Journal of the Faculty of Engineering and Architecture of Gazi University 34:4 (2019) 1987-2005

Mühendislik Mimarlık Fakültesi Dergisi
Journal of The Faculty of Engineering and Architecture of Gazi University
Elektronik / Online ISSN : 1304 - 4915
Basılı / Printed ISSN : 1300 - 1884

An interactive approach with filtering to find a preferred solution of a decision maker

Ceren Tuncer Şakar[1*], Gülşah Karakaya[2]

[1]Department of Industrial Engineering, Hacettepe University, Ankara, 06800, Turkey
[2]Department of Business Administration, Middle East Technical University, Ankara, 06800, Turkey

Highlights:

- An interactive algorithm to find the best solution of a decision maker
- Designed for multiple criteria problems in discrete solution space
- Good convergence performance and speed
- Application to university selection problem with five criteria

Keywords:

- Filtering
- Interactive Approach
- Multiple Criteria Decision Making
- L_α function

Article Info:

Research Article
Received: 06.04.2018
Accepted: 15.12.2018

DOI:

10.17341/gazimmfd.571652

Correspondence:

Author: Ceren Tuncer Şakar
e-mail: cerents@hacettepe.edu.tr
phone: +90 312 297 8950

Graphical/Tabular Abstract

Many decision making problems involve multiple criteria that need to be considered simultaneously, and these criteria generally conflict with each other. It is a difficult process for a decision maker (DM) to reach a final solution in such problems. In this study, a new interactive approach is proposed to converge towards the most preferred solution of the DM among discrete alternatives that are evaluated with multiple criteria.

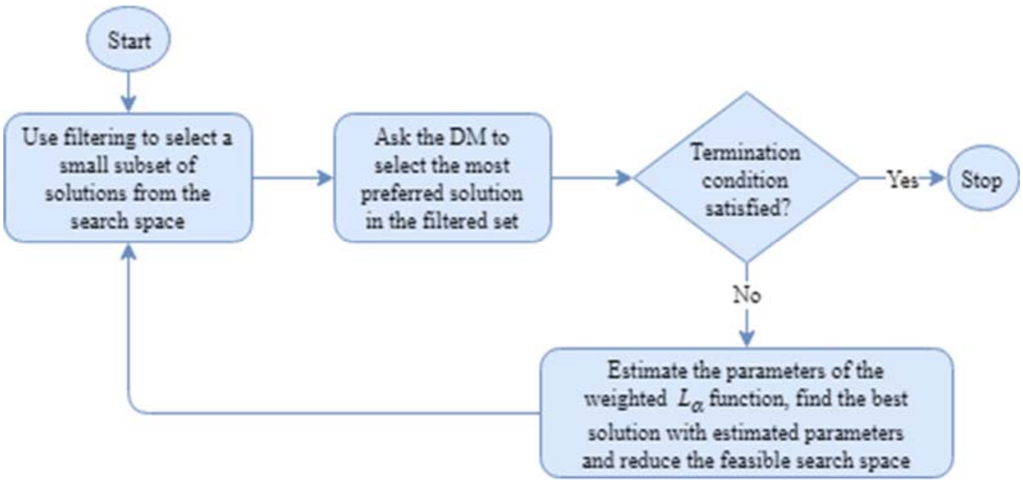


Figure A. Main steps of the proposed approach

Purpose: For multiple criteria choice problems, this study proposes an approach that distinguishes a highly preferred solution of the DM with low cognitive burden.

Theory and Methods:

At each iteration of the proposed interactive approach, the DM evaluates a small number of solutions and selects the most preferred one. The preferences of the DM are modeled by a weighted L_α function and parameters of this function (weight values and α) are estimated iteratively. Weighted L_α function is flexible and capable of representing a wide variety of preference structures with the use of different parameters. The solutions presented to the DM are determined with a filtering procedure to enhance convergence performance.

Results:

The proposed approach is tested with different sets of university data from the rankings of Times Higher Education. The results show that in various problem settings, the approach successfully converges to the most preferred solution of the DM with low number of iterations. The approach is also shown to outperform benchmark algorithms.

Conclusion:

Various forms of DM preferences can be represented through a weighted L_α preference function, and the proposed approach is able to find a highly preferred solution of the DM for different forms. DM is required to perform a simple task in the approach and the number of iterations is low.

Journal of the Faculty of Engineering and Architecture of Gazi University 34:4 (2019) 1987-2005

Mühendislik Mimarlık Fakültesi Dergisi
Journal of The Faculty of Engineering and Architecture of Gazi University
Elektronik / Online ISSN : 1304 - 4915
Basılı / Printed ISSN : 1300 - 1884

Karar vericinin tercih ettiği çözümü bulmak için filtreleme yardımlı etkileşimli bir yöntem

Ceren Tuncer Şakar[1]*, Gülşah Karakaya[2]

[1]Hacettepe Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği Bölümü, Ankara, 06800, Türkiye
[2]Orta Doğu Teknik Üniversitesi, İktisadi ve İdari Bilimler Fakültesi, İşletme Bölümü, Ankara, 06800, Türkiye

Ö N E Ç I K A N L A R

- Çok kriterli problemlerde L_α tercih fonksiyonları
- Filtreleme tabanlı, karar verici ile etkileşimli yöntem
- Test problemleri ile analizler

Makale Bilgileri
Araştırma Makalesi
Geliş: 06.04.2018
Kabul: 15.12.2018

DOI:
10.17341/gazimmfd.571652

Anahtar Kelimeler:
Filtreleme,
etkileşimli yöntem,
çok kriterli karar verme,
L_α fonksiyonu

ÖZET

Çoklu ve birbirleriyle çelişen amaçlara sahip karar verme problemlerinde karar vericinin en çok tercih ettiği çözümü belirlemek kolay ve doğrudan bir iş değildir. Bu çalışmada, mevcut çözümler arasında karar vericinin en çok tercih ettiğine yakınsamak üzerine etkileşimli bir yöntem önerilmiştir. Önerilen yöntem, ardışık iterasyonlar boyunca karar vericiden tercih bilgisi toplamakta ve bu bilgiler doğrultusunda çözüm uzayını daraltmaktadır. Karar verici, her iterasyonda sınırlı sayıda çözümü değerlendirmekte ve bilişsel yükü düşük kararlar vermektedir. Bu kararlar, gerçek karar verici davranışlarına uygun ve esnek bir tercih fonksiyonu olan ağırlıklı L_α fonksiyonu ile modellenmiştir. Önerilen yöntemin etkinliğini artırmak için her iterasyonda değerlendirme için sunulan çözümler bir filtreleme yöntemi yardımı ile belirlenmiştir. Yöntemin performansını değerlendirmek için Times Higher Education'ın yıllık olarak yaptığı üniversite sıralama verileri kullanılmıştır. Beş kriterde değerlendirilen üç farklı üniversite kümesi ile deneyler yapılmıştır. Sonuçlar, önerilen yöntemin en çok tercih edilen çözüme yakınsamakta başarılı olduğunu göstermektedir.

An interactive approach with filtering to find a preferred solution of a decision maker

H I G H L I G H T S

- L_α preference functions in multiple criteria decision making problems
- Filtering based method interacting with decision maker
- Analysis with test problems

Article Info
Research Article
Received: 06.04.2018
Accepted: 15.12.2018

DOI:
10.17341/gazimmfd.571652

Keywords:
Filtering,
interactive approach,
multiple criteria decision making,
L_α function

ABSTRACT

In decision making problems that contain multiple and conflicting objectives, it is not an easy and straightforward task to determine the most preferred solution of a decision maker. In this study, an interactive method is proposed to converge to the most preferred solution of the decision maker. The proposed method elicits preference information from the decision maker in consecutive iterations and reduces the solution space accordingly. The decision maker evaluates a limited number of solutions in each iteration and makes decisions that impose low cognitive burden. These decisions are modeled by a weighted L_α function, a preference function that is flexible and compatible with behavior of real decision makers. To enhance the efficiency of the proposed method, the solutions presented for evaluation in each iteration are determined with the help of a filtering approach. Data from annual university rankings of Times Higher Education is used to assess the performance of the proposed method. Experiments are carried out with three different university sets evaluated with respect to five criteria. The results show that the proposed method successfully converges to the most preferred solution.

