. Aynı zamanda popülasyondan rassal olarak bireylerin karşılaştırma kümeleri seçilir. Adayların her biri daha sonra karşılaştırma kümesindeki adaylarla karşılaştırılır. Bir aday karşılaştırma kümesinde baskın ve diğeri değilse üreme için ikincisi seçilir. Eğer hiçbiri baskın değil veya her ikisi de baskınsa o zaman kazananı seçmek için paylaşım kullanılmalıdır. t_{dom} örnek büyüklüğü (karşılaştırma kümesinin boyutu), hâkimiyet baskısı olarak adlandırılan seçim baskısı üzerinde kontrolü verir. Kazanan rassal olarak seçilirse, genetik sürüklenme popülasyonun Paretonun tek bir bölgesine yaklaşmasına neden olur.

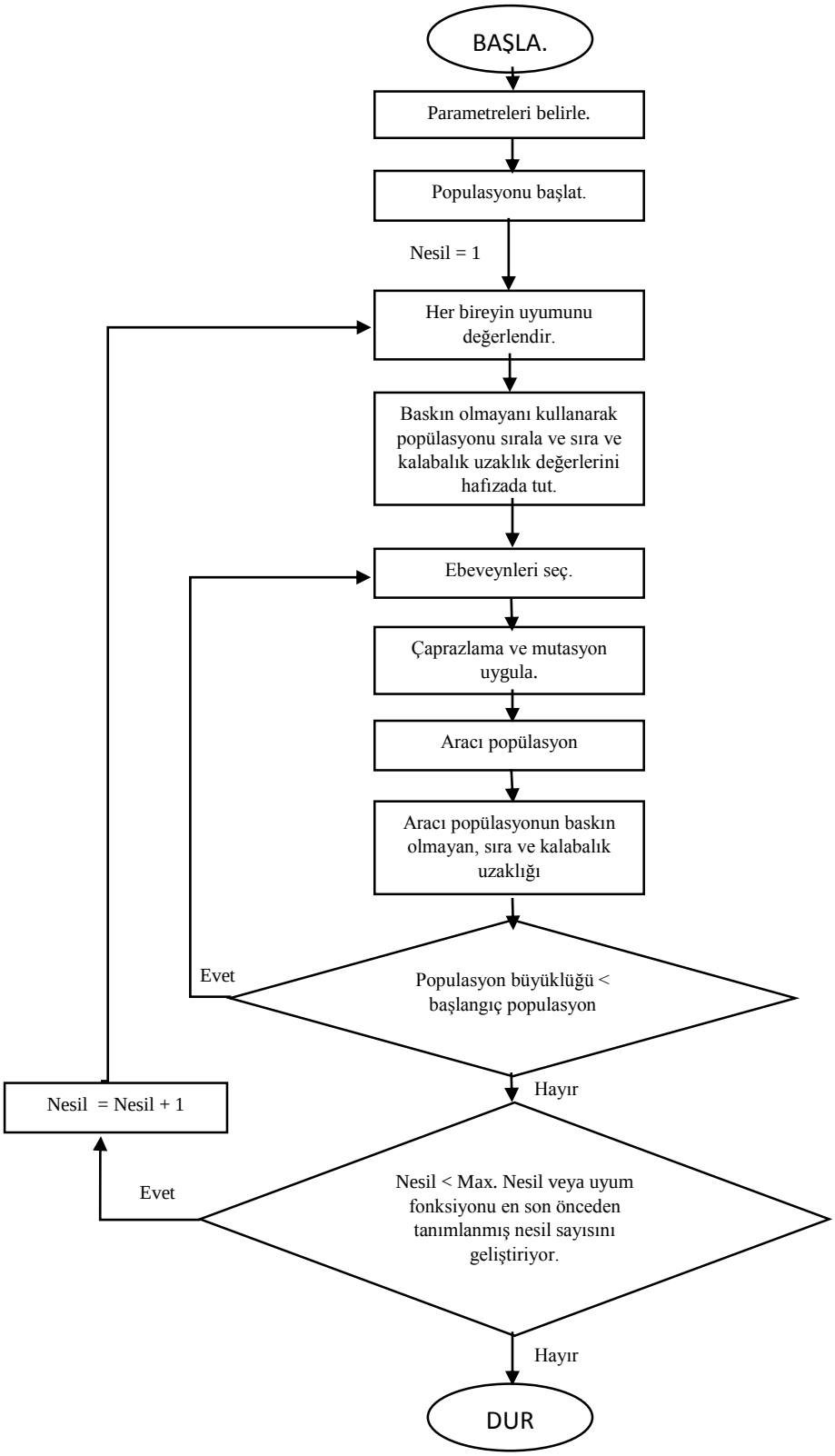
Hâkim olmayan sınır bölgesinde paylaşımda ise; uyum paylaşımının amacı, arama uzayındaki farklı zirveleri bir dizi üzerinden nüfusa dağıtmaktır.

Çalışmada iki problem çözülmüş ve sonuçları değerlendirilmiştir. Sonuç olarak niched, Pareto tekniğinin uygulanmasına ilişkin bulunan ön sonuçlar cesaretlendirici bulunmuştur. Çalışma sonucunda NPGA'nın performansının çeşitli parametrelerin ayarlarına duyarlı olduğu bulunmuştur. Özellikle, yeterince büyük bir nüfusun etkili bir şekilde araştırılması ve Pareto frontun genişliğine örnek olması önemlidir. Ancak NPGA'nın davranışı en çok uygulanan seçim baskısı derecesinden etkilenecek gibi görünmektedir. Seçim baskısı için sadece turnuva boyutu t_{size} ve turnuva seçimi ile klasik genetik algoritmada prematüre yakınsama kritiktir. Bu yüzden t_{dom} NPGA'nın

yakınsamasını direkt olarak etkilemektedir. Sonuç olarak, hâkimiyet turnuvasının sadece hâkimiyet ilişkisine dayalı olmadığı, aynı zamanda da antisimetrik ve geçişli bağlantıya dayandığı gösterilmektedir. Benzer şekilde, denklik sınıfı paylaşımı sadece Pareto çözüm sınırında değil, aynı zamanda herhangi bir denklik sınıfında da yararlıdır. Böylece, NPGA sadece çok amaçlı problemlerde Pareto yaklaşımı tarafından tetiklenen değil, aynı zamanda herhangi bir kısmi sıralı alanı aramak için kullanılabilir.

2.3.3.2. *NSGA-II (Baskın Olmayan Sıralama Genetik Algoritma-II)*

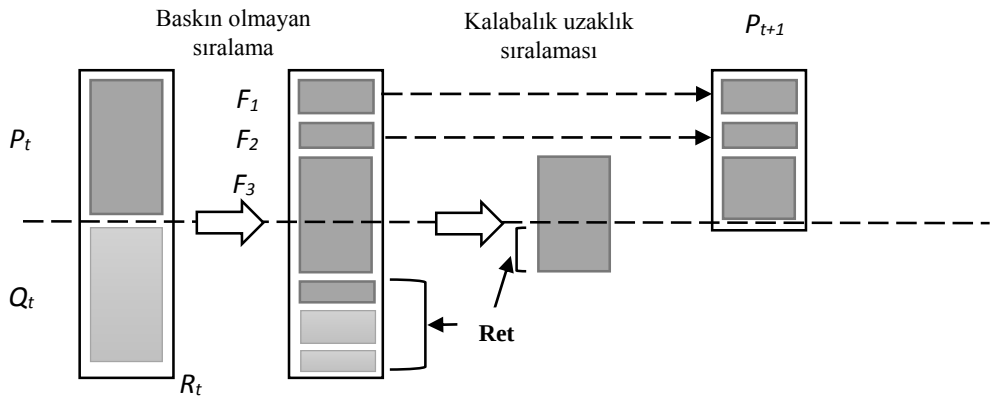
NSGA-II algoritması Deb (2002) tarafından önerilmiştir. Baskın olmayan sıralamayı ve paylaşımı kullanan çok amaçlı evrimsel algoritmalar üç ana faktörden dolayı eleştirilmektedir. Bunlardan birincisi, $O(MN^3)$ hesap karmaşıklığı (M amaç sayısı, N popülasyon büyüklüğüdür.); ikincisi, elitist olmayan yaklaşımı ve sonuncusu paylaşım parametresini belirleme ihtiyacıdır. Bu nedenle Deb ve diğerleri bu üç zorluğu ortadan kaldırmak için NSGA-II'yi geliştirmişlerdir. Algoritmanın yapısı Şekil 2.4'te gösterilmektedir.



Şekil 2.4. *NSGA-II Genel Yapısı*
Kaynak: *Coello vd., 2002*

NSGA-II‘de öncelikle ebeveyn popülasyonu rassal olarak oluşturulur. Popülasyon, Pareto üstünlüklerine dayalı olarak sıralanır. Her çözümün uygunluğu, kendi bastırılmamışlık seviyesine eşit olarak atanır. Sonra, bilinen ikili turnuva seçimi, çaprazlama ve mutasyon işlemcileri, N boyutlu popülasyonun bireylerini oluşturmak için ebeveyn popülasyona uygulanır. Seçkinlik işlemi ise, önceki en iyi bastırılmamış çözümlerle mevcut popülasyon karşılaştırılarak gerçekleştirilecektir. Bu yüzden, başlangıç jenerasyonundan sonra prosedür farklı bir şekilde işler. Klasik genetik algoritmadan farklı olarak kalabalık uzaklığı dikkate alarak popülasyonu oluşturur. Bireyin bir sonraki popülasyona geçmesi için kalabalık uzaklığının yüksek olması beklenir. Bu nedenle kalabalık uzaklığı düşük olanlar bir sonraki popülasyona geçemez ve elenirler.

NSGA’nın genel çalışma prosedürü Şekil 2.5’te gösterilmektedir:



Şekil 2.5. *NSGA-II Genel Çalışma Prosedürü*
Kaynak: *Deb, 2002*

2014 yılında Masri ve diğerleri tarafından yapılan çalışmada SPEA2, PAES ve NSGA-II karşılaştırılmış ve NSGA-II’nin üstünlüğü gösterilmiştir. 2015 yılında Ghoreishi ve arkadaşları tarafından yapılan çalışmada NSGA-II, ∈-NSGA-II, ∈-MOEA, PAES, PESAII ve SPEA2 karşılaştırılmış ve çalışma sonucunda NSGA-II, ∈-NSGA-II, ∈-MOEA algoritmalarının diğer yöntemlere kıyasla daha makul bir sürede daha iyi çözümler sunmuşlardır. Ayrıca 2002 yılında Zitzler ve diğerleri tarafından yapılan çalışmada SPEA2 önerilmiş, SPEA2 ve NSGA-II’nin PAES ve SPEA’ya göre tüm problemlerde daha iyi sonuçlar elde ettiği gösterilmiştir.

Yukarıda bahsedilen çalışmalarda NSGA-II’nin ön plana çıkması nedeniyle bu tezde NSGA-II algoritması ele alınmış ve Knowles ile Corne (2003) tarafından

geliştirilen test problemleri NSGA-II algoritması ile çözülmüştür. Çözümler, izleyen yöntem başlığı altında anlatılacaktır.

2.3.4. Ağırlıklandırılmış toplam skalerleştirme yöntemi

Ağırlıklandırılmış toplam skalerleştirme yöntemi (ATY) 1955 yılında Gass ve Saaty tarafından geliştirilmiş ve çok amaçlı problemlerin (ÇOP) çözümleri için en çok kullanılan skalerleştirme yöntemidir. Ağırlıklandırılmış toplam skalerleştirme yöntemi karar verici tarafından belirlenen ağırlık katsayıları ile her bir amacı ilişkilendirir ve amaçların ağırlıklandırılmış toplam fonksiyonunu en iyiler.

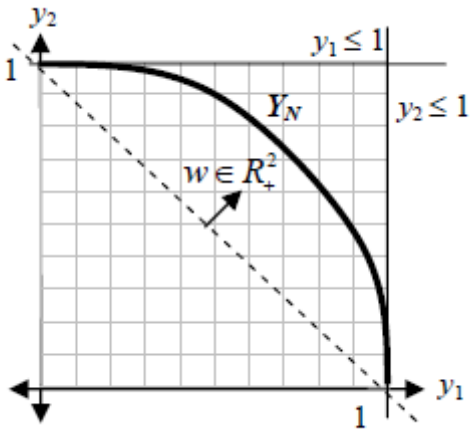
$w' = (w_1, \dots, w_n)$ ağırlık vektörü olsun ve her w_i değeri bir $f_i\,(x), i = 1, \dots, n$ amaç fonksiyonu ile ilişkilendirilir.

Verilen ağırlık vektörü için yazılmış skaler problem şu şekildedir:

$$\min_{x \in X} \sum_{i=1}^{n} w_i f_i(x) \qquad (WSS(w)) \tag{2.7}$$

(WSS(w)) skaler probleminin çözümler kümesi Sol*(WSS(w))* ile gösterilecektir.

(WSS(w)) probleminin iki boyutlu grafiksel gösterimi Şekil 2.6'da gösterilmektedir. Grafikte y_1, f_1'i ve y_2, f_2'yi temsil etmektedir.



Şekil 2.6. *Ağırlıklandırılmış Toplam Skalerleştirme Yöntemi İki Boyutlu Gösterim*
Kaynak: *Üstün, 2007*

2.3.5. Konik skalerleştirme yöntemi

Kasımbeyli tarafından geliştirilen konik skalerleştirme yönteminin (KSY) mantığı oldukça basittir: $w \in C^{\#}$ ağırlık vektörü, $r \in R^n$ referans noktasından oluşan tercih parametreleri seçilir ve l_1-normunun kullanıldığı $(w, \alpha) \in C^{a*}$(ya da $(w, \alpha) \in$

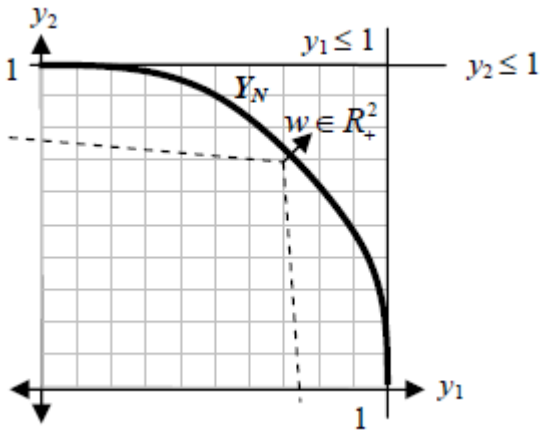
C^{a0}ya da $(w,\alpha) \in C^{a\#}$) şeklinde bir α genişletme parametresi belirlenir ve skaler problem çözülür.

$$\min_{x\in X} \sum_{i=1}^{n} w_i(f_i(x) - r_i) + \alpha \sum_{i=1}^{n} |f_i(x) - r_i| \qquad (CS(w.\alpha,a)) \qquad (2.8)$$

Skaler problem $(CS(w.\alpha,r))$'nin çözümler kümesi $Sol\big(CS(w,\alpha,r)\big)$ şeklinde gösterilmiştir. $r = (r_1,\dots,r_n)$ referans noktası, bu noktalara yakın olan etkin elementlerin bulunmasını istediği durumlarda karar verici tarafından belirlenebilir. Konik skalerleştirme yöntemi referans noktasının belirlenmesi için herhangi bir kısıtlama koymamaktadır. Bu noktalar keyfi olarak seçilebilir.