*Sorumlu Yazar/Corresponding Author: cerents@hacettepe.edu.tr, kgulsah@metu.edu.tr / Tel: +90 312 297 8950 / 132

1. GİRİŞ (INTRODUCTION)

Gerçek problemlerin birçoğu, karar verme aşamasında değerlendirilecek birden fazla kriter içerir. Birçok organizasyonu ilgilendiren bu problemler için sıklıkla Çok Kriterli Karar Verme (ÇKKV) yöntemleri kullanılır. Örneğin, Özel ve Özyörük [1], Dağdeviren ve Erarslan [2] ve Akyüz [3] çok kriterli tedarikçi seçimi problemini ele almışlardır. Yurdakul ve İpek [4] üretim ve taşıma sistemlerinin belirlenmesi, Aksoy vd. [5] performans değerlendirme, Temiz ve Erol [6] ise çizelgeleme problemine çok kriterli yaklaşımlar uygulamışlardır. ÇKKV problemlerinde göz önüne alınan kriterler çoğu zaman birbirleriyle çelişir ve bunun sonucunda da problem için tek bir en iyi çözüm bulmak imkânsız hale gelir. Bu problemlerde en iyi çözüm kavramı yerine etkin (efficient) çözümler kümesi kavramı ortaya çıkar. Etkin çözümler ÇKKV problemlerinde aralarından seçim yapılabilecek mantıklı çözümleri ifade eder. Bir etkin çözümde bir kriter iyi bir değere sahipken başka bir etkin çözümde başka bir kriter iyi değere sahip olabilir. Etkin çözümler kümesinde, bir kriter değerini iyileştirmek mutlaka en az bir diğer kriter değerinin kötüleşmesiyle sonuçlanır. ÇKKV problemlerinde, farklı Karar Vericiler (KV'ler) farklı etkin çözümleri en iyi çözümler olarak belirleyebilirler çünkü farklı KV'ler aynı problemdeki kriterler için farklı tercihlere sahip olabilirler. Teorik olarak, bir problemdeki etkin çözümlerin her birisini diğerlerinden fazla tercih edecek bir KV olabilir.

ÇKKV literatüründeki birçok çalışma, KV'lerin alternatifler arasındaki tercihlerini tercih fonksiyonları yoluyla ifade edilebileceğini varsaymaktadır. Bu fonksiyonlara bazı çalışmalarda değer veya fayda fonksiyonu da denilmektedir. Bu fonksiyonlar, karar alternatiflerinin göz önüne alınan kriterlerdeki değerlendirmelerini kullanarak bu değerleri toplam genel bir ölçüte çevirirler. Bu ölçütler alternatiflerin KV için tercih edilebilirliğini gösterir ve alternatifler arasındaki kıyaslamalarda kullanılır. Roy [7] tarafından belirtildiği gibi, çok kriterli analizlerde ilgilenilen karar verme problemleri üç tip olarak gruplanabilir: (i) en iyi alternatifin veya az sayıdaki en iyi alternatiflerin arandığı *seçim problemi*, (ii) alternatiflerin gruplandığı ve grupların tercih sırasına sokulduğu *sıralı gruplama problemi* ve (iii) tüm alternatiflerin tercih sırasına sokulduğu *sıralama problemleri*. Tercih fonksiyonlarıyla elde edilen ölçütler bahsedilen üç problem tipinde de karar verme için kullanılabilir.

Bu makale, seçim yapılan çok kriterli karar analizi problemlerine odaklanmaktadır. Geliştiren yöntem, çok kriterli bir problemdeki alternatiflerden KV'nin en çok tercih ettiğine yakınsamak üzerine kurulmuştur. KV tercihleri bir L_α^w fonksiyonu ile modellenmiştir. KV'den bilgi elde etmek için kullanılan yaklaşım, ardışık iterasyonlar boyunca KV'nin kendisine sunulan alternatifler arasından en çok tercih ettiğini seçmesidir. KV'nin bilişsel yükünü azaltmak amacıyla her seferinde KV'ye sınırlı sayıda alternatif sunulmakta ve alternatifler arasında kayıtsız kalınması durumuna izin verilmektedir. Algoritmanın bitiminde KV'ye kendisi için değeri yüksek son bir alternatif sağlanmaktadır. Geliştirilen yöntem 5 kriterli bir üniversite seçimi problemi üzerinde test edilmiştir. KV, 100 adet etkin üniversite içeren 3 veri setinde değişik tercih fonksiyonu varsayımları için son çözüme yönlendirilmiştir. Makale şu şekilde organize edilmiştir: Bölüm 2, literatürde KV tercihlerinin modellenmesi ve toplanması ile ilgili yapılan çalışmaların bir taramasını içermektedir. Bölüm 3, kullanılan önemli kavramların tanımları ve açıklamaları ile başlamaktadır. Daha sonra geliştirilen yöntem anlatılmaktadır. Bölüm 4'te öncelikle uygulamaların yapıldığı vaka çalışması ve verilerin nasıl elde edildiği anlatılmaktadır. Ardından önerilen algoritmanın performansını değerlendirmek için kullanılan karşılaştırma algoritmaları, deneylerde kullanılan parametrelerin detayları ve algoritmanın performansının test edileceği ölçütler verilmektedir. Algoritmaların vaka çalışması üzerinden karşılaştırmaları ve istatistiksel analizler de bu bölümde verilmektedir. Çalışma Bölüm 5'te sonuçlar ve gelecek çalışmalarla sonlandırılmaktadır.

2. LİTERATÜR TARAMASI (LITERATURE REVIEW)

KV'lerin tercihlerini gösteren fonksiyonlar doğrusal (linear), toplamsal (additive), içbükeyimsi (quasiconcave) ve genel tekdüze (general monotone) gibi değişik formlarda modellenebilir. Toplamsal tercih fonksiyonu klasik bir formdur ve yaygın olarak kullanılmaktadır. Toplamsal tercih fonksiyonlarında alternatiflerin ele alınan kriterlerdeki değerlendirmeleri toplanarak genel bir tercih ölçütü elde edilir. Bu fonksiyonun temelleri Keeney ve Raiffa [8] ve Wakker'ın [9] çalışmalarında incelenebilir. Jacquet-Lagrèze ve Siskos [10] toplamsal fayda fonksiyonları üzerine kurulmuş olan ve yaygın olarak çalışılan Utility Additive (UTA) yöntemini önermiştir. UTA yöntemi, KV'nin alternatifler arasındaki tercih bilgisini kullanarak KV'nin kararlarıyla uyumlu toplamsal fayda fonksiyonlarını bulmayı hedeflemektedir. Bu yöntemde KV'den tercihlerini güçlü tercih, zayıf tercih veya kayıtsızlık olarak ifade etmesi beklenmektedir. UTA yöntemi, ÇKKV'de sıralı regresyon (ordinal regression) paradigmasının başlangıcı olmuş ve arkasından gelen çalışmalarda farklı formlarda fayda fonksiyonları da değerlendirilmeye başlanmıştır. Siskos vd. [11] bu çalışmaların bir derlemesini sunmuştur. Geleneksel sıralı regresyon temelli modellerde, üzerinde çalışılan karar verme problemiyle uyumlu fayda fonksiyonları elde edildikten sonra bunlardan biri belirlenen bir kurala göre seçilir. Bu seçimi iyi tanımlanmamış ve isteğe göre yapılan bir karar olarak gören Greco vd. [12], KV'nin tercihleriyle uyumlu tüm fonksiyonları göz önüne alan ve değerlendiren gürbüz sıralı regresyon (robust ordinal regression) yöntemini önermiştir. Alternatiflerin gürbüz bir şekilde değerlendirilmesini hedefleyen başka bir yaklaşım sunan Kadzinski ve Tervonen [13] olasılıksal bir çalışma yapmıştır. KV'nin yaptığı tercihlerle uyumlu olan tüm tercih fonksiyonları göz önünde bulundurularak bir alternatifin bir başkasına tercih edilmesinin olasılığı bulunmuştur. Ayrıca, alternatiflerin verilen bir tercih sıralaması numarasına sahip olmalarının olasılıkları hesaplanmıştır.

Bazı araştırmacılar, KV'lerin kararlarının altında yatan tercih fonksiyonlarıyla ilgili yapılan bazı varsayımların çok sınırlayıcı olduğunu ve bunların yapılan tercihlerle uyumlu bir fonksiyon elde etmeyi olursuz hale getirebileceğini savunmuşlardır. Angilella vd. [14] UTA'nın fayda fonksiyonu ile ilgili yaptığı varsayımların problemi olursuz hale getirebileceğini belirtmiş ve buna karşı bir bulanık integraller (fuzzy integrals) çalışma çerçevesi önermiştir. Bu çerçevede kriterler arasındaki etkileşimi göz önüne alabilen toplamsal olmayan tercih fonksiyonlarıyla çalışmışlardır. Marichal ve Roubens [15] birbirleriyle etkileşen kriterleri ve bulanık integralleri kullanan başka bir çalışma yapmıştır. Burada kriterlerin, alternatiflerin ve ayrıca kriterlerin etkileşimlerinin kısmi sıralamaları kullanılarak kriter ağırlıkları belirlenmiştir. Benabbou vd. [16], doğrusal olmayan amaç birleştirme fonksiyonlarının tercih ortaya çıkarma sürecine esneklik kattığını savunmuştur. Yaptıkları çalışmada bu tür fonksiyonların parametrelerini belirlemek için bir mini-maks pişmanlık yaklaşımı kullanılmış ve KV ile etkileşimlerden faydalanılmıştır.

Literatürdeki diğer bazı çalışmalarda, tercih ortaya çıkarma sürecindeki belirsizliklerle ve eksik bilgiyle başa çıkılmaya çalışılmıştır. KV tercihlerini çok çaba gerektirmeyen bir yöntemle elde etmek isteyen Salo ve Hämäläinen [17], çalışmalarında KV'nin verdiği kısmi tercih ifadelerini kabul etmiş ve bunlardan bilgi elde etmiştir. Sarabando ve Dias [18] belirsiz bilginin mevcut olduğu, sadece kriter ağırlıklarının sıralamasının ve alternatiflerin kriterlerdeki değerlendirmelerinin sıralamasının bulunduğu bir durumu ele almıştır. Bu çalışma Monte-Carlo simülasyon teknikleri ile problemin parametrelerini tahmin etmiş ve daha sonra bu tahminlerle alternatiflerin bir sıralamasını bulmuştur; çalışmada çok nitelikli toplamsal bir birleştirme modeli kullanılmıştır. Ağırlıklarla ilgili kardinal değerlerin yerine sadece sıralama bilgisinin yer aldığı ÇKKV yöntemlerinin bir karşılaştırması için Ahn ve Park'ın [19] çalışması incelenebilir. Bazı durumlarda, KV bir takım alternatifler arasında kayıtsız da kalabilir, bu alternatifler arasında seçim yapamayabilir. Bu durumu göz önüne alan Branke vd. [20], gürbüz sıralı regresyonda kayıtsızlık eşikleri kullanılmasını önermiştir. Buna göre bir alternatifin bir diğerine tercih edilebilmesi için bu alternatifin genel değerinin diğerininkinden farkı en az belirlenmiş olan bir eşik kadar olmalıdır. Branke vd. [21] de kayıtsızlık eşiği kavramını kullanmış ve KV için istenen alternatifi bulmak için gereken etkileşimi azaltmak için çeşitli sezgisel yaklaşımlar önermiştir. Karakaya ve Köksalan [22] bir açık artırma probleminde interaktif bir yaklaşımla alıcı durumunda olan KV'nin tercih parametrelerini tahmin ederken modellerine kayıtsızlık olasılığını dâhil etmiştir. Burada çok aşamalı bir açık artırma ortamında KV ile interaktif bir şekilde doğrusal tercih fonksiyonlarının parametrelerini ortaya çıkarırken kayıtsızlık eşikleri kullanmıştır. Tercih modellerindeki kayıtsızlık eşiklerinin kullanımının genel bir anlatımı için Pirlot ve Vincke'ye [23] başvurulabilir. Bu kaynakta kayıtsızlık eşiklerinin gösterimleri ve modellenmesi, genetik, karar destek sistemleri, sınıflandırma, psikoloji gibi çeşitli alanlardaki örnek uygulamaları bulunmaktadır.

Literatürde gerçek KV'ler içeren karar verme problemlerinde içbükeyimsi tercih fonksiyonlarının enbüyüklenmesinin insan davranışını iyi bir şekilde temsil ettiği savunulmuştur (örneğin Silderberg ve Suen [24], Nicholson ve Snyder [25]). Enbüyükleme tipinde olan amaçlar içeren problemler için bu tercih fonksiyonları ile bir uygulama örneği Ulu ve Köksalan'ın [26] çalışmasında görülebilir. Enbüyükleme tipinde içbükeyimsi tercih fonksiyonları kullanarak en iyi çözümü bulmaya yönelik yaygın bir algoritma için Korhonen vd. [27] incelenebilir. Amaçlar enküçükleme tipinde olduğu zaman ise dışbükeyimsi tercih fonksiyonları uygun ve kullanışlı fonksiyonlar olarak öne çıkmaktadır. Bu makalede kullanılan tercih fonksiyonu bir enküçükleme fonksiyonudur ve dışbükeyimsi tercih fonksiyonlar ailesine aittir. Tüm kriterlerin en iyi değerlerinden oluşan ideal vektörden ağırlıklı L_α uzaklığı enküçükleyen bu tercih fonksiyonu ($L_\alpha^{\boldsymbol{w}}$ fonksiyonu), kriter ağırlıkları ($\boldsymbol{w}$) ve uzaklık parametresi (α) değiştirilerek esnek bir şekilde kullanılabilir. Örneğin, α değeri 1 alınarak doğrusal tercih fonksiyonları, 2 alınarak karesel tercih fonksiyonları, sonsuz alınarak Tchebycheff tercih fonksiyonları temsil edilebilir. Gerektikçe bunlar dışında ara değerler kullanılarak farklı KV davranışları modellenebilir. Literatürde $L_\alpha^{\boldsymbol{w}}$ tercih fonksiyonu Karakaya ve Köksalan'ın [28] ve Karakaya vd.'nin [29] çalışmalarındaki gibi temsili tercih fonksiyonu olarak ya da Lokman ve Köksalan'ın [30] ve Lokman vd.'nin [31] çalışmalarındaki gibi test fonksiyonu olarak kullanılmaktadır.