Açıkça görülmektedir ki $\alpha=0$ olduğu durumda (veya $f(X) \subseteq \{r\} \pm R_+^n$) konik skalerleştirme yönteminin amaç fonksiyonu ağırlıklandırılmış toplam skalerleştirme yönteminin amaç fonksiyonuna dönüşür. Uygun çözüm kümesi üzerinden böyle bir amaç fonksiyonunun enküçüklenmesi, sadece amaç uzayını destekleyen destek hiper düzlem üzerindeki noktaları elde edebilecektir. α arttıkça koni küçülecektir.

Konik skalerleştirmenin çözüm uzayı Şekil 2.7'de gösterilmektedir. Grafikte y_1, f_1'i ve y_2, f_2'yi temsil etmektedir.



Şekil 2.7. *Konik Skalerleştirme Yöntemi Grafik Gösterimi*
Kaynak: *Üstün, 2007*

Üstün (2007) tarafından yapılan çalışmada skalerleştirme yöntemleri arasında karşılaştırma yapılmıştır. Karşılaştırma tablosu Çizelge 2.1'de gösterilmektedir.

Çizelge 2.1. *Skalerleştirme Yöntemlerinin Karşılaştırılması*

	Ölçütler										
Yöntemler	Amaç fonksiyonu değerleriyle ilgili istek düzeyleri	Amaç fonksiyonlarının ağırlıkları	Amaç fonksiyonlarının değerleriyle ilgili düzey şartları	Amaç fonksiyonları ve uygun çözüm alanı üzerinde dışbükeylik şartı	Pareto etkin değer kümesi üzerinde kompaktlık şartı	Zayıf Pareto etkin değerler	Pareto Etkin değerler	Has Pareto etkin değerler	Ek kısıt sayısı	Ek değişken sayısı	Ek problem sayısı.
Ağırlıklandırılmış toplam skalerleştirme yöntemi		+		+	+	+	+	+	0	0	0
ε-kısıt yöntemi			+			+	+		p-1	0	p-1
Elastik kısıt yöntemi			+			+	+		p-1	p-1	p-1
Benson yöntemi	+					+	+		p	p	0
Uzlaşık programlam yöntemi	+				+	+	+	+	p	1	p
Konik skalerleştirme yöntemi	+	+				+	+	+	0	0	0
Hedef programlama	+	+							p	2p	0

Kaynak: *Üstün, 2007*

Çizelge 2.1'de görüldüğü gibi konik skalerleştirme yöntemi, birçok açıdan diğer skalerleştirme yöntemlerinden üstündür. Üstün (2007) yaptığı doktora tezinde ayrıca şu bilgilere yer vermiştir. 'Örneğin, konik skalerleştirme yöntemi karar vericinin tercihlerinden istek düzeyini ve amaç fonksiyonu ağırlıklarını modele yansıttığından 1. ve 2. sütunda + olarak işaretlenmiştir. Uygun değer alanı üzerine, dışbükeylik ve kompactlık şartı gerektirmediğinden 3. ve 4. sütunlar boş bırakılmıştır. Konik skalerleştirme yöntemine rezarvasyon düzeyleri kolaylıkla eklenebilir. Konik skalerleştirme yöntemine rezarvasyon düzeylerinin eklenmesi halinde ele alınan özellikler açısından eksiksiz bir yöntem olacağı sonucuna ulaşılabilir.' Bu nedenle bu çalışmada skalerleştirme yöntemi olarak konik skalerleştirme yöntemi ele alınmıştır.

3. ÇOK AMAÇLI KARESEL ATAMA PROBLEMLERİNİN ÇÖZÜMÜ İÇİN MATEMATİKSEL VE SEZGİSEL ÇÖZÜM YÖNTEMLERİ

Bu bölümde mQAP çözümü için kullanılan ve önerilen çözüm yaklaşımları ile kullanılan test problemleri verilmiştir.

3.1. Test Problemleri

Çalışmada Knowles ve Corne (2003) tarafından geliştirilen test problemlerine ağırlıklandırılmış toplam skalerleştirme yöntemi, konik skalerleştirme, NSGA-II ve melez yöntem ile çözüm aranmıştır. Test problemlerine http://www.cs.bham.ac.uk/~jdk/mQAP/#gens linkinden ulaşılabilir. (Erişim tarihi: 29.12.2015) Ele alınan test problemleri Çizelge 3.1'de gösterilmektedir.

Çizelge 3.1. *Test Problemleri*

Problem Adı	Lokasyon Sayısı	Amaç Sayısı	Index
KC10-2fl-1rl	10	2	Real-like
KC10-2fl-1uni	10	2	Uniform
KC10-2fl-2rl	10	2	Real-like
KC10-2fl-2uni	10	2	Uniform
KC10-2fl-3rl	10	2	Real-like
KC10-2fl-3uni	10	2	Uniform
KC10-2fl-4rl	10	2	Real-like
KC10-2fl-5rl	10	2	Real-like
KC20-2fl-1rl	20	2	Real-like
KC20-2fl-1uni	20	2	Uniform
KC20-2fl-2rl	20	2	Real-like
KC20-2fl-2uni	20	2	Uniform
KC20-2fl-3rl	20	2	Real-like
KC20-2fl-3uni	20	2	Uniform
KC20-2fl-4rl	20	2	Real-like
KC20-2fl-5rl	20	2	Real-like
KC30-2fl-1rl	30	2	Real-like
KC30-3fl-1rl	30	3	Real-like
KC30-3fl-1uni	30	3	Uniform
KC30-3fl-2rl	30	3	Real-like
KC30-3fl-2uni	30	3	Uniform
KC30-3fl-3rl	30	3	Real-like
KC30-3fl-3uni	30	3	Uniform

Kaynak: *http://www.cs.bham.ac.uk/~jdk/mQAP/#gens*

Çalışmada 10 lokasyon ve 2 amaçlı, 20 lokasyon ve 2 amaçlı, 30 lokasyon ve 2 amaçlı ile 30 lokasyon ve 3 amaçlı problemler ele alınmıştır.

Örnek olması adına KC10-2fl-1rl problemi tablosu Çizelge 3.2'de gösterilmektedir.

KC10-2fl-1rl probleminde üç adet matris bulunmaktadır. Bunlardan ilki uzaklık matrisi, diğer ikisi akış matrisidir. Problemi iki amaçlı yapan da akış matrisi sayısıdır.

Çizelge 3.2. *KC10-2fl-1rl Problemi Uzaklık ve Akış Matrisi*

	1	2	3	4	5	6	7	8	9	10
1	0	3	60	62	155	29	47	78	83	102
2	3	0	57	61	152	27	44	74	80	99
3	60	57	0	58	95	47	32	19	25	48
4	62	61	58	0	141	33	27	61	61	65
5	155	152	95	141	0	142	123	82	80	80
6	29	27	47	33	142	0	21	60	63	78
7	47	44	32	27	123	21	0	41	43	57
8	78	74	19	61	82	60	41	0	7	31
9	83	80	25	61	80	63	43	7	0	23
10	102	99	48	65	80	78	57	31	23	0
	1	2	3	4	5	6	7	8	9	10
1	0	0	132	0	2558	667	0	572	1	200
2	0	0	990	140	9445	1397	7	0	100	0
3	132	990	0	7	40	2213	0	1	0	0
4	0	140	7	0	58	0	1	0	4	70
5	2558	9445	40	58	0	3	0	0	0	0
6	667	1397	2213	0	3	0	139	3	5169	101
7	0	7	0	1	0	139	0	0	5659	0
8	572	0	1	0	0	3	0	0	3388	1982
9	1	100	0	4	0	5169	5659	3388	0	1023
10	200	0	0	70	0	101	0	1982	1023	0
	1	2	3	4	5	6	7	8	9	10
1	0	1	0	5379	0	0	1	0	329	856
2	1	0	2029	531	15	80	197	17	274	241
3	0	2029	0	1605	0	194	0	2	4723	0
4	5379	531	1605	0	24	68	0	0	4847	2205
5	0	15	0	24	0	1355	5124	1610	0	0
6	0	80	194	68	1355	0	549	0	151	2
7	1	197	0	0	5124	549	0	5955	0	0
8	0	17	2	0	1610	0	5955	0	553	710
9	329	274	4723	4847	0	151	0	553	0	758
10	856	241	0	2205	0	2	0	710	758	0

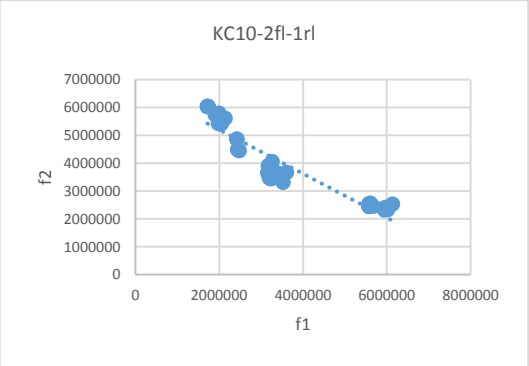
Kaynak: *http://www.cs.bham.ac.uk/~jdk/mQAP/#gens*

3.2. Ağırlıklandırılmış Toplam Skalerleştirme Yöntemi İle Çözüm

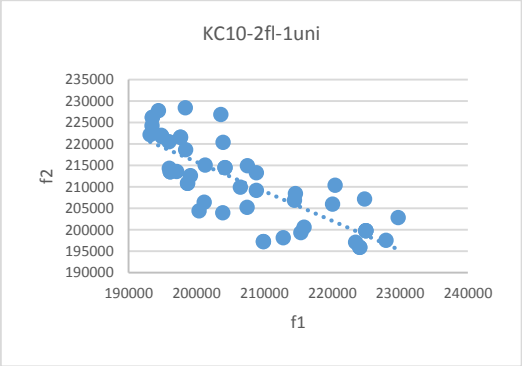
Test problemleri, GAMS (23.3 versiyonu) paket programı kullanılarak ağırlıklandırılmış toplam skalerleştirme yöntemi ile çözülmüştür. Ağırlıklar 1'den 50'ye

kadar alınmış ve sonuçta elde edilen f_1 ve f_2 sonuçlarının ortalaması konik skalerleştirme yönteminde referans noktası olarak alınmıştır.

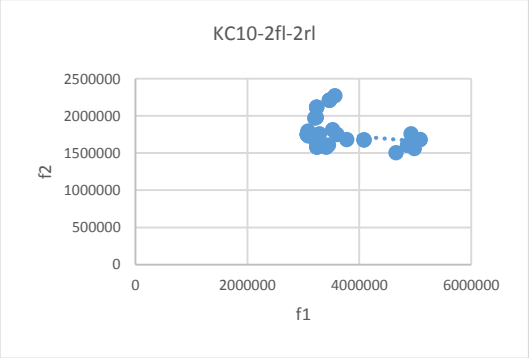
Ağırlıklandırılmış toplam skalerleştirme yöntemi (ATY) ile bulunan sonuçlar 2 amaçlı problemler için aşağıda gösterilmektedir.



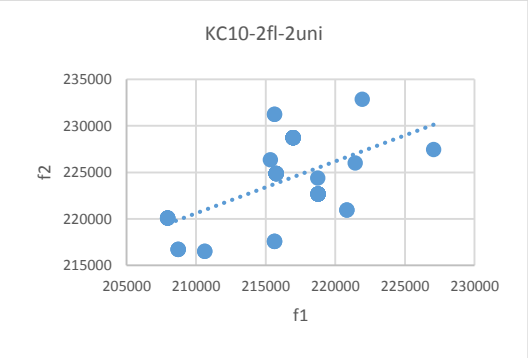
Şekil 3.1. *KC10-2fl-1rl ATY ile Çözüm*



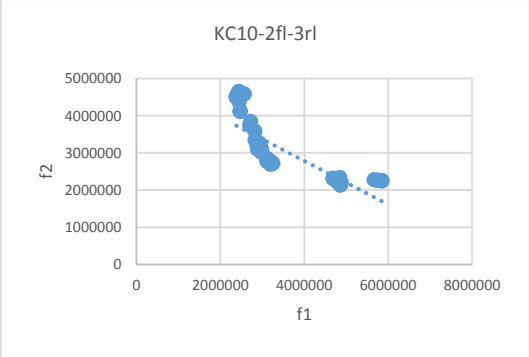
Şekil 3.2. *KC10-2fl-1uni ATY ile Çözüm*



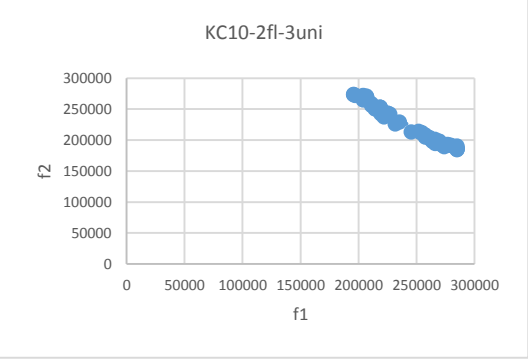
Şekil 3.3. *KC10-2fl-2rl ATY ile Çözüm*



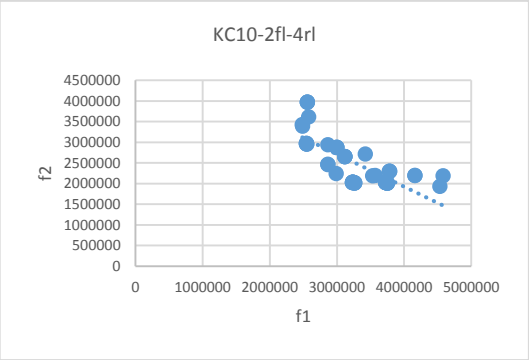
Şekil 3.4. *KC10-2fl-2uni ATY ile Çözüm*



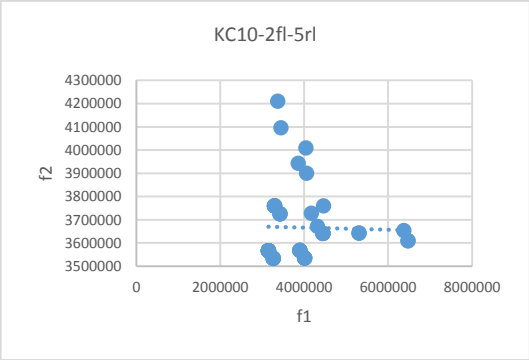
Şekil 3.5. *KC10-2fl-3rl ATY ile Çözüm*



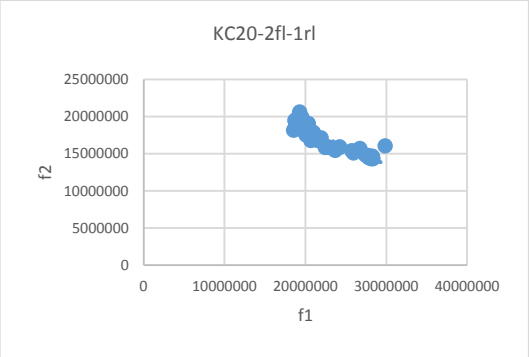
Şekil 3.6. *KC10-2fl-3uni ATY ile Çözüm*



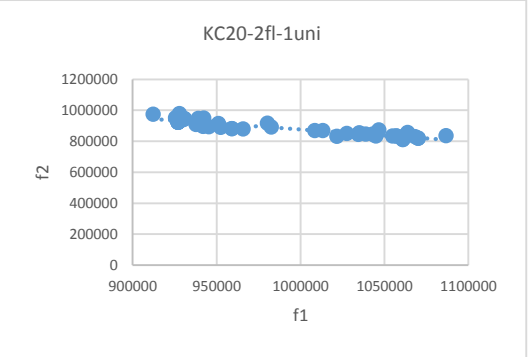
Şekil 3.7. *KC10-2fl-4rl ATY ile Çözüm*



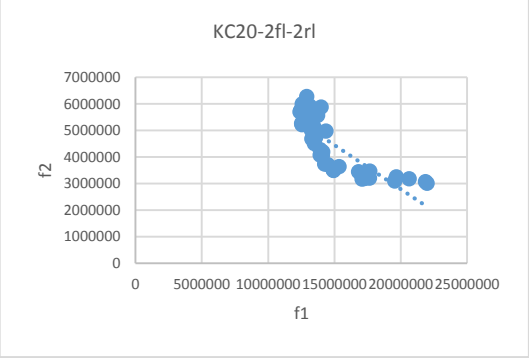
Şekil 3.8. *KC10-2fl-5rl ATY ile Çözüm*



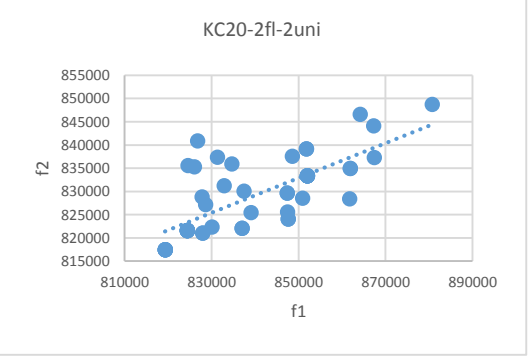
Şekil 3.9. *KC20-2fl-1rl ATY ile Çözüm*



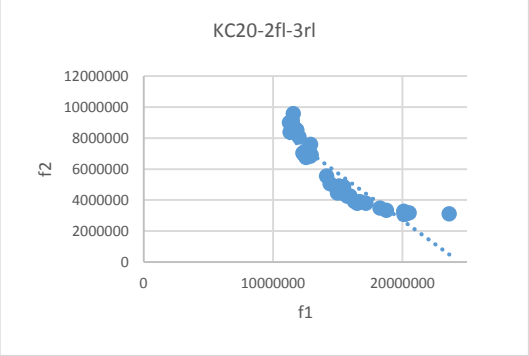
Şekil 3.10. *KC20-2fl-1uni ATY ile Çözüm*



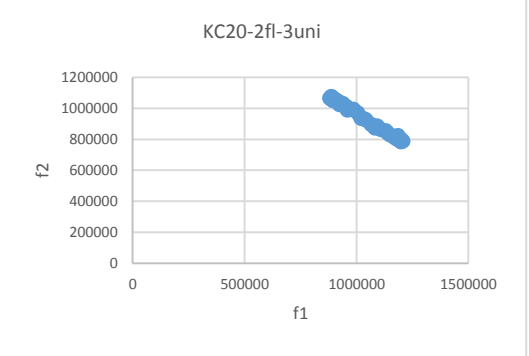
Şekil 3.11. *KC20-2fl-2rl ATY ile Çözüm*



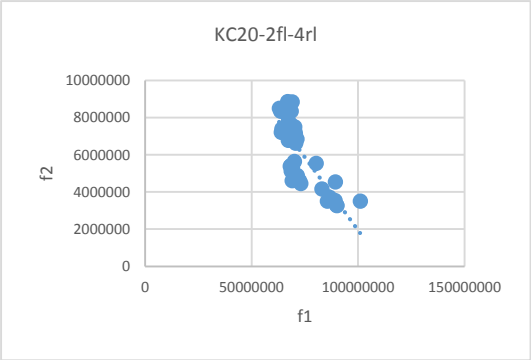
Şekil 3.12. *KC20-2fl-2uni ATY ile Çözüm*



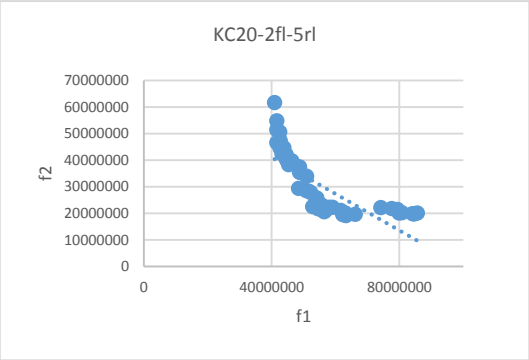
Şekil 3.13. *KC20-2fl-3rl ATY ile Çözüm*



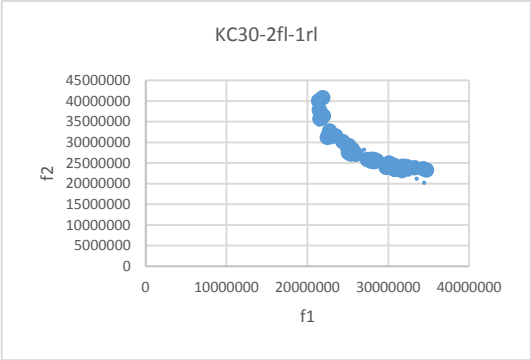
Şekil 3.14. *KC20-2fl-3uni ATY ile Çözüm*



Şekil 3.15. *KC20-2fl-4rl ATY ile Çözüm*



Şekil 3.16. *KC20-2fl-5rl ATY ile Çözüm*



Şekil 3.17. *KC30-2fl-1rl ATY ile Çözüm*

Ağırlıklandırılmış toplam skalerleştirme yöntemi ile pareto yüzeyler elde edilmiştir. Ancak sonuçların iyi olup olmadığı konik skalerleştirme yöntemi ile karşılaştırıldığında yorumlanacaktır.

3.3. Konik Skalerleştirme Yöntemi İle Çözüm

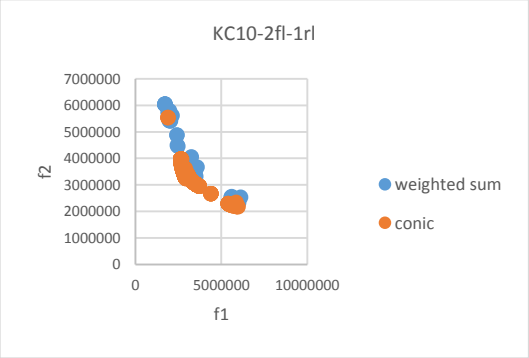
Ağırlıklandırılmış toplam skalerleştirme yönteminden elde edilen sonuçlarla konik skalerleştirme yöntemi test problemleri için GAMS (23.3 versiyonu) kodu yazılmış ve çözülmüştür. Ancak 3 amaçlı problemlerde çözüm süresi çok fazla olduğu için sadece KC30-3fl-1rl problemi çözülmüş ve problem çözümü 505 saat sürmüştür. Makul bir sürede çözülemediği için 3 amaçlı diğer problemler çözülmemiştir. Bunun yanı sıra 20 lokasyon ve 2 amaçlı problemlerin çözümü ortalama 11 saat, 10 lokasyon ve 2 amaçlı problemlerin çözümü de ortalama 10 dakika sürmüştür. Problemin boyutu arttıkça çözüm süresi de artmakta ve makul çözüm süresinden uzaklaşmaktadır.

Konik skalerleştirme fonksiyonu şu şekildedir:

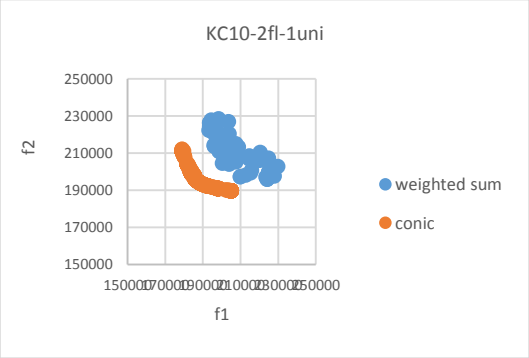
$$\min_{x \in X} \sum_{i=1}^{n} w_i (f_i(x) - r_i) + \alpha \sum_{i=1}^{n} |f_i(x) - r_i| \qquad (3.1)$$

Yazılan programda w_i'ler ağırlıklandırılmış toplam skalerleştirme yöntemindeki gibi 1 ile 50 arasında alınmıştır. r_i'ler ağırlıklandırılmış toplam skalerleştirme yönteminde bulunan f_i değerlerinin ortalaması olarak alınmış ve referans değer olarak belirlenmiştir. α ise karar verici tarafından belirlenen bir değişken olarak tanımlanmıştır.

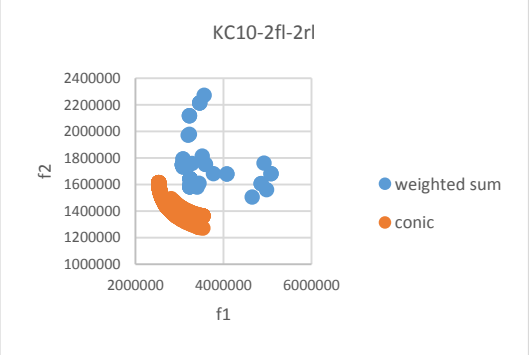
Konik skalerleştirme ile de 10 lokasyonlu ve 2 amaçlı, 20 lokasyonlu ve 20 amaçlı, 30 lokasyonlu ve 2 amaçlı ile KC30-3fl-1rl problemleri çözülmüş ve elde edilen sonuçlar ağırlıklandırılmış toplam skalerleştirme yönteminde elde edilen sonuçlar ile karşılaştırılmıştır. Karşılaştırma grafikleri aşağıda gösterilmektedir.



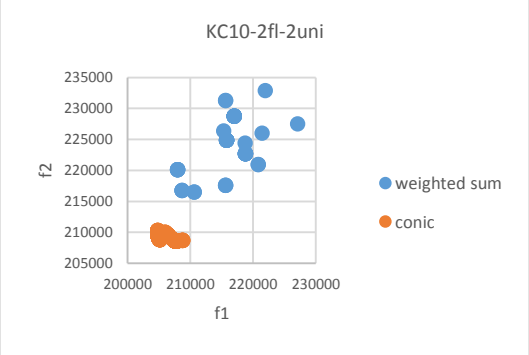
Şekil 3.18. *KC10-2fl-1rl ATY ile KSY karşılaştırma*



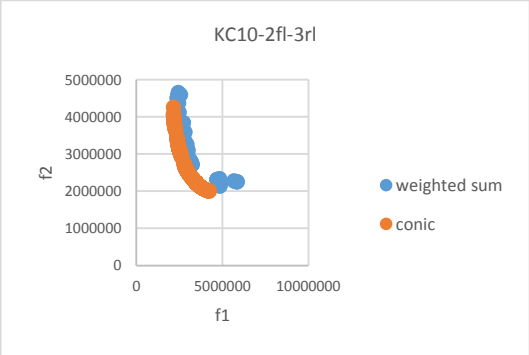
Şekil 3.19. *KC10-2fl-1uni ATY ile KSY karşılaştırma*



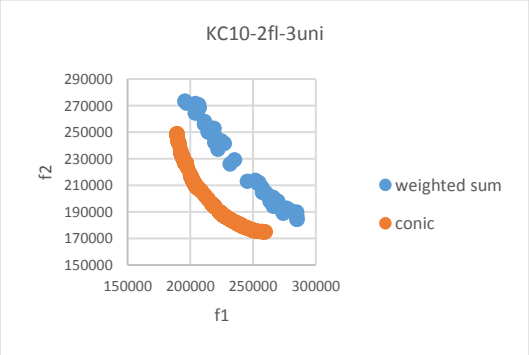
Şekil 3.20. *KC10-2fl-2rl ATY ile KSY karşılaştırma*



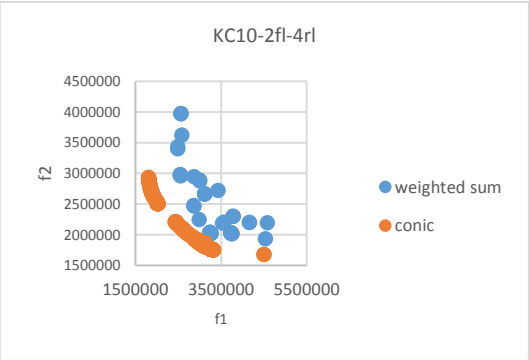
Şekil 3.21. *KC10-2fl-2uni ATY ile KSY karşılaştırma*