Bu makalede geliştirilen yöntemin uygulaması, dünya üniversitelerinin çeşitli kategorilerde yıllık olarak sıralamasını gerçekleştiren Times Higher Education (THE)'dan alınan veriler üzerinde yapılmıştır. Akademik programların sıralanması yaygın olan ve periyodik olarak merakla beklenen bir konudur. THE dışında US News & World Report ve Financial Times gibi yayınlar da üniversitelerin ve MBA programlarının sıralamalarını içermektedir. Bu sıralamalar yapılırken akademik programları değerlendirmede önemli olduğu düşünülen kriterler ve bu kriterlerin önem ağırlıkları baştan belirlenmektedir. Bu kriterler belirlenirken eğitim kalitesi ve olanakları, üretilen yayınların kalitesi ve sayısı, akademik kadronun ve öğrencilerin uluslararası ve cinsiyet dağılımları, mezunların işleri ve kazançları gibi birçok faktör göz önüne alınmaktadır. Bickerstaffe ve Ridgers [32] ve Peters [33] yapılan sıralamaların etkin değerlendirme yöntemleri olduğunu ve akademik programların tercih edilebilirliğini yüksek oranda etkilediğini savunmuştur. Keeney vd. [34] ve Liu ve Cheng [35] gibi başka araştırmacılar ise sıralama için belirlenen kriterleri ve sıralama ölçütünü elde etmek için kullanılan yöntemleri eleştirmişlerdir. Üniversiteler ile diğer yüksek öğretim sistemleri için uygulanmakta olan sıralama yöntemlerinin bir karşılaştırması için Millot'un [36] derlemesi incelenebilir. Hazelkorn [37] yüksek öğretim kurumlarının sıralamalarında kullanılan ölçütlerin, sıralamaların öğrenci ve akademik çalışanların tercihleri üzerine etkilerinin ve sıralamaların politik faktörlerinin genel bir anlatımını sunmuştur. Köksalan ve Tuncer [38] ve Köksalan vd. [39] akademik programların yayınlanan kriter değerlerini kullanarak kendi sıralama yöntemlerini geliştirmişlerdir. Ancak bahsedilen çalışmaların hiçbiri bu makalede çalışılan KV tercih parametrelerini ortaya çıkarma alanında değildir. Burada yapılan çalışma, THE'nin yayınlamış olduğu dünya üniversitelerini ve bunların belirlenen kriterlerdeki değerlendirmelerini veri seti olarak kullanmaktadır. Ancak, THE'nin bu kriterlerin önem ağırlıkları için belirlediği değerleri ve birleştirme metodolojisini kullanmamaktadır.

3. TANIMLAR VE METOT
(DEFINITIONS AND METHODOLOGY)

3.1. Önemli Tanımlar (Important Definitions)

Bu bölümde sırasıyla baskınlık/efficiency kavramları, L_α^w tercih fonksiyonunun tanımı ve geliştirilen yaklaşımda kullanılan filtreleme yönteminin açıklaması verilmektedir.

ÇKKV problemlerinde değerlendirilen kriterlere iyileşme yönü eklendiğinde (enbüyükleme/enküçükleme) problemin amaçları elde edilmiş olur. Genel olarak n tane enbüyükleme amacına sahip çok amaçlı bir problem şu şekilde tanımlanabilir: (Eş. 1)

$$\textit{Enbüyükle } f(\boldsymbol{x})=\left(f_1(\boldsymbol{x}),\ldots,f_n(\boldsymbol{x})\right)^T,\ \textit{kısıtlar: } \boldsymbol{x}\in\boldsymbol{X} \qquad (1)$$

Burada $\boldsymbol{x}$ karar değişkeni vektörünü, $\boldsymbol{X}$ karar uzayındaki uygun bölgeyi ve f_j de j. amaç fonksiyonunu göstermektedir. Her karar vektörüne karşılık bir amaç vektörü ortaya çıkmaktadır. Buna bağlı olarak da $\boldsymbol{X}$ kümesinin amaç uzayındaki imajı olan $\boldsymbol{Z}$ kümesi oluşmaktadır. $\boldsymbol{Z}$ kümesindeki bir $\boldsymbol{z} = (z_1, \ldots, z_n)^T$ vektörü baskınsa buna karşılık gelen karar değişkeni vektörü de etkin çözüm olur. Baskınlık şu şekilde tanımlanabilir: $y_j \geq z_j$ koşulunu bütün $i = 1, \ldots, n$ için sağlayacak ve $y_j > z_j$ koşulunu en az bir j için sağlayacak bir $\boldsymbol{y} \in \boldsymbol{Z}$ yoktur. Diğer durumda $\boldsymbol{z}$ baskın olmayan bir vektördür. Etkin çözümler ÇKKV'de üzerinde yoğunlaşılması gereken mantıklı çözümler kümesi olduğu için bunların belirlenmesi önemlidir.

Geliştirilen yöntemde, KV'nin tercihlerini ideal vektörden ağırlıklı L_α uzaklığı enküçükleyecek şekilde yaptığı varsayılmaktadır. İdeal vektör, tüm kriterlerde en iyi değere sahip olan ütopik bir çözümdür. İdeal vektör $\boldsymbol{z}^* \in R^n$ olarak gösterilirse $\boldsymbol{z} \in R^n$ çözümünün ideal vektöre olan ağırlıklı L_α uzaklığı şu şekilde hesaplanır: (Eş. 2)

$$L_\alpha^w(|\boldsymbol{z}^*\text{-}\boldsymbol{z}|)=\begin{cases}\left(\sum_{j=1}^{n}\left(w_j|z_j^*\text{-}z_j|\right)^\alpha\right)^{1/\alpha} & \text{eğer } 1\leq\alpha<\infty\\ \underset{j=1,\ldots,n}{maks}\{w_j|z_j^*\text{-}z_j|\} & \text{eğer } \alpha=\infty\end{cases} \qquad (2)$$

Hesaplamada kullanılan ağırlık vektörü $\boldsymbol{w}$, her kriterin ağırlığı sıfırdan farklı pozitif bir değer olacak ve tüm kriterlerin ağırlıklarını toplamı 1 olacak şekilde tanımlanmıştır. Bir kriterin ağırlığı ne kadar büyükse o kriterde ideal vektöre yakınlık o kadar önem kazanır.

KV, kendisine sunulan alternatiflerden en çok tercih ettiğini seçeceği zaman ideal vektöre L_α^w uzaklığı en düşük alternatifi seçmektedir. Diğer bir deyişle, kriterlerin ağırlıklarını da düşünerek ideal vektöre en benzer olan alternatifi seçmektedir. Giriş bölümünde belirtildiği gibi, L_α^w fonksiyonu KV davranışlarını modellemeye uygun ve değişik α ve $\boldsymbol{w}$ seçimleriyle farklı tercihleri temsil edebilen esnek bir fonksiyondur.

KV'yi tercih ettiği çözüme yönlendirmek için önerilen yöntem, ardışık iterasyonlar boyunca KV'den tercih bilgisi toplamaktadır. Her iterasyonda, KV'den kendisine sunulan alternatiflerden en çok tercih ettiğini seçmesi istenmektedir. KV tercihleri hakkında daha fazla bilgi elde edebilmek ve son çözüme hızlı bir şekilde ulaşabilmek amacıyla sunulacak bu alternatiflerin belirlenmesinde bir filtreleme yöntemi kullanılmıştır. Filtreleme, büyük ve sınırlı sayıda eleman arasından daha küçük temsili kümelerin seçilmesi işlemidir. Filtreleme yapılırken elde edilen küçük kümenin mümkün olduğunca birbirinden farklı elemanlar içermesi ve orijinal kümeyi iyi temsil etmesi hedeflenir. Bu makalede kullanılan *komşuluğun dışındaki ilk nokta* filtreleme yöntemi Steuer'de [40] anlatılmaktadır. Bu yöntem aşağıda adımlar halinde açıklanmıştır. X filtreleme uygulanmak istenen noktalar kümesini, F filtre tarafından tutulan noktalar kümesini, d uzaklık parametresini, p elde edilmek istenen filtrelenmiş nokta sayısını göstermektedir.