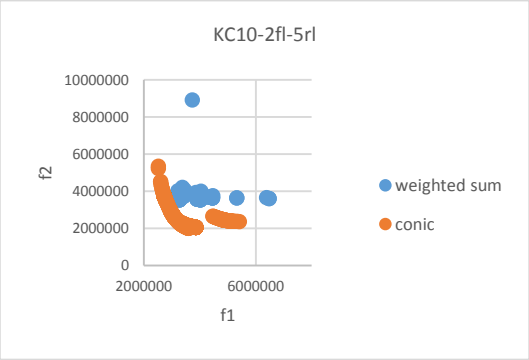
Şekil 3.22. *KC10-2fl-3rl ATY ile KSY karşılaştırma*



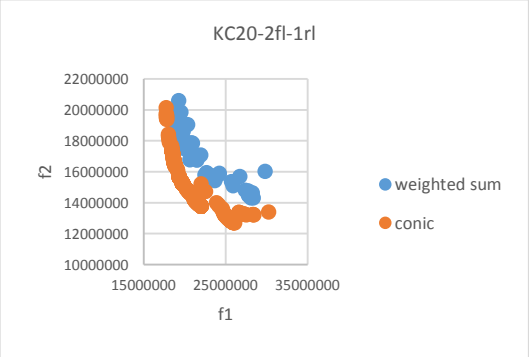
Şekil 3.23. *KC10-2fl-3uni ATY ile KSY karşılaştırma*



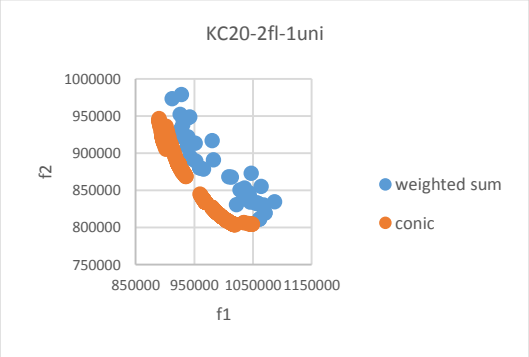
Şekil 3.24. *KC10-2fl-4rl ATY ile KSY karşılaştırma*



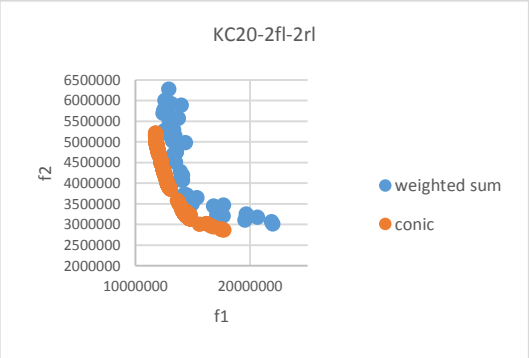
Şekil 3.25. *KC10-2fl-5rl ATY ile KSY karşılaştırma*



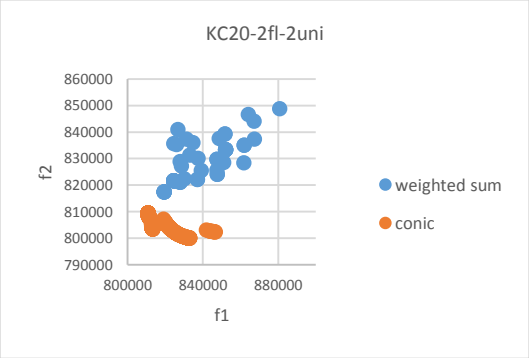
Şekil 3.26. *KC20-2fl-1rl ATY ile KSY karşılaştırma*



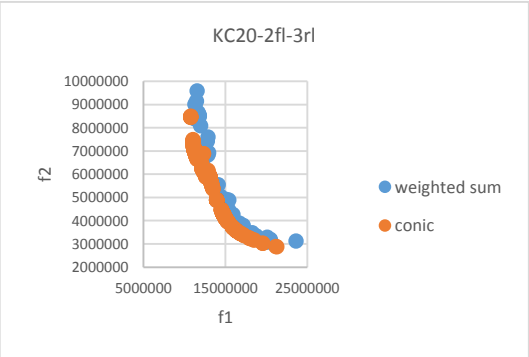
Şekil 3.27. *KC20-2fl-1uni ATY ile KSY karşılaştırma*



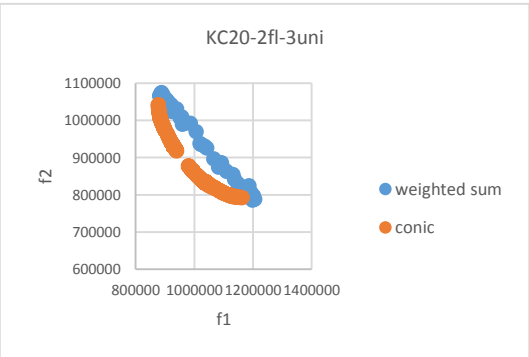
Şekil 3.28. *KC20-2fl-2rl ATY ile KSY karşılaştırma*



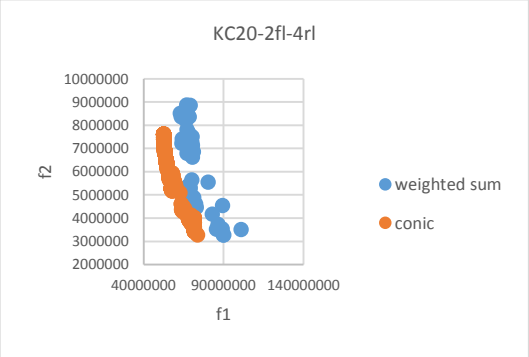
Şekil 3.29. *KC20-2fl-2uni ATY ile KSY karşılaştırma*



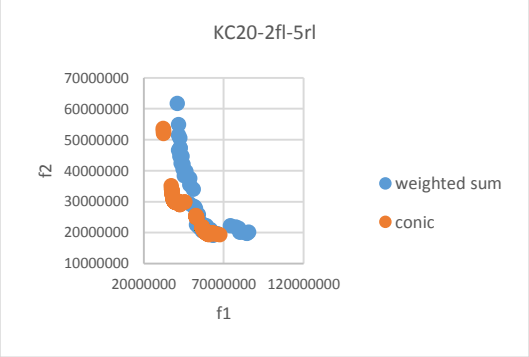
Şekil 3.30. *KC20-2fl-3rl ATY ile KSY karşılaştırma*



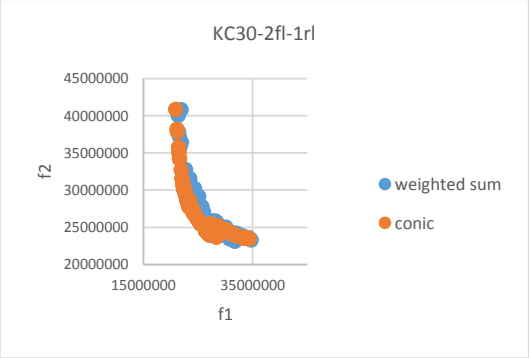
Şekil 3.31. *KC20-2fl-3uni ATY ile KSY karşılaştırma*



Şekil 3.32. *KC20-2fl-4rl ATY ile KSY karşılaştırma*



Şekil 3.33. *KC20-2fl-5rl ATY ile KSY karşılaştırma*

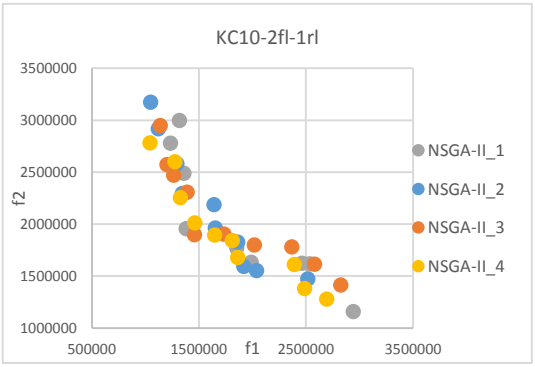


Şekil 3.34. *KC30-2fl-1rl ATY ile KSY karşılaştırma*

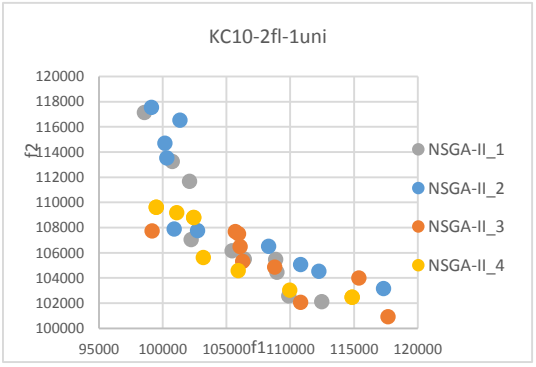
Grafiklerde de görüldüğü gibi Konik Skalerleştirme yöntemi Ağırlıklandırılmış Toplam Skalerleştirme Yöntemi ile kıyaslandığında daha çok pareto yüzey bulmaktadır.

3.4. NSGA-II ile Çözüm

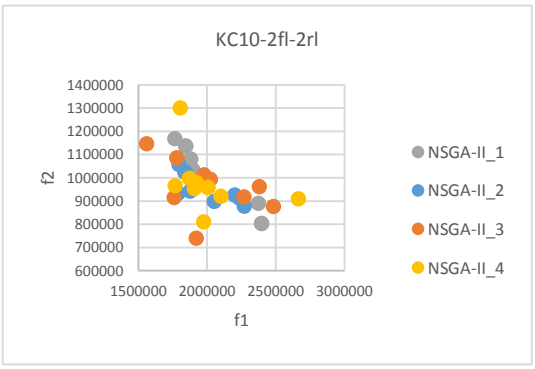
NSGA-II algoritması için Visual Basic Application kullanılmış ve model kodlanmıştır. Problem çözülürken çaprazlama oranı, mutasyon oranı ve popülasyon sayısı değiştirilmiştir ve aynı problem için NSGA-II'de 4 farklı sonuç elde edilmiştir. NSGA-II_1 olarak belirlenen modelde mutasyon oranı = 0,01, çaprazlama oranı = 0,5 ve popülasyon sayısı = 500; NSGA-II_2'de mutasyon oranı = 0,01, çaprazlama oranı = 0,75 ve popülasyon sayısı = 500; NSGA-II_3'de mutasyon oranı = 0,01, çaprazlama oranı = 0, 5 ve popülasyon sayısı = 1000 ve son olarak NSGA-II_4'te mutasyon oranı = 0,1, çaprazlama oranı = 0, 5 ve popülasyon sayısı = 500, tüm modeller için tekrar sayısı 100'dür. 4 farklı şekilde çözülen problemlerin pareto yüzeyleri aşağıda grafiklerle gösterilmektedir.



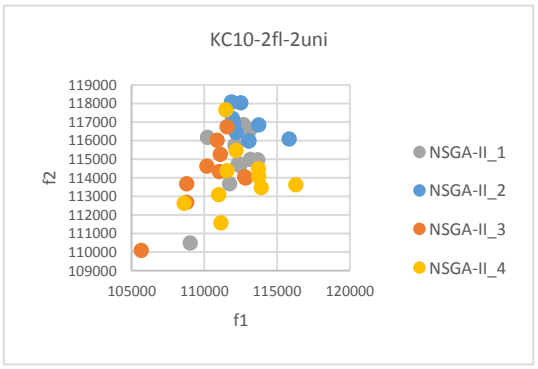
Şekil 3.35. *KC10-2fl-1rl NSGA-II karşılaştırma*



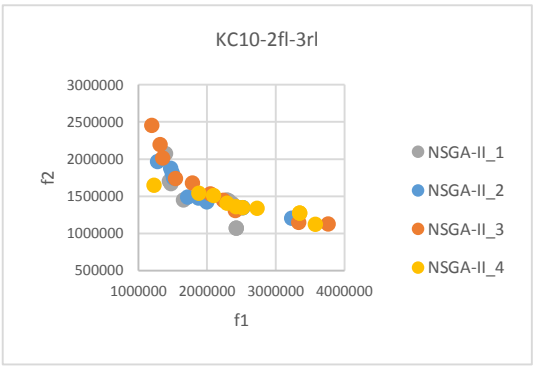
Şekil 3.36. *KC10-2fl-1uni NSGA-II karşılaştırma*



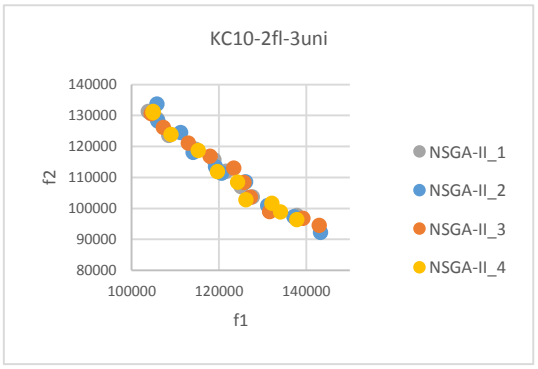
Şekil 3.37. *KC10-2fl-2rl NSGA-II karşılaştırma*



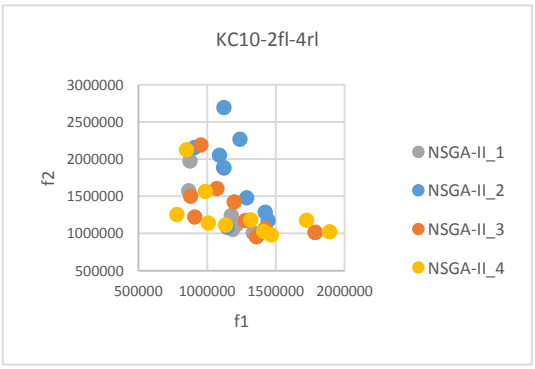
Şekil 3.38. *KC10-2fl-2uni NSGA-II karşılaştırma*



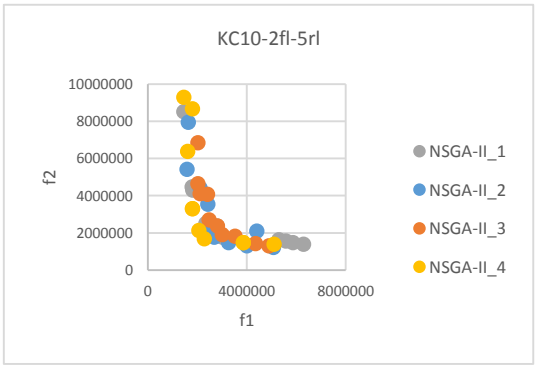
Şekil 3.39. *KC10-2fl-3rl NSGA-II karşılaştırma*



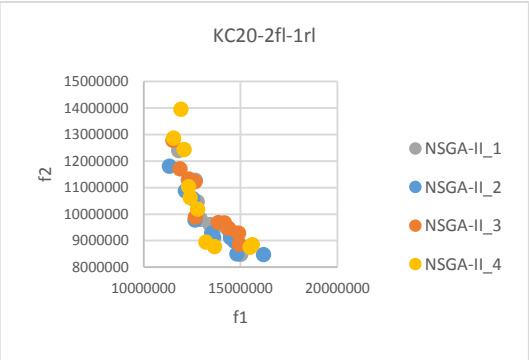
Şekil 3.40. *KC10-2fl-3uni NSGA-II karşılaştırma*



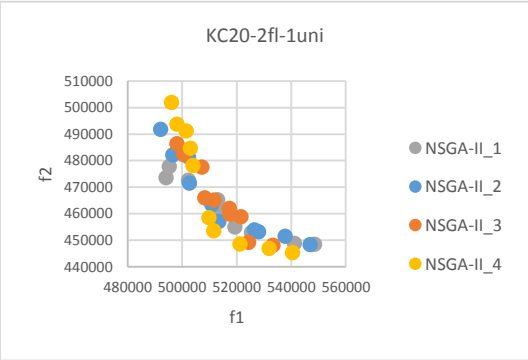
Şekil 3.41. *KC10-2fl-4rl NSGA-II karşılaştırma*



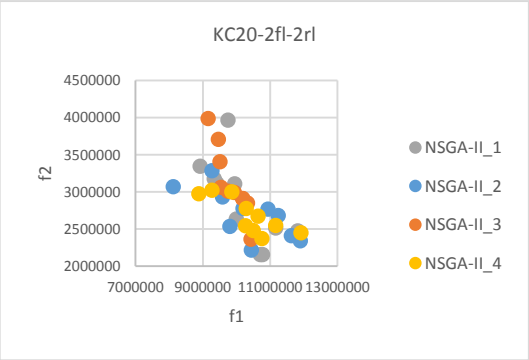
Şekil 3.42. *KC10-2fl-5rl NSGA-II karşılaştırma*



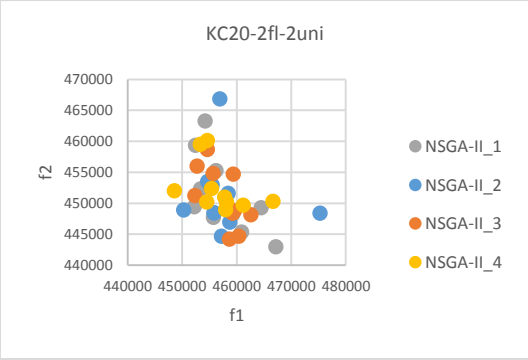
Şekil 3.43. *KC20-2fl-1rl NSGA-II karşılaştırma*



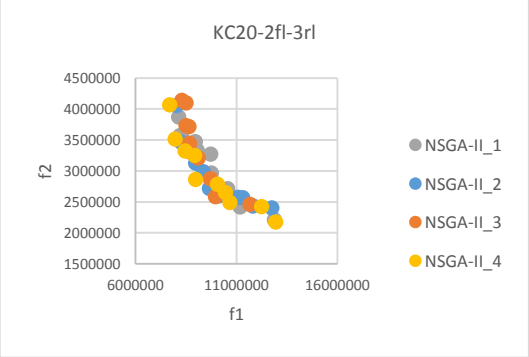
Şekil 3.44. *KC20-2fl-1uni NSGA-II karşılaştırma*



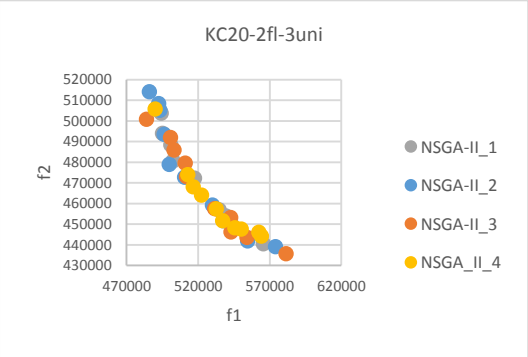
Şekil 3.45. *KC20-2fl-2rl NSGA-II karşılaştırma*



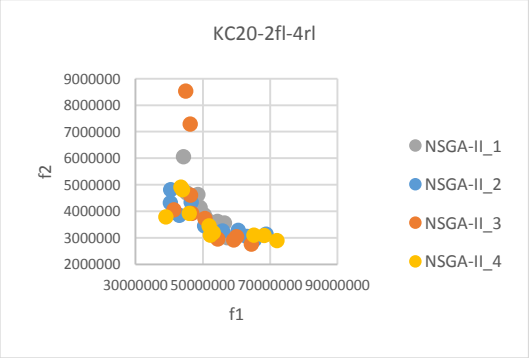
Şekil 3.46. *KC20-2fl-2uni NSGA-II karşılaştırma*



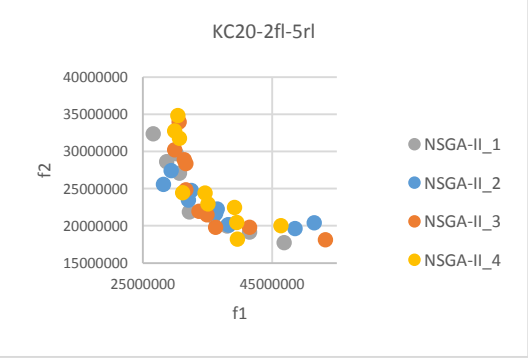
Şekil 3.47. *KC20-2fl-3rl NSGA-II karşılaştırma*



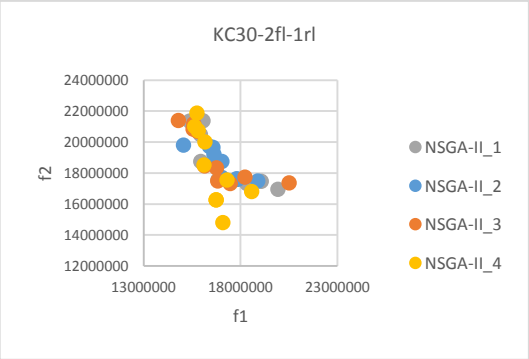
Şekil 3.48. *KC20-2fl-3uni NSGA-II karşılaştırma*



Şekil 3.49. *KC20-2fl-4rl NSGA-II karşılaştırma*



Şekil 3.50. *KC20-2fl-5rl NSGA-II karşılaştırma*



Şekil 3.51. *KC30-2fl-1rl NSGA-II karşılaştırma*

Grafiklerde görüldüğü gibi NSGA-II algoritmasında parametreler değiştirildiğinde farklı sonuçlar bulunmaktadır. NSGA-II algoritması ile tüm test problemleri için pareto yüzey bulunamamıştır. Bir sonraki bölümde Konik Skalerleştirme yöntemi ve NSGA-II algoritması ile elde edilen sonuçlar karşılaştırılmaktadır.

3.5. Konik Skalerleştirme ve NSGA-II Algoritmasının Karşılaştırılması

Knowles ve Corne 2002 yılında yaptıkları çalışmada parametre (permutasyon) uzayındaki uzaklık ölçütünün bir gösterge olarak alınabileceğinden bahsetmişlerdir. Bu kapsamda 1999 yılında Bachelet tarafından geliştirilen diameter metriğini ele almış ve performans ölçütü olarak belirlemişlerdir. Bachelet P popülasyonunun diameter formülünü aşağıdaki gibi tanımlamıştır:

$$dmm(P) = \frac{\sum_{\pi \epsilon P} \sum_{\mu \epsilon P} dist\ (\pi,\mu)}{\left| P \right|^2} \tag{3.2}$$

Formülde $dist\ (\pi,\mu)$ bir çözümün bir diğerine dönüştürülmesi için uygulanması gereken ikili değişimin en küçük sayısı olarak π ve μ çözümleri arasındaki uzaklık ölçüsüdür. P ise populasyon büyüklüğüdür. İki çözüm arasındaki uzaklığın popülasyonun karesine bölümü diameter metriğini vermektedir.

Bachelet (1999) çalışmasında diameterın büyük olmasının çeşitliliğin fazla olduğunu gösterdiğini dile getirmiştir. Bu nedenle karşılaştırma yüksek olanın daha iyi sonuç verdiği düşüncesi ile yapılmaktadır.

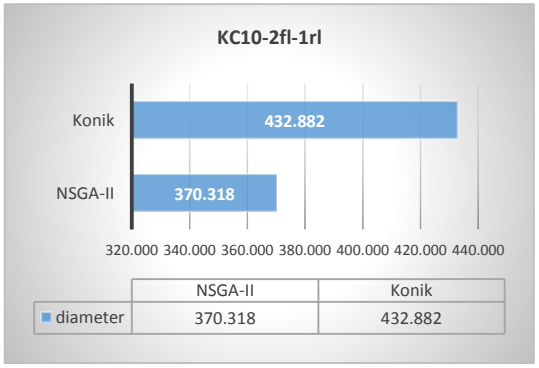
Konik skalerleştirme ve NSGA-II algoritması da iki amaçlı problemler için diameterlar hesaplanarak karşılaştırılmıştır. 3.4. NSGA-II bölümünde bahsedilen 4 farklı çözümün ortalaması alınarak tek bir NSGA-II değeri elde edilmiştir.

Karşılaştırma yapılan tablo izleyen sayfada Çizelge 3.3'de gösterilmektedir.

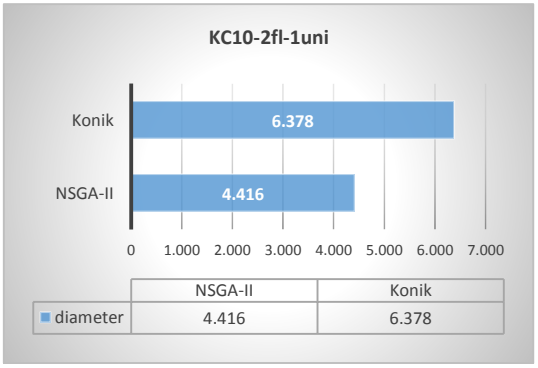
Çizelge 3.3. *Konik Skalerleştirme – NSGA-II diameter karşılaştırma*

Problem	NSGA-II	Konik Skalerleştirme
KC10-2fl-1rl	370.318	432.882
KC10-2fl-1uni	4.416	6.378
KC10-2fl-2rl	325.855	155.762
KC10-2fl-2uni	4.458	782
KC10-2fl-3rl	367.312	401.294
KC10-2fl-3uni	4.842	16.860
KC10-2fl-4rl	389.446	380.342
KC10-2fl-5rl	1.610.559	515.146
KC20-2fl-1rl	956.204	1.639.296
KC20-2fl-1uni	10.465	30.590
KC20-2fl-2rl	776.318	760.189
KC20-2fl-2uni	9.295	5.233
KC20-2fl-3rl	718.149	1.176.523
KC20-2fl-3uni	23.176	54.994
KC20-2fl-4rl	4.134.239	2.551.831
KC20-2fl-5rl	4.652.019	5.551.017
KC30-2fl-1rl	1.183.783	1.973.953

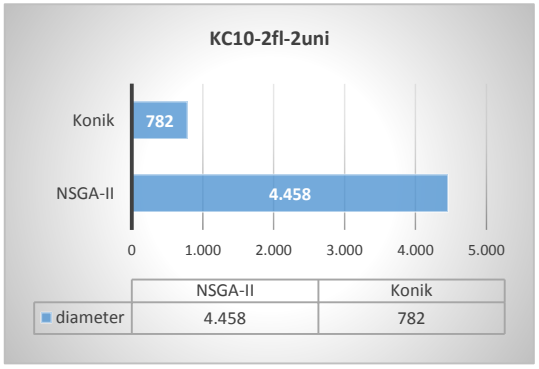
Ayrıca örnek olması açısından karşılaştırma sonuçlarından 8 tanesi aşağıdaki grafiklerde de gösterilmektedir.



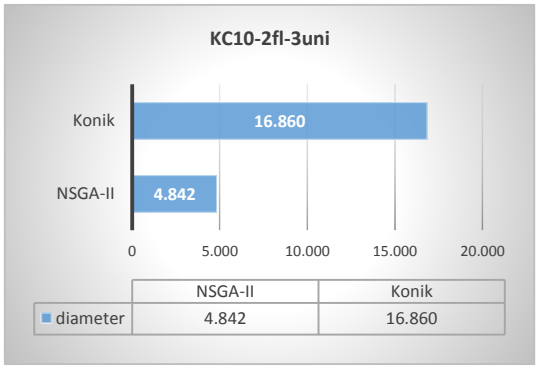
Şekil 3.52. *KC10-2fl-1rl NSGA-II – KSY diameter karşılaştırma*



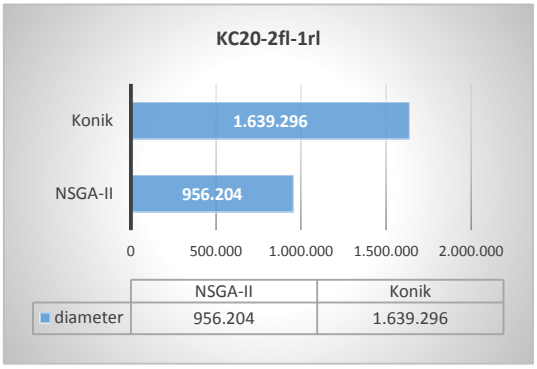
Şekil 3.53. *KC10-2fl-1uni NSGA-II – KSY diameter karşılaştırma*



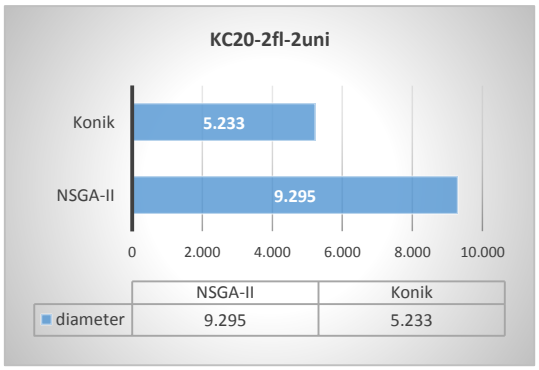
Şekil 3.54. *KC10-2fl-2uni NSGA-II – KSY diameter karşılaştırma*