- $|X| = k$ iken X'teki tüm noktaları $x_1, x_2, \ldots, x_k$ şeklinde bir liste halinde sırala. p değerini belirle. d için ilk değeri belirle.
- $F = \emptyset$, $t = 1$ olarak belirle ve $F \leftarrow F \cup \{x_t\}$ olarak güncelle. Adım 4'e devam et.
- x_t'nin F'deki tüm elemanlardan uzaklığını hesapla. Eğer hesaplanan tüm uzaklıklar $\geq d$ ise, $F \leftarrow F \cup \{x_t\}$ olarak güncelle.
- $t \leftarrow t+1$ yap. $t = k+1$ ise Adım 5'e devam et. Değilse Adım 3'e dön.
- Eğer $|F| = p$ ise dur. $|F| < p$ ise d değerini düşür. $|F| > p$ ise d değerini yükselt. Adım 2'ye dön.

X kümesinin Adım 1'de nasıl sıralanacağına dair bir kural yoktur, rassal bir sıralama yapılabilir. Ancak Adım 2'de belirtildiği gibi listedeki ilk noktanın her zaman filtrelenmiş elemanlar kümesine dâhil olacağı unutulmamalıdır. İstenen p değerine ulaşmak için farklı d değerleri denenebilir. Steuer [40] bu deneme-yanılma sürecini kısaltmak için d değerlerinin sistematik olarak belirlenmesi üzerine bir yaklaşım önermiştir. Bu yaklaşımda eğer filtrelenen nokta sayısı istenilenin altındaysa d belli bir değer kadar azaltılmakta, üstündeyse artırılmaktadır. Ardışık şekilde denemeler yapılmakta, denemelerde yakalanan başarıya göre bu değişim değeri iki katına çıkarılmakta veya yarıya düşürülmektedir.

3.2. Önerilen Yöntembilim (The Proposed Methodology)

Bu çalışmada, çoklu kriterlerle değerlendirilen ayrık alternatifler arasından seçim yapacak olan bir KV'yi son çözümüne yönlendirmek üzerine kurulu bir yöntem önerilmiştir. Önceki bölümde açıklandığı gibi, KV'nin tercihlerini bir $L_\alpha^{\boldsymbol{w}}$ fonksiyonuna göre yaptığı kabul edilmektedir. Yöntem, KV'ye ardışık iterasyonlar boyunca belirlenen sınırlı sayıda çözüm sunmakta ve ondan bu çözümler arasında en çok tercih ettiğini seçmesini istemektedir. KV'nin yaptığı tercih doğrultusunda her iterasyonda $L_\alpha^{\boldsymbol{w}}$ fonksiyonunun parametreleri tahmin edilmektedir. Bu tahminler sonucunda KV için tercih edilebilir olamayacak çözümler elenmekte, kalan çözümler arasından KV için tercih edilebilirliği yüksek bir çözümü de içerecek şekilde yeni bir alt küme belirlenmektedir. Bu alt küme belirlenirken KV'den daha fazla bilgi elde edebilmek ve iyi çözümlere hızlı bir şekilde yaklaşmak amacıyla bir filtreleme metodundan faydalanılmaktadır. Bu şekilde oluşturulan alt küme bir sonraki iterasyonda tercih yapması için KV'ye sunularak sürece devam edilmektedir. Algoritma durma koşulu sağlanana kadar devam etmekte ve sonunda KV'ye kendisi için tercih edilebilirliği yüksek bir çözüm önererek durmaktadır. Algoritma adımlar halinde açıklanmıştır.

Z arasından seçim yapılmak istenen ayrık baskın noktalar kümesi, k her iterasyonda KV'ye sunulacak nokta sayısı olsun. İterasyon sayacı olarak $r=1$ belirle, $n=0$ belirle.

- Filtreleme yöntemiyle Z kümesinden k-n tane nokta seç. Eğer $n=0$ ise seçilen tüm noktaları KV'ye sun, değilse filtrelenen k-1 çözüm ile birlikte mevcut en iyi çözümü KV'ye sun. r iterasyonunda KV'ye sunulan çözümler kümesi S^r olsun. KV'den S^r içindeki k noktadan en çok tercih ettiğini seçmesini iste. Seçilen çözümü "mevcut en iyi çözüm" olarak adlandır. $Z \leftarrow Z \setminus S^r$ olarak güncelle. Durma koşulu sağlandıysa Adım 4'e git, sağlanmadıysa $r \leftarrow r+1$ olarak güncelle ve Adım 2'ye devam et.
- KV'nin Adım 1'de yaptığı tüm tercihlerin bilgilerini kullanarak tercih fonksiyonunun α ve $\boldsymbol{w}$ parametrelerini tahmin et, bunlar α_r ve $\boldsymbol{w_r}$ olsun. Z kümesi içinde α_r değerine bağlı olarak oluşan baskılanan bölgede olan - tercih edilebilir olamayacak - noktaları bul, bu noktaların kümesi B^r olsun. $Z \leftarrow Z \setminus B^r$ olarak güncelle.
- Eğer $|Z| < k$ ise Z'deki noktalar ve mevcut en iyi çözümü KV'ye sun, KV'den en çok tercih ettiğini seçmesini iste ve Adım 4'e git. Değilse, tahmin edilen α_r ve $\boldsymbol{w_r}$ değerleriyle Z'deki noktalardan mevcut en iyi çözüm dışındaki en iyi tercih fonksiyonu değerine sahip olan noktayı bul. Bu noktayı Z'deki noktalar içinde en başa koyarak filtreleme listesini oluştur. n=1 olarak güncelle ve Adım 1'e git.
- Dur, mevcut en iyi çözüm KV'nin en çok tercih ettiği çözümlerden biridir.

Adım 1'de kullanılan filtreleme yöntemi sayesinde KV'ye seçim yapması için sunulan alternatifler etkili bir şekilde belirlenmektedir. Alternatif kümesini iyi temsil eden birbirinden farklı alternatifler sunmanın, rassal alternatifler sunmaya göre daha iyi tercih fonksiyonu parametresi tahminleri ile sonuçlanacağı düşünülmektedir. Bu sayede algoritmanın az sayıda iterasyon sonrasında iyi çözümlerle sonlanması hedeflenmektedir. Her iterasyonda KV'ye seçim yapması için sunulan çözümler arasında KV'nin o aşamaya kadar en iyi olarak belirlediği çözüm, mevcut en iyi çözüm, de yer almaktadır. Bu sayede algoritmanın işleyişi boyunca elde edilen en iyi çözüm muhafaza edilmektedir. KV, kendisine sunulan çözüm kümesinden en çok tercih ettiğini belirledikten sonra diğer çözümler alternatif kümesinden çıkarılmaktadır. Bu çözümlerin KV için en iyi çözüm olma ihtimali yoktur. Adım 1'in sonunda kontrol edilen durma koşulu, daha iyi anlaşılabilmesi amacıyla tüm adımların anlatılmasından sonra açıklanacaktır.

Adım 2'de $L_\alpha^{\boldsymbol{w}}$ fonksiyonunun KV'nin Adım 1'de şu ana kadar yaptığı seçimleri destekleyecek α ve $\boldsymbol{w}$ parametreleri bir matematiksel model yoluyla tahmin edilmektedir. Karakaya vd.'de [29] detayları verilen bu model aşağıda özetlenmiştir.