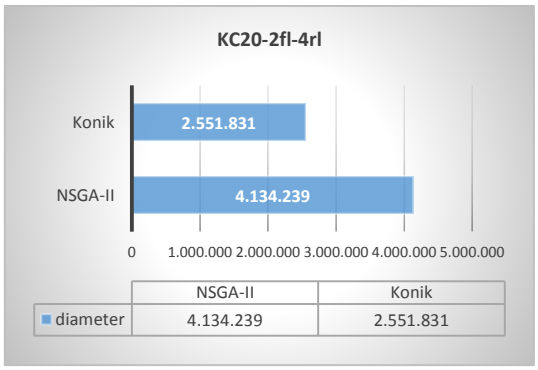
Şekil 3.55. *KC10-2fl-3uni NSGA-II – KSY diameter karşılaştırma*



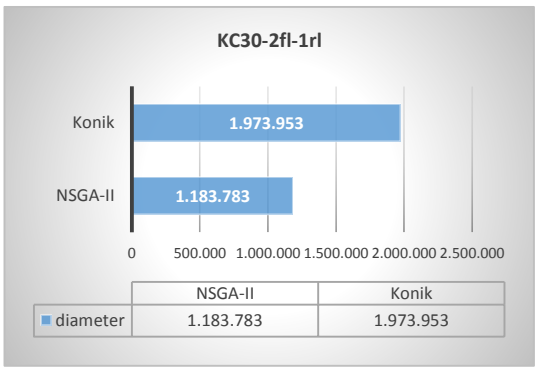
Şekil 3.56. *KC20-2fl-1rl NSGA-II – KSY diameter karşılaştırma*



Şekil 3.57. *KC20-2fl-2uni NSGA-II – KSY diameter karşılaştırma*



Şekil 3.58. *KC20-2fl-4rl NSGA-II – KSY diameter karşılaştırma*



Şekil 3.59. *KC30-2fl-1rl NSGA-II – KSY diameter karşılaştırma*

Grafiklerde de görüldüğü gibi konik skalerleştirme yöntemi ile NSGA-II karşılaştırıldığında ikisinin de farklı problemlerde daha iyi sonuçlar verebildiği ortaya çıkmaktadır. Ele alınan on yedi test probleminden on tanesinde konik skalerleştirme yöntemi, yedi tanesinde de NSGA-II daha iyi sonuç vermiştir. Bir sonraki bölümde bu iki yöntemin güçlü yönlerinin alındığı melez algoritma anlatılacaktır.

Çözüm yöntemleri süre olarak karşılaştırılmıştır. Elde edilen sonuçlar Çizelge 3.4'te gösterilmektedir.

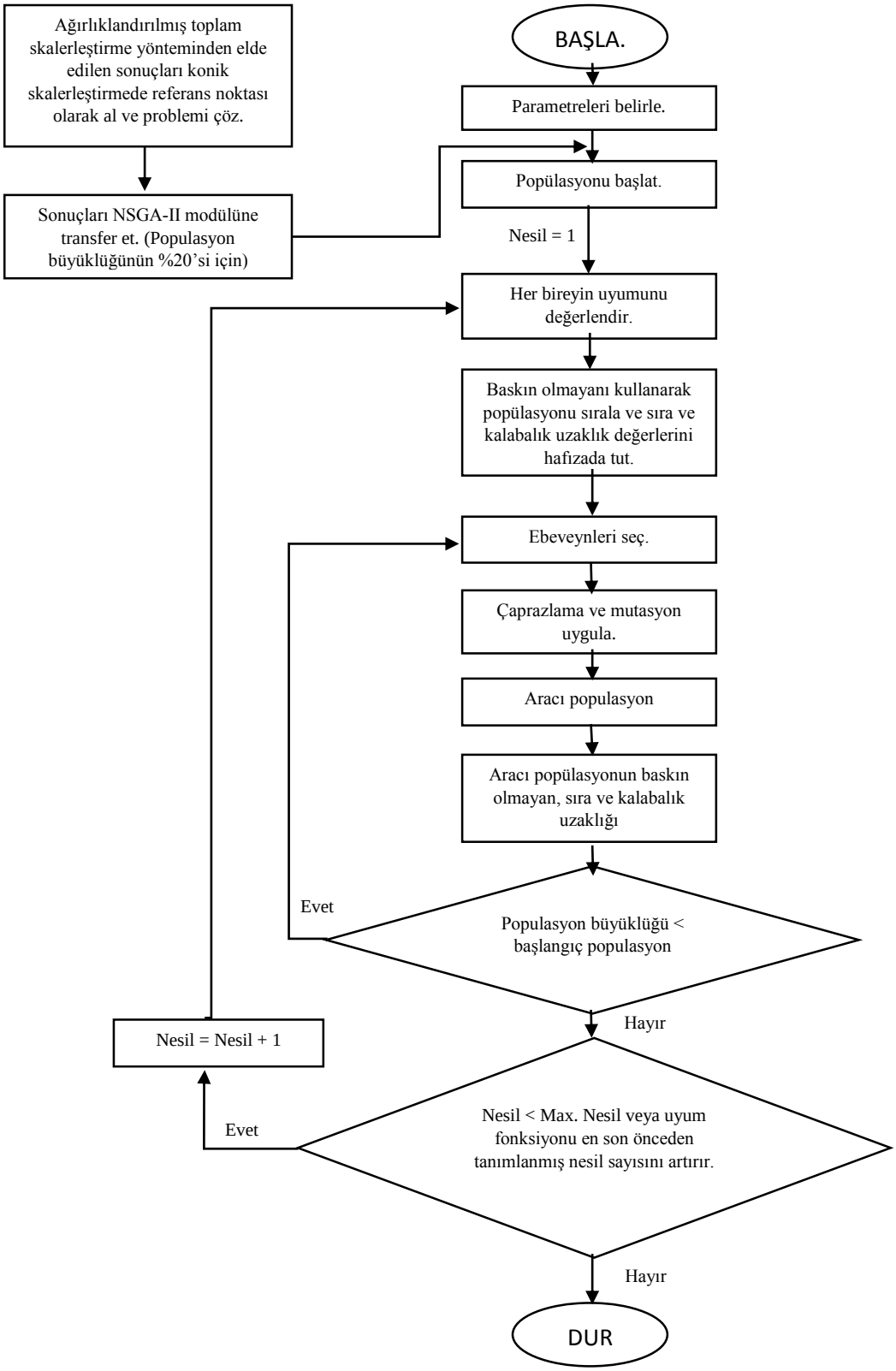
Çizelge 3.4. *NSGA-II – Konik Skalerleştirme Çözüm Süresi Karşılaştırma*

Problem	NSGA-II (dk.)	Konik Skalerleştirme (dk.)
KC10-2fl-1rl	2,2	10,0
KC10-2fl-1uni	2,0	7,0
KC10-2fl-2rl	1,7	3,0
KC10-2fl-2uni	1,4	9,0
KC10-2fl-3rl	1,5	14,8
KC10-2fl-3uni	1,4	10,0
KC10-2fl-4rl	1,4	11,8
KC10-2fl-5rl	1,3	4,0
KC20-2fl-1rl	2,2	170,0
KC20-2fl-1uni	2,2	513,0
KC20-2fl-2rl	2,3	232,0
KC20-2fl-2uni	2,3	150,0
KC20-2fl-3rl	2,3	1.405,0
KC20-2fl-3uni	2,2	2.020,0
KC20-2fl-4rl	0,8	177,0
KC20-2fl-5rl	0,9	226,0
KC30-2fl-1rl	3,4	3.216,0
KC30-3fl-1rl	3,5	30.115,0

Yukarıdaki tabloda görüldüğü üzere konik skalerleştirme yöntemi ile problemler makul sürede çözülememiştir. NSGA-II algoritmasında ise en fazla 3,5 dakikada problemler çözülmüştür.

3.6. Melez Algoritma

Test problemleri konik skalerleştirme yöntemi ile çözülmüş ve konik skalerleştirmeden elde edilen sonuçlar NSGA-II algoritmasına girdi olacak şekilde melez algoritma oluşturulmuştur. Konik skalerleştirme ile problemler GAMS paket programında çözülmüş ve sonuçlar VBA'ya aktarılmış, melez algoritma da VBA ile kodlanmıştır. Oluşturulan algoritmanın yapısı aşağıdaki şekilde gösterilmektedir.



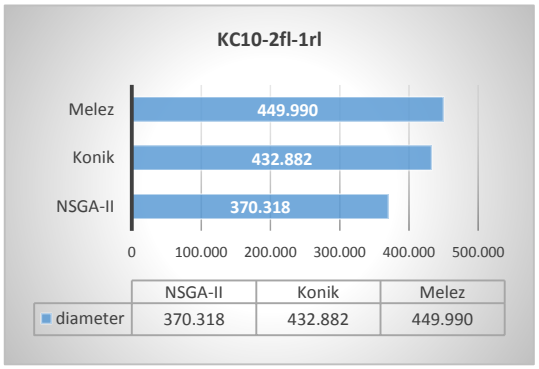
Şekil 3.60. *Melez Algoritma Yapısı*

Bu bölümde konik skalerleştirme, NSGA-II ve geliştirilen melez algoritma diameter metriği ile karşılaştırılmıştır. İki amaçlı test problemleri için bulunan diameterların yöntem bazında değerlendirme sonuçları aşağıdaki tabloda gösterilmektedir.

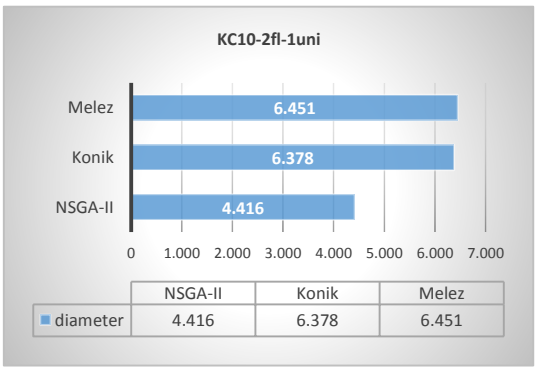
Çizelge 3.5. *NSGA-II – Konik Skalerleştirme – Melez Algoritma Diameter Karşılaştırma*

Problem	NSGA-II	Konik Skalerleştirme	Melez Algoritma
KC10-2fl-1rl	370.318	432.882	449.990
KC10-2fl-1uni	4.416	6.378	6.451
KC10-2fl-2rl	325.855	155.762	312.896
KC10-2fl-2uni	4.458	782	20.980
KC10-2fl-3rl	367.312	401.294	417.904
KC10-2fl-3uni	4.842	16.860	16.013
KC10-2fl-4rl	389.446	380.342	354.249
KC10-2fl-5rl	1.610.559	515.146	1.672.450
KC20-2fl-1rl	956.204	1.639.296	1.374.347
KC20-2fl-1uni	10.465	30.590	58.935
KC20-2fl-2rl	776.318	760.189	764.382
KC20-2fl-2uni	9.295	5.233	80.814
KC20-2fl-3rl	718.149	1.176.523	890.217
KC20-2fl-3uni	23.176	54.994	63.379
KC20-2fl-4rl	4.134.239	2.551.831	4.664.329
KC20-2fl-5rl	4.652.019	5.551.017	4.450.468
KC30-2fl-1rl	1.183.783	1.973.953	1.337.680

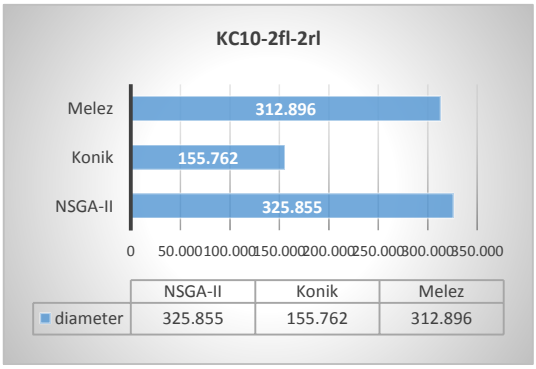
Karşılaştırma grafikleri aşağıda gösterilmektedir.



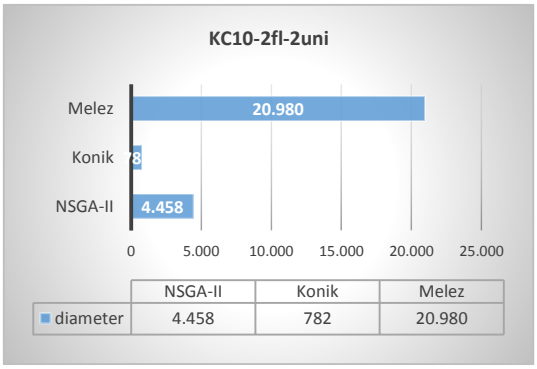
Şekil 3.61. *KC10-2fl-1rl NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



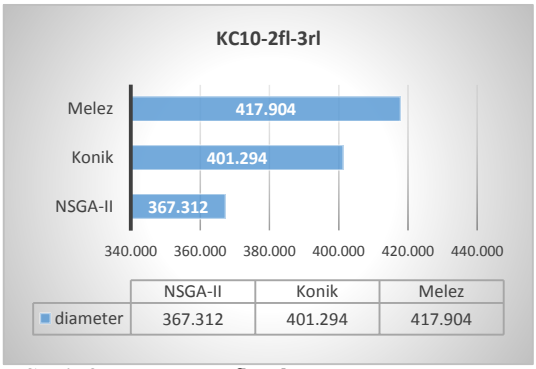
Şekil 3.62. *KC10-2fl-1uni NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



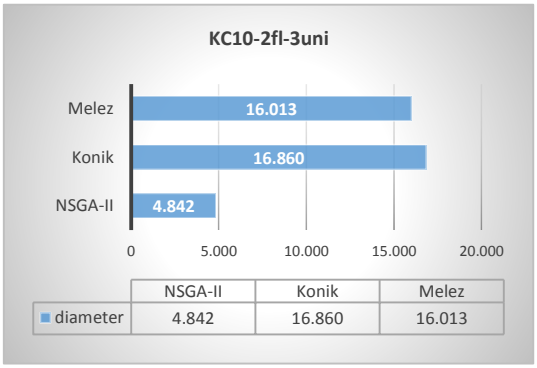
Şekil 3.63. *KC10-2fl-2rl NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



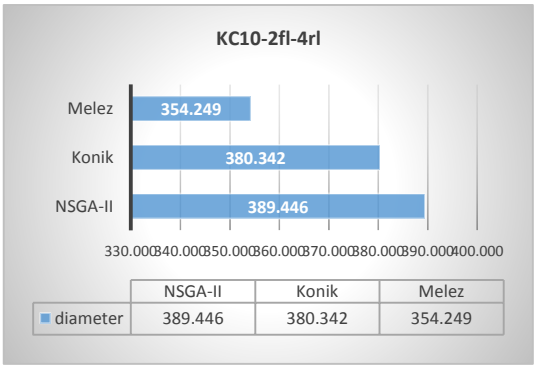
Şekil 3.64. *KC10-2fl-2uni NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



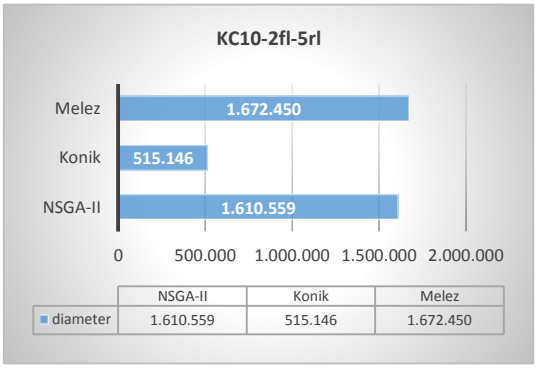
Şekil 3.65. *KC10-2fl-3rl NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



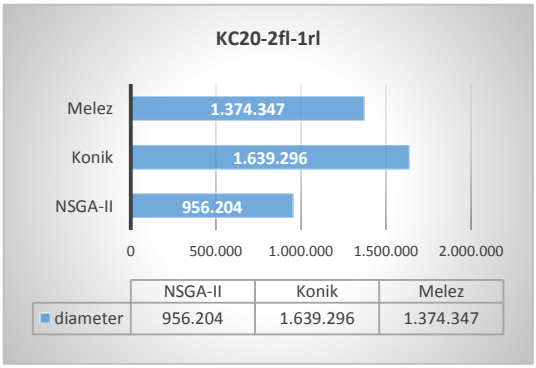
Şekil 3.66. *KC10-2fl-3uni NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



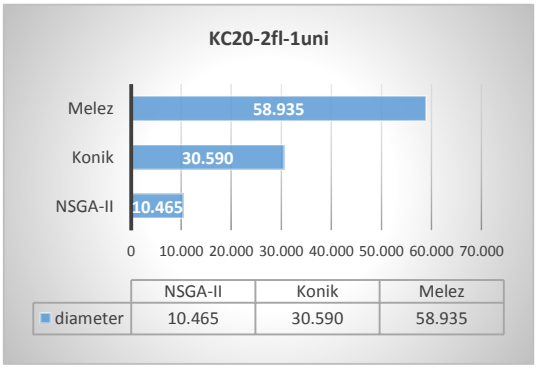
Şekil 3.67. *KC10-2fl-4rl NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



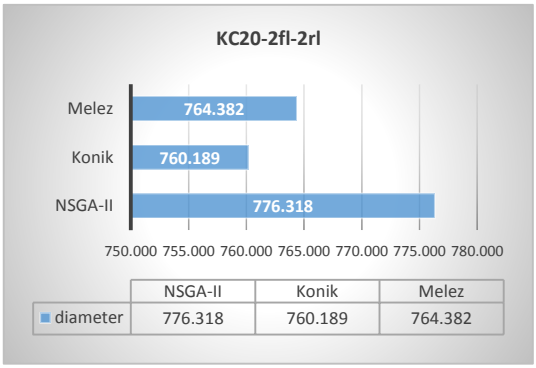
Şekil 3.68. *KC10-2fl-5rl NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



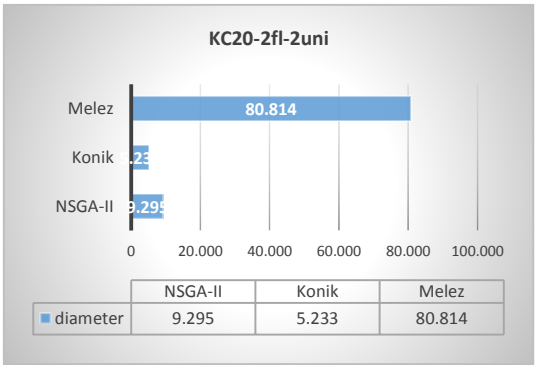
Şekil 3.69. *KC20-2fl-1rl NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



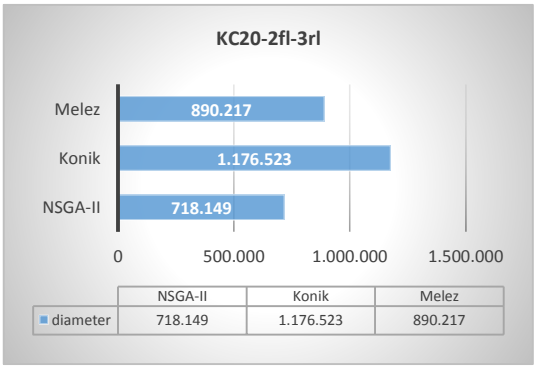
Şekil 3.70. *KC20-2fl-1uni NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



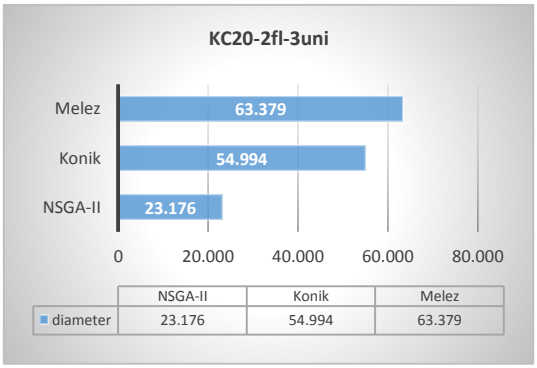
Şekil 3.71. *KC20-2fl-2rl NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



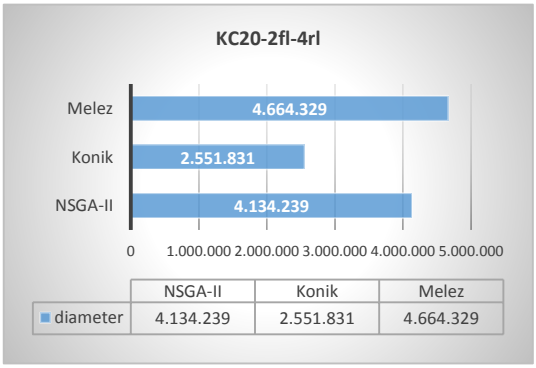
Şekil 3.72. *KC20-2fl-2uni NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



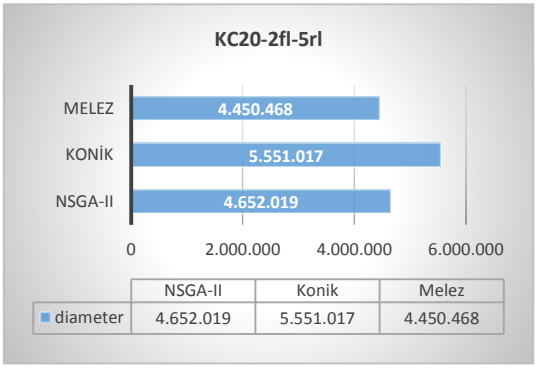
Şekil 3.73. *KC20-2fl-3rl NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



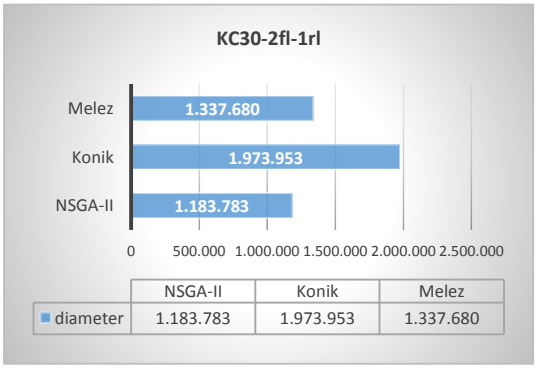
Şekil 3.74. *KC20-2fl-3uni NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



Şekil 3.75. *KC20-2fl-4rl NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



Şekil 3.76. *KC20-2fl-5rl NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*



Şekil 3.77. *KC30-2fl-1rl NSGA-II – KSY – Melez Algoritma diameter karşılaştırma*

Sonuçlar değerlendirildiğinde üç farklı yöntem ile çözülen on yedi test probleminden dokuz tanesinde melez algoritma, beş tanesinde konik skalerleştirme ve üç tanesinde NSGA-II algoritması daha iyi sonuç vermiştir. Geliştirilen melez algoritmanın diğer yöntemlerle rekabet edebilir düzeyde olduğu görülmüştür.

4. SONUÇLAR

Çalışmada çok amaçlı problemlerde çözüm yöntemi olarak kullanılan konik skalerleştirme ve NSGA-II yöntemi ele alınmış, mQAP test problemleri bu iki yöntemle çözülmüş ve sonuçta iki yöntemin de farklı problemlerde daha iyi sonuç verebildiği görülmüştür. Buradan yola çıkılarak her iki yöntemin güçlü yönleri ele alınarak melez bir algoritma geliştirilmiş ve aynı test problemleri melez algoritma ile de çözülmüştür. Çalışma sonucunda melez algoritmanın iyi sonuçlar verdiği görülmüştür. Konik skalerleştirme, NSGA-II ve melez algoritma karşılaştırılmış ve çözülen on yedi problemin dokuz tanesinde melez algoritmanın, beş tanesinde konik skalerleştirmenin ve üç tanesinde NSGA-II'nin daha iyi sonuçlar verdiği görülmüştür. Ayrıca, daha önce literatürde Knowles ve Corne (2003) ile Day (2005) tarafından yapılan çalışmalar sonucunda elde edilen diameter metrik bulguları konik skalerleştirme yöntemi ve melez algoritma ile elde edilen sonuçlarla karşılaştırılmış, elde edilen sonuçlar Çizelge 3.6'da gösterilmiştir.

Çizelge 3.6. *Literatür – Konik Skalerleştirme – Melez Algoritma Karşılaştırma*

Problem	Knowles-Corne	Day, MOMGA-II	Day, MOMGA-IIa	Konik Skalerleştirme	Melez algoritma
KC10-2fl-1rl	7.000	-	-	432.882	449.990
KC10-2fl-1uni	7.000	-	-	6.378	6.451
KC10-2fl-3uni	8.000	-	-	16.860	16.013
KC20-2fl-1uni	15.000	11.400	13.700	30.590	58.935
KC20-2fl-2uni	14.000	7.200	3.670	5.233	80.814
KC20-2fl-3uni	16.000	12.300	15.500	54.994	63.379

Çalışma, oluşturulan yeni melez algoritmanın diğer yöntemlerle rekabet edebilir düzeyde olduğunu ortaya çıkarmıştır.

KAYNAKÇA

Acan, A. ve Ünveren, A. (2005). Evolutionary multiobjective optimization with a segment-based external memory support for the multiobjective quadratic assignment problem. *IEEE Congress on Evolutionary Computation*, 3, 2723-2729.

Almeida, C. P., Gonçalves, R. A., Goldbarg, E. F., Goldbarg, M. C. ve Delgado, M. R. (2014). Transgenetic algorithms for the multi-objective quadratic assignment problem. *Brazilian Conference on Intelligent Systems*, 312-317.

Ammar, E. E. (2008). On solutions of fuzzy random multiobjective quadratic programming with applications in portfolio problem. *Information Sciences*, 178, 468-484.

Ammar, E. E. (2009). On fuzzy random multiobjective quadratic programming. *European Journal of Operational Research*, 193, 329-341.

Bachelet, V. (1999). Metaheuristiques paralleles hybrides: Application au probleme d'affectation quadratique.

Coello, C. A., Lamont, G. B. ve Van Veldhuizen, D. A. (2007). *Evolutionary Algorithms for Solving Multi-Objective Problems.* New York: Springer, Verlag Inc.

Day, R. O. ve Lamont, G. B. (2005). Multiobjective quadratic assignment problem solved by an explicit building block search algorithm MOMGA-IIa. *Springer-Verlag Berlin Heidelberg*, 3448, 91-100.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on evolutionary computation*, 6(2), 182-197.

Dickey, J. W. ve Hopkins, J. W. (1972). Campus building arrangement using Topaz. *Transportation Research*, 6, 59-68.

Drugan, M. (2014). Multi-objective quadratic assignment problem instances generator with a known optimum solution. *Springer International Publishing Switzerland*, 8672, 559–568.

Francis, R. L. ve White, J. A. (1974). Facility layout and location: An analytical approach. *Prentice-Hall, Englewood Cliffs, NJ.*

Garrett, D. ve Dasgupta, D. (2006). Analyzing the performance of hybrid evolutionary algorithms for multiobjective quadratic assignment Problem, 1710-1717. Canada.

Garrett, D. ve Dasgupta, D. (2009). An empirical comparison of memetic algorithm strategies on the multiobjective quadratic assignment problem. *IEEE Sympossium on Computational Intelligence in Multi Criteria Decision Making*, 80-87.

Gass, S. ve Saaty, T. (1955). The computational algorithm for the parametric objective function. *Naval Research Logistics Quarterly*, 39-45.

Ghoreishi, S. N., Sorensen, J. C. ve Jorgensen, B. N. (2015). Comparative study of evolutionary multi-objective optimization algorithms for a non-linear greenhouse climate control problem. *IEEE*.

Gutiérrez, E. ve Brizuela, C. A. (2007). An experimental study of the multi-objective Go with the winners algorithm on the biobjective QAP with correlated flow matrices. *IEEE Congress on Systems, Man, and Cybernetics*, 1476-1481.

Heffley, D. R. (1972). The quadratic assignment problem: A note. *Econometrica*, 40(6), 1155-1163.

Heffley, D. R. (1980). Decomposition of the Koopmans–Beckmann problem. *Regional Science and Urban Economics*, 10(4), 571-580.

Holland, J. H. (1975). Adaptation in natural and artificial systems. *MIT Press.*

Horn, J., Nafpliotis, N. ve Goldberg, D. E. (1994). A Niched Pareto Genetic Algorithm for multiobjective optimization. *Proceedings of the First IEEE Conference Evolutionary Computation*, 82-87.

Ibáñez, M. L., Paquete, L. ve Stützle, T. (2005). *Hybrid population-based algorithms for the bi-objective quadratic assignment problem.* Netherlands: Kluwer Academic Publishers.