Parametreler

α : L_α metriğinin tahmin edilen parametresi
z_{ij} : i alternatifinin j kriterindeki değeri
z_j* : j kriterinin ideal değeri

Karar değişkenleri

ε : KV'nin tercih ettiği alternatifin tercih fonksiyonu değerinin diğer alternatiflerin değerlerinden minimum farkı
w_j : kriter j'nin ağırlığı

$$Maks\ \varepsilon \tag{3}$$

$$\left[\sum_{j=1}^{n}\left(w_j\left(z_j^*-z_{pj}\right)\right)^{\alpha}\right]^{1/\alpha} \geq \left[\sum_{j=1}^{n}\left(w_j\left(z_j^*-z_{mj}\right)\right)^{\alpha}\right]^{1/\alpha} + \varepsilon$$
$$\forall\, şu\ ana\ kadarki\ tüm\ iterasyonlarda\ çözüm\ p'ye\ tercih\ edilen\ çözüm\ m \tag{4}$$

$$w_j \geq \varepsilon \quad \forall j \tag{5}$$

$$\textstyle\sum_{j=1}^{n} w_j = 1 \tag{6}$$

$$\varepsilon \geq 0 \tag{7}$$

Her çözümün tercih değeri, ideal vektörden $L_\alpha^{\boldsymbol{w}}$ uzaklığı ile hesaplanmaktadır. Eş. 4, KV'nin şu ana kadar yaptığı tüm tercihleri modele bilgi olarak dâhil etmektedir. Her iterasyonda KV'ye sunulacak olan alternatif sayısı k iken ilk iterasyonda KV bunlardan birini tercih ettiği çözüm olarak seçtiğinde Eş. 4 k-1 kere yazılarak tercih edilen çözümün tahmin edilen tercih fonksiyonu değerinin diğer çözümlerin değerlerinden en az ε kadar büyük olması sağlanmaktadır. Algoritma ilerledikçe KV'nin yaptığı yeni tercihler de bu yolla ek kısıtlar olarak modele eklenmektedir. Amaç fonksiyonu (Eş. 3) KV tercihlerini destekleyecek en büyük ε

değerini verecek w_j değerlerini bulmayı hedeflemektedir. Eş. 5 bulunan w_j değerlerinin "merkezi" ağırlıklar olmasını sağlamaktadır. Merkezi ağırlıklar kullanmak, Köksalan'da [41] gösterildiği gibi belirlenen ağırlık vektörünün en yakın kısıttan en uzakta yer almasını ve ağırlık uzayının uç değerlere odaklanılmadan makul bir şekilde daraltılmasını sağlamaktadır. Eş. 6 w_j ağırlıklarını toplamları 1 olacak şekilde normalize etmektedir. Son olarak Eş. 7 ε değerinin negatif olmamasını garantilemektedir. Açıklanan bu model α için 1 değeri seçilerek çözülmeye başlanmakta ve olurlu bir çözüm bulunana kadar bu değer tam sayı şeklinde artırılmaktadır. Algoritma ilerledikçe ve KV yeni tercihler yaptıkça, daha önce belirlenen α değeriyle olurlu bir çözüm bulunamaması durumu oluşabilir. Bu durumda α değeri tekrar artırılmaktadır. Bu sayede KV tercihlerini destekleyecek en küçük tam sayı değeri α parametresi olarak seçilmektedir. Mümkün olduğunca küçük α değerlerinin kullanılma sebebi modelin çözüm karmaşıklığını azaltmak ve büyük α değerleriyle ortaya çıkabilen yuvarlama hatalarından kaçınmaktır. Giriş bölümünde açıklandığı gibi, $L_\alpha^{\boldsymbol{w}}$ fonksiyonu farklı α ve $\boldsymbol{w}$ değerleri ile esnek bir şekilde çeşitlendirilebilmekte ve geniş bir aralıktaki KV davranışlarını modellemek için kullanılabilmektedir. Bu sebeple, KV tercihlerini destekleyecek hiçbir α değerinin bulunamaması durumu beklenmemektedir. Ancak, bu beklenmedik durumun oluşması halinde ortaya çıkacak olan olursuzluk durumundan kurtulmak için uygulanabilecek yaklaşımlar için Chinneck'te [42] anlatılan yöntemler kullanılabilir. Bu yöntemler en az sayıda kısıtın modelden çıkarılmasıyla uygunluğun yakalanması, uygunluktan en büyük sapmanın enküçüklenmesi ve uygunluktan toplam sapmanın enküçüklenmesi şeklinde örneklenebilir. Her iterasyonda KV'nin yaptığı seçimler ve belirlenen α değeri doğrultusunda çözüm kümesindeki bazı noktalar için en çok tercih edilen çözüm olma olasılığı kalmamaktadır. KV'nin seçimi ve belirlenen α değeri kriter uzayında bir baskılanan bölge oluşturmaktadır, Karakaya vd. [29] bu bölgelerin gösterimini yapmıştır. Bu bölgedeki noktalar Adım 2'nin sonunda mevcut noktalar kümesinden elenmektedir. Bu sayede ek bilgi vermeyecek sorular KV'ye sorulmamakta ve algoritma hızlanmaktadır.

Adım 3'te seçim kümesinin eleman sayısının her iterasyonda KV'ye sunulan çözüm sayısının altına düşmesi durumunda, kümedeki tüm elemanlar ve mevcut en iyi çözüm KV'ye sunulmaktadır. KV'nin bu çözümler arasından yaptığı seçim son çözüm olmakta ve algoritma sonlanmaktadır. Eğer seçim için değerlendirilen kümedeki eleman sayısı daha yüksekse, son tahmin edilen α ve $\boldsymbol{w}$ parametreleri ile bu küme içindeki en iyi tahmini tercih fonksiyonu değerine sahip çözüm belirlenmektedir. Bu çözümün bir sonraki iterasyonda KV'ye sunulacak olan çözümler arasında yer almasının algoritmayı iyileştireceği düşünülmüştür. Bu sayede, elde edilen α ve $\boldsymbol{w}$ tahminleri KV için tercih edilebilirliği yüksek çözümleri ayırt etmek ve algoritmayı bu doğrultuda hızlandırmak amacıyla kullanılabilmektedir. Mevcut parametre tahminleriyle en iyi olan çözümün bir sonraki iterasyonda KV'ye sunulacak olan kümeye dâhil olmasını sağlamak amacıyla bu çözüm filtreleme listesinin başına konulmaktadır. Adım 1'de kontrol edilen durma koşulu için seçilebilecek farklı kurallar vardır. En basit ve doğrudan yöntem, KV'nin algoritmayı sonlandırmak istemesi ve mevcut en iyi çözümü son çözüm olarak kabul etmesidir. Bu şekilde KV kendi belirlediği ölçüde bilişsel yükle karşılaşmış olur ve son çözümün kendisi için yeterince iyi olduğunu dolaylı olarak kabul etmiş olur. Bu çalışmada deneylerde kullanılan KV varsayımsal olduğu ve bu KV için çok sayıda tercih fonksiyonu parametresi ile test yapılacağı için KV'nin isteği ile durulması uygulanabilir değildir. Kullanılabilecek diğer bir yaklaşım, önceden belirlenen maksimum iterasyon sayısına ulaşıldığında algoritmayı sonlandırmaktır. KV'nin her iterasyonda yapması gereken değerlendirme sayısını ve sürecin alacağı zamanı düşünerek iterasyon sayısı bir üst limitle sınırlanabilir. Ancak, bu limiti baştan belirlemenin öznel bir karar olduğu ve bu kararın üzerinde çalışılan probleme göre farklılaşması gerektiği düşünülmüştür. Bu çalışmadaki deneylerde kullanılan durma koşulu, algoritmanın ardışık iterasyonlarda aldığı sonuçlardaki farklılaşmanın belli bir seviyenin altına düşmesidir. Bu şekilde önemli bir fark yaratmayan iterasyonlardan kaçınılmakta ve farklı KV'ler ve test problemleri için uygulanabilecek nesnel bir kural uygulanmış olmaktadır. Uygulanan kural şu şekilde açıklanabilir: Ardışık iki iterasyonda (1) mevcut en iyi çözüm aynıysa, (2) tahmin edilen α değeri aynıysa ve (3) tahmin edilen $\boldsymbol{w}$ vektörleri içindeki ağırlıkların mutlak farklarının en büyüğü belirlenen bir eşik değerinden küçükse algoritma durdurulmaktadır. Vaka Çalışması kısmında ayrıntılı olarak sunulacağı üzere, algoritmanın durma koşulu sağlandıktan sonraki ilerleyişi de ayrıca raporlanmaktadır. Bu sayede, bahsedilen diğer durma koşullarının uygulanması durumunda sonuçların ne olacağı üzerine de bilgi elde edilebilecektir.

Adım 1'de, KV kendisine sunulan çözümlerden en çok tercih ettiğini belirteceği sırada bazı çözümlerin tercih değerleri birbirine çok yakın olabilir. Gerçek bir KV böyle bir durumda iki veya daha fazla çözüm arasında kayıtsız kalabilir. Bu durumun üstesinden gelmek için KV'nin arasında kayıtsız kaldığı çözümlerden biri rassal olarak en iyi çözüm olarak atanmaktadır. Kayıtsız kalınan diğer çözümler mevcut en iyi çözümle birlikte Z kümesinden çıkarılmakta ve mevcut en iyi çözüm bir sonraki iterasyona taşınırken bunlar algoritma sonlanana kadar farklı bir yerde tutulmaktadır. Algoritma bitiminde son α ve $\boldsymbol{w}$ tahminleriyle bu çözümlerin baskılanan bölgede yer alıp almadığı kontrol edilmektedir. Baskılanan bölgede yer alan çözümler değerlendirmeden muaf tutulurken baskılanmayan çözümler (eğer varsa) mevcut en iyi çözümle birlikte son olarak KV'ye sunulmaktadır. KV bunlar arasından en çok tercih ettiğini belirledikten sonra algoritma sonlanmaktadır.

4. VAKA ÇALIŞMASI (CASE STUDY)

4.1. Test Problemi (Test Problem)

THE dergisi, yıllık olarak çeşitli kategorilerde üniversite sıralamaları yapmaktadır ve yayınlamaktadır. Bu çalışmada kullanılan veri THE'nin 2018 yılı Dünya Üniversiteleri sıralamasından alınmıştır. Bu sıralamada beş kriterle değerlendirilen 1102 üniversite bulunmaktadır [43]. Bu kriterler *öğretim*, *uluslararası imaj*, *araştırma*, *atıflar* ve *sektör geliri* olarak belirlenmiştir. THE, kendi değerlendirmelerinde bu beş kriter için sırasıyla 0,3, 0,075, 0,3, 0,3 ve 0,025 ağırlıklarını kullanmaktadır. Her üniversitenin bu kriterlerdeki değerleri 0 ve 100 arasında verilmektedir. THE bu değerleri ve belirlenen ağırlıkları ağırlıklı toplam yöntemiyle birleştirip her üniversite için bir skor hesaplamakta ve sıralamayı da bu skorlar doğrultusunda yapmaktadır. Bizim çalışmamızda kullanılmak üzere bu 1102 üniversiteden 100'er adet etkin üniversite içeren üç küme seçilmiştir. Bu kümelerin her birinin içinde birbirini baskılamayan 100 üniversite bulunmaktadır; baskınlık testi ağırlıkları göz önüne almamaktadır, sadece kriter değerlerini kullanmaktadır. İlk problem kümesi THE listesinin üst sıralamalarından seçilmiştir, bu kümede THE sıralaması 350 ve 500 arasında olan 100 etkin üniversite bulunmaktadır. Daha üst sıralardan üniversitelerin seçilememe sebebi bunların diğer üniversiteleri baskılaması sonucu 100 adet etkin üniversite bulunamamasıdır. Bu küme makalede "Problem Kümesi 1" olarak adlandırılmaktadır. Daha sonra bu üniversiteler çıkarıldıktan sonra tekrar 100 etkin üniversite aranmış ve bu sefer THE sıralaması 500 ve 800 arasında olan farklı 100 üniversite bulunmuştur, bu kümeye "Problem Kümesi 2" denilmektedir. Son olarak da "Problem Kümesi 3" THE sıralaması 800 ile 1000 arasında olan 100 etkin üniversiteyi içermektedir. Bu şekilde çözüm uzayının farklı bölgelerinde uygulamalar yapmak mümkün olmuştur.

Etkin alternatif kümeleri seçildikten sonra KV'nin tercihlerinin simüle edildiği L_α^w fonksiyonunun ağırlık parametrelerinin belirlenmesi gerekmektedir. Bunlar belirlenirken tek bir ağırlık vektörü yerine ağırlık uzayını mümkün oldukça iyi temsil edecek bir ağırlık vektörü kümesi amaçlanmıştır. Bu kümenin elde edilmesinde Steuer'de [44] küme ayrıklaştırma bölümünde açıklanan 50-50 stratejisi kullanılmıştır. Bu kaynakta gösterildiği üzere, ağırlık vektörlerini tekdüze dağılımdan seçmek ağırlıkların orta değerlerde yoğunlaşmasına neden olmaktadır. Diğer tarafta, ağırlıkları Weibull dağılımdan seçmek ise bazı ağırlıkları yüksek değerlerde, diğer ağırlıkları ise düşük değerlerde toplamaktadır. 50-50 stratejisinde önerildiği gibi ağırlıkların yarısı tekdüze dağılımdan, diğer yarısı ise Weibull dağılımdan üretilirse elde edilen ağırlıklar dengeli ve temsil yeteneği yüksek bir dağılıma sahip olmaktadır. Bu makalede önerilen yöntemde de bu yaklaşım kullanılmıştır. Ayrıca, küçük ağırlıkların gerçek bir KV için ayırt edilebilir olmayacağı ve uygulamaların gerçekçiliğini azaltacağı düşünülerek oluşturulacak ağırlıkların her kriter için 0,01'den büyük olmasına karar verilmiştir. Bu doğrultuda, öncelikle her bir kriterin ağırlığı 0 – 1 aralığında tekdüze dağılımdan rassal olarak üretilen çok sayıda ağırlık vektörü oluşturulmuştur. Oluşturulan bu vektörlerin her biri vektörün L_1 normuna bölünerek normalize edilmiş ve ağırlıkların toplamının 1 olması sağlanmıştır. Bunların arasından her elemanı 0,01'den büyük olan 500 adet alınmıştır. Daha sonra Steuer'de [44] önerildiği gibi ölçek parametresi 0,1 ve biçim parametresi 0,3 seçilerek Weibull dağılımdan yeni rassal ağırlık vektörleri üretilmiştir. Üretilen vektörler yine L_1 normlarına bölünerek normalize edilmiştir ve bunlardan her elemanı 0,01'den büyük olan 500 tanesi alınmıştır. Elde edilen toplam 1000 adet ağırlık vektörü, filtreleme yapılarak temsili bir kümenin seçilmesi için rassal bir şekilde karıştırılarak sıralanmıştır. THE tarafından kullanılan ağırlıklar da çalışmaya dâhil edilmek istendiğinden bu vektör de listenin başına eklenmiştir. Daha sonra bölüm 3.1.3'te anlatılan komşuluğun dışındaki ilk nokta filtreleme yöntemi kullanılmış ve 50 adet ağırlık vektörü seçilmiştir. Deneylerde kullanılan bu vektörler Tablo 1'de verilmiştir. Tüm makalede *öğretim*, *uluslararası imaj*, *araştırma*, *atıflar* ve *sektör geliri* kriterlerinin ağırlıkları sırasıyla w_1, w_2, w_3, w_4, w_5 olarak gösterilmektedir.