Kasımbeyli, R. (2013). A conic scalarization method in multi-objective optimization. *Journal of Global Optimization*, 279-297.

Knowles, J. ve Corne, D. (1999). The Pareto Archived Evolution Strategy : A new baseline algorithm for pareto multiobjective optimisation. *Proceedings of the 1999 Congress on Evolutionary Computation,* 98-105.

Knowles, J. ve Corne, D. (2002). Towards landscape analyses to inform the design of a hybrid local search for the multiobjective quadratic assignment problem. *Soft Computing Systems: Design, Management and Applications*, 271-279, Amsterdam.

Knowles, J. ve Corne, D. (2003). Instance generator and test suites for the multiobjective quadratic assignment problem. *Evolutionary Multi-Criterion Optimization (EMO) Second International Conference,* Portugal.

Koopmans, T. C. ve Beckmann, B. C. (1957). Assignment problems and the location of economic activities. *Econometrica*, 53-76.

Li, H. ve Silva, D. L. (2009). An elitist GRASP metaheuristic for the multi-objective quadratic assignment problem. *Springer-Verlag Berlin Heidelberg*, 5467, 481-494.

Liefooghe, A., Verel, S. ve Hao, J. K. (2014). A hybrid metaheuristic for multiobjective unconstrained binary quadratic programming. *Applied Soft Computing*, 16, 10-19.

Masri, H., Abdelkhalek, O. ve Krichen, S. (2014). A comparison of multiple objective evolutionary algorithms for solving the multi-objective node placement problem. *IEEE,* 152-157.

Özkale, C. ve Fığlalı, A. (2013). Evaluation of the multiobjective ant colony algorithm performances on biobjective quadratic assignment problems. *Applied Mathematical Modelling*, 37, 7822–7838.

Paquete, L. ve Stützle, T. (2006). A study of stochastic local search algorithms for the biobjective QAP with correlated flow matrices. *European Journal of Operational Research*, 169, 943-959.

Steinberg, L. (1961). The backboard wiring problem: A placement algorithm. *SIAM*, 3, 37-50.

Üstün, Ö. (2007). Çok Amaçlı Portföy Optimizasyon Problemi ve Çözüm Yaklaşımları. Eskişehir.

Zhao, M., Abraham, A., Grosan, C. ve Liu, H. (2008). A fuzzy particle swarm approach to multiobjective quadratic assignment problems. *Second Asia International Conference on Modelling & Simulation*, 516-521.

Zitzler, E., Laumanns, M. ve Thiele, L. (2002). SPEA2: Improving the strength pareto evolutionary algorithm. *Evolutionary Methods for Design Optimization and Control*, 95-100.

EK 1. MELEZ ALGORİTMA VBA KODU

```
 'problem parametreleri
Dim n As Integer ' problem boyut Tesis sayisi
Dim m As Integer ' amac sayisi
Dim dist() As Long
Dim flow() As Long
Dim cozumler() As Long

'NSGA parametreleri
Dim Nn() As Integer
Dim Fn() As Integer
Dim sp() As Integer

'GA parametreleri
Dim mo As Single
Dim co As Single
Dim ts As Integer
Dim pb As Integer

Type Tbirey
    no As Integer
    x() As Integer 'qap icin bir cozum, permutasyon
    F() As Long ' amac fonksiyonların degerleri
    s() As Integer
    n As Integer
    rank As Integer 'NSGA- rank
    distance As Double ' crowded distance
End Type
Dim pop() As Tbirey
Dim gpop() As Tbirey
Dim rt() As Tbirey
Sub GAParametreleri()
    mo = 0.01
    co = 0.5
    ts = 100
    pb = 500
End Sub
```

```
Sub baslangicToplulugu()

    ReDim pop(pb)
    For i = 1 To pb
        ReDim pop(i).x(n)
        Call permutasyon(pop(i).x())
        ReDim pop(i).F(m)
    Next

        For j = 1 To pb - 100
        Call amacHesapla(pop(j))
        Next



        For g = 491 To 500
            For k = 1 To 2
             pop(g).F(k) = Cells(g, 78 + k)
            Next
        Next

End Sub
Sub problemOku()
    Dim parametreler As String
    parametreler = Cells(1, 1)
    p = Split(parametreler)
    n = Val(p(2))
    m = Val(p(5))

'uzaklik matrisi okunuyor
    ReDim dist(n, n)
    Cells(2, 1).Activate
    For i = 1 To n
        For j = 1 To n
            dist(i, j) = ActiveCell(i, j)
        Next
    Next

'akis matrisi okunuyor
```

```
    ReDim flow(m, n, n)
    For k = 1 To m
        Cells(k * (n + 1) + 2, 1).Activate
        For i = 1 To n
            For j = 1 To n
                flow(k, i, j) = ActiveCell(i, j)
            Next
        Next
    Next
End Sub

Sub permutasyon(ByRef s() As Integer)
    For i = 1 To n
        s(i) = i
    Next

    For i = n To 2 Step -1
        yer = 1 + Int(Rnd * i)
        gecici = s(i)
        s(i) = s(yer)
        s(yer) = gecici
    Next
End Sub

Sub amacHesapla(s As Tbirey)

    For k = 1 To m

        s.F(k) = 0
        For i = 1 To n - 1
            For j = i + 1 To n
                s.F(k) = s.F(k) + dist(i, j) * flow(k, s.x(i), s.x(j))
            Next

        Next


    Next
```

```
End Sub

Sub pareto(pop() As Tbirey)
    Dim pb As Integer
    pb = UBound(pop)

    For i = 1 To pb
        ReDim pop(i).s(0)
    Next

    Dim sp() As Integer
    Dim q As Integer
    Dim p As Integer

    ReDim Fn(0)
    For p = 1 To pb
    With pop(p)
        .n = 0
        For q = 1 To pb
            If p <> q Then
                If baskinmi(p, q, pop) Then
                    Call ekle(.s(), q)
                ElseIf baskinmi(q, p, pop) Then
                    .n = .n + 1
                End If
            End If
        Next
        If .n = 0 Then
            .rank = 1
            Call ekle(Fn(), p)
        End If
    End With
    Next

    Call crowdingDistance(Fn, pop)
    'Cells(20, 1).Activate
    'Call yazdır
```

```
    ii = 1
    While UBound(Fn) > 0
        Dim QS() As Integer
        ReDim QS(0)
        For j = 1 To UBound(Fn)
            p = Fn(j)
            For k = 1 To UBound(pop(p).s)
                q = pop(p).s(k)
                pop(q).n = pop(q).n - 1
                If pop(q).n = 0 Then
                    pop(q).rank = ii + 1
                    Call ekle(QS, q)
                End If
            Next
        Next
        ii = ii + 1
        Fn = QS
        Call crowdingDistance(Fn, pop)
    '    Cells(20 + 20 * (ii - 1), 1).Activate
    '    Call yazdır
    Wend
    'Call hepsini_yazdir

End Sub


Sub yazdır()
    For i = 1 To pbs
        For j = 1 To amacs
            ActiveCell(i, j) = degerler(i, j)
            ActiveCell(i, 4) = bireyler(i).n
            ActiveCell(i, 5) = bireyler(i).rank
            For k = 1 To UBound(bireyler(i).s)
                ActiveCell(i, 5 + k) = bireyler(i).s(k)
            Next
        Next
    Next

    ActiveCell(pbs + 2, 1) = "F(i)"
```

```
    ActiveCell(pbs + 3, 1) = "dist."
    For i = 1 To UBound(F)
        ActiveCell(pbs + 2, i + 1) = F(i)
        ActiveCell(pbs + 3, i + 1) = bireyler(F(i)).distance
    Next
End Sub

Sub hepsini_yazdir()
    For i = 1 To pbs
        Cells(i, 1) = i
        For j = 1 To amacs
            Cells(i, j + 1) = degerler(i, j)
        Next
        Cells(i, amacs + 3) = bireyler(i).rank
        Cells(i, amacs + 4) = bireyler(i).distance
    Next
End Sub

Function baskinmi(a, b, pop() As Tbirey) As Boolean
    baskinmi = True
    For i = 1 To m
        If pop(a).F(i) > pop(b).F(i) Then
            baskinmi = False
            Exit For
        End If
    Next
End Function

Sub ekle(d() As Integer, a As Integer)
    Dim uzunluk As Integer
    uzunluk = UBound(d)
    uzunluk = uzunluk + 1
    ReDim Preserve d(uzunluk)
    d(uzunluk) = a
End Sub

Sub crowdingDistance(Fn() As Integer, pop() As Tbirey)
    l = UBound(Fn)
```

```
If l < 3 Then
   For i = 1 To l
      pop(Fn(i)).distance = 1000000
   Next
   Exit Sub
End If
Dim objs()
Dim Ikumesi()
ReDim objs(l, m)
ReDim Ikumesi(l, m)
' amac degleri objs dizisine aktariliyor.. Daha sonra burada sirlanacaklar..
For i = 1 To l
   For j = 1 To m
      Ikumesi(i, j) = Fn(i) ' daha sonra amaclara gore siralanacak
      objs(i, j) = pop(Fn(i)).F(j)
   Next
Next

For i = 1 To l
   pop(Fn(i)).distance = 0
Next



For j = 1 To m
   For i = 1 To l - 1
      For k = i + 1 To l
         If objs(k, j) < objs(i, j) Then
            gecici = objs(k, j)
            objs(k, j) = objs(i, j)
            objs(i, j) = gecici

            geciciIndis = Ikumesi(k, j)
            Ikumesi(k, j) = Ikumesi(i, j)
            Ikumesi(i, j) = geciciIndis
         End If

      Next
   Next
```

```
        ilk = Ikumesi(1, j)
        son = Ikumesi(l, j)

        pop(ilk).distance = 1000000
        pop(son).distance = 1000000

        For i = 2 To l - 1
          With pop(Ikumesi(i, j))
            .distance = .distance + (objs(i + 1, j) - objs(i - 1, j)) / (objs(l, j) - objs(1, j))
          End With
        Next
    Next 'amac icin dongu

End Sub



Sub caprazlama()
    ReDim gpop(pb)
    Randomize
    Dim eb1() As Integer, eb2() As Integer

    For i = 1 To pb
        ReDim gpop(i).x(n)
        ReDim gpop(i).F(m)
       'crowding distance'a gore turnuva secimi  uygulaniyor

          'turnuvaya tabi tutulacak bireyler b11, b12, b21, b22
          Dim b11 As Integer, b12 As Integer, b21 As Integer, b22 As Integer
          'caprazlama yapilacak bireyler cb1, cb2
          Dim cb1 As Integer, cb2 As Integer

          b11 = Int(1 + Rnd() * pb): b12 = Int(1 + Rnd() * pb)
          b21 = Int(1 + Rnd() * pb): b22 = Int(1 + Rnd() * pb)

          If pop(b11).distance > pop(b12).distance Then cb1 = b11 Else cb1 = b12
          If pop(b21).distance > pop(b22).distance Then cb2 = b21 Else cb2 = b22
```

```
        'caprazlancak bireyler secildi....
        ReDim eb1(n)
        ReDim eb2(n)

        eb1 = pop(cb1).x
        eb2 = pop(cb2).x

        'sira temelli caprazlama
        For k = 1 To n
          If Rnd < 0.5 Then
            gpop(i).x(k) = eb1(k)
            For l = 1 To n
              If eb1(k) = eb2(l) Then
                eb2(l) = 0
                Exit For
              End If
            Next
          End If
        Next

        For j = 1 To n
          If eb2(j) <> 0 Then
            yer = 1
            While gpop(i).x(yer) <> 0: yer = yer + 1: Wend
            gpop(i).x(yer) = eb2(j)
          End If
        Next
        Call amacHesapla(gpop(i))
      Next

End Sub
Sub mutasyon()
    For i = 1 To pb
    With gpop(i)
      If Rnd < mo Then
        rn1 = 1 + Int(Rnd * (n - 1))
        rn2 = 2 + Int(Rnd * (n - 2))
        If rn1 > rn2 Then
```

```
            gecici = rn1: rn1 = rn2: rn2 = gecici
        End If
        aes = rn2 - rn1 - 1 'aradaki eleman sayisi
        If aes < 2 Then
            gecici = .x(rn1)
            .x(rn1) = .x(rn2)
            .x(rn2) = gecici
        Else
            For k = rn1 To rn1 + Int(aes / 2)
                gecici = .x(k)
                .x(k) = .x(rn2 - k + 1)
                .x(rn2 - k + 1) = gecici
            Next
        End If
      End If
    End With
    Call amacHesapla(gpop(i))
    Next
End Sub
Sub sirala(rt() As Tbirey)
    Dim gb As Tbirey
    For i = 1 To 2 * n - 1
      For j = i + 1 To 2 * n
        If rt(j).rank < rt(i).rank Then
            gb = rt(j)
            rt(j) = rt(i)
            rt(i) = gb
        End If
      Next
    Next

    For i = 1 To 2 * n - 1
      For j = i + 1 To 2 * n
        If (rt(i).rank = rt(j).rank) And _
        (rt(i).distance < rt(j).distance) Then
            gb = rt(j)
            rt(j) = rt(i)
            rt(i) = gb
```

```
            End If
        Next
    Next

End Sub

Sub ga()
    ReDim rt(2 * n)

    For i = 1 To ts
        'turnuva operatoru caprazlanan operatorunun icinde kullanildi
        caprazlama ' Qt populasyonu Pt'den turetilecek
        mutasyon ' Qt uzerinde mo'ya gore mutasyon yap

        ' Qt ve Pt populasyonlarini birlestirerek Rt'yi olustur
        For j = 1 To 2 * n
            If j <= n Then rt(j) = pop(j) Else rt(j) = gpop(j - n)
        Next
        ' paretoyu calistir
        Call pareto(rt)
        ' Rt'den yeni Pt'yi olustur.
        Call sirala(rt)
        For j = 1 To n
            pop(j) = rt(j)
        Next


    Next

End Sub
Sub AnaProgram()
    Call problemOku
    Call GAParametreleri
    Call baslangicToplulugu
    Call pareto(pop)
    Call ga

    For sayac = 1 To pb
```

```
        For i = 1 To n
            Cells(4 + sayac, 33 + i) = pop(sayac).x(i)
        Next

        For k = 1 To m
            Cells(4 + sayac, 55 + k) = pop(sayac).F(k)
        Next
        Cells(4 + sayac, 57 + k) = pop(sayac).rank
        Cells(4 + sayac, 58 + k) = Round(pop(sayac).distance, 2)

    Next

End Sub
```