Belirlenmesi gereken diğer bir parametre de L_α^w fonksiyonun α parametresidir. Simülasyonlarda kullanılan L_α^w fonksiyonları için farklı α değerleri kullanılarak çeşitli tercih fonksiyonlarının temsil edilmesi amaçlanmıştır. Kullanılan α değerleri 1, 2, 3, 4, 5, 6 ve sonsuzdur. Böylelikle yaygın şekilde kullanılan doğrusal tercih fonksiyonları, karesel tercih fonksiyonları ve Tchebycheff tercih fonksiyonlarının yanı sıra çeşitli tercih fonksiyonları da ele alınmıştır.

Oluşturulan 3 problem kümesi için de 50 adet ağırlık vektörü ve 7 adet α değerinin tüm kombinasyonları için önerilen yöntem uygulanmıştır. Bu şekilde her problem kümesi için 350, toplamda da 1050 tane uygulama yapılmıştır. Her uygulamada yöntemin amacı o uygulama için geçerli olan parametrelere göre KV'nin en çok tercih ettiği çözüme yakınsamaktır. Her KV'nin kriterlere verdiği ağırlıklar farklı olabilir, bu ağırlıklar ve tercih fonksiyonuna göre her KV için farklı bir çözüm en iyi olacaktır.

4.2. Uygulamalar (Applications)

Önerilen algoritmanın performansını değerlendirmek için bir karşılaştırma algoritması düzenlenmiştir. Düzenlenen bu algoritmada filtreme yöntemi kullanılmayıp, KV'ye seçim yapması için verilen çözümler (mevcut en iyi dışındakiler) rassal bir şekilde seçilmektedir. Böylelikle uygulan filtreleme yönteminin KV'ye sorulan soru seçimine ve performansa etkisi araştırılacaktır. Makalenin bundan sonraki kısımlarında önerilen algoritma "Filtrelemeli Algoritma", karşılaştırma için düzenlenen algoritma da "Rassal Algoritma" olarak adlandırılacaktır. Bu iki algoritma yalnızca filtreleme yöntemi yerine rassal seçimin kullanılmasıyla ayrışmakta olup diğer tüm süreçler aynı şekilde uygulanmaktadır. Algoritmalar C++ programlama dilinde derlenip, tüm deneyler Intel(R) Core(TM) i7-6700 CPU @ 3.40 GHz, 8 GB RAM, 64-bit Microsoft Windows 10 işletim sistemli bir bilgisayarda gerçekleştirilmiştir. Matematiksel modeller GAMS 23.9 kullanılarak çözülmüştür.

Algoritmalar KV'ye her iterasyonda 5 çözüm sunulacak şekilde çalıştırılmıştır. KV'nin yaptığı seçimlere göre baskılanan bölgelerde yer alan ve değerlendirme dışı

Tablo 1. Deneylerde kullanılan ağırlık vektörleri (Weight vectors used in the experiments)

#	w_1	w_2	w_3	w_4	w_5	#	w_1	w_2	w_3	w_4	w_5
1	0,3000	0,3000	0,3000	0,0250	0,0750	26	0,3868	0,2074	0,3365	0,0142	0,0551
2	0,1189	0,1804	0,3360	0,2521	0,1126	27	0,3531	0,0174	0,0198	0,5994	0,0103
3	0,0709	0,2565	0,0950	0,3624	0,2152	28	0,2311	0,1087	0,0903	0,0516	0,5184
4	0,0989	0,7565	0,0655	0,0466	0,0325	29	0,0199	0,2738	0,3677	0,0108	0,3278
5	0,0981	0,3494	0,1792	0,0455	0,3278	30	0,0810	0,0278	0,8162	0,0244	0,0506
6	0,6089	0,0729	0,1747	0,1214	0,0221	31	0,0221	0,4081	0,1416	0,3811	0,0471
7	0,5301	0,0236	0,0136	0,2609	0,1717	32	0,3234	0,0725	0,2186	0,2572	0,1282
8	0,0157	0,1042	0,1401	0,5650	0,1749	33	0,0100	0,1978	0,2633	0,2322	0,2966
9	0,0901	0,4693	0,3319	0,0970	0,0117	34	0,0179	0,5262	0,0584	0,0492	0,3484
10	0,0445	0,0346	0,1494	0,3106	0,4610	35	0,0631	0,2125	0,5167	0,0863	0,1214
11	0,0319	0,0276	0,0221	0,0170	0,9014	36	0,2485	0,0343	0,0106	0,5214	0,1852
12	0,4836	0,1879	0,0735	0,0258	0,2292	37	0,0381	0,0136	0,2853	0,1273	0,5357
13	0,2380	0,1121	0,2863	0,1022	0,2614	38	0,3788	0,0140	0,5709	0,0260	0,0103
14	0,3603	0,0788	0,0658	0,2156	0,2795	39	0,6436	0,0132	0,0228	0,0503	0,2701
15	0,7505	0,1131	0,0558	0,0175	0,0631	40	0,0342	0,0218	0,2550	0,4043	0,2848
16	0,0202	0,0549	0,0327	0,1696	0,7227	41	0,2088	0,0147	0,5768	0,0520	0,1479
17	0,0262	0,0435	0,0298	0,7641	0,1365	42	0,0425	0,6064	0,0295	0,1819	0,1398
18	0,0153	0,0232	0,5035	0,3347	0,1234	43	0,0230	0,0191	0,5586	0,1104	0,2889
19	0,0112	0,2351	0,1371	0,1486	0,4680	44	0,1990	0,4533	0,1090	0,1330	0,1057
20	0,3237	0,0261	0,4133	0,2024	0,0345	45	0,5782	0,2373	0,0118	0,1429	0,0298
21	0,1829	0,3201	0,0475	0,1931	0,2565	46	0,0159	0,3587	0,0165	0,0422	0,5667
22	0,1093	0,0680	0,3381	0,4685	0,0161	47	0,0153	0,0359	0,2417	0,6845	0,0225
23	0,4372	0,3899	0,0954	0,0526	0,0248	48	0,4895	0,0218	0,2998	0,0227	0,1662
24	0,1209	0,1950	0,0218	0,6498	0,0125	49	0,1943	0,2724	0,1808	0,3315	0,0211
25	0,3895	0,2484	0,0326	0,2578	0,0717	50	0,0190	0,3798	0,2354	0,1928	0,1729

bırakılan çözümlerin sayısına bağlı olarak son iterasyondaki çözüm sayısı 5'ten az olabilir. Ayrıca büyük α değerlerinden kaynaklanabilecek yuvarlama hatalarından ve uygulamada yaşanabilecek olası sorunlardan kaçınmak için Karakaya vd.'nin [29] çalışmasında önerildiği gibi tahmin edilen α değerlerinin 4'ten büyük olması durumunda bu değerler sonsuz kabul edilmiştir. Ardışık iki iterasyonda mevcut en iyi çözümün ve tahmin edilen α değerinin aynı olduğu durumda tahmin edilen $\boldsymbol{w}$ vektörleri içindeki ağırlıkların mutlak farklarının en büyüğü 10^{-3}'ten küçükse algoritma durmaktadır. Ayrıca, KV'nin çözümler arasında kayıtsız kalabileceği eşik değeri 10^{-8} olarak belirlenmiştir. Son olarak filtreleme yönteminde Öklit uzaklığı kullanılmıştır.

Algoritmaların karşılaştırmasında kullanılacak bazı ölçütler belirlenmiştir. Bunlardan ilki, *mutlak fark*, algoritmaların önerdiği çözümlerin $L_\alpha^{\boldsymbol{w}}$ fonksiyonu (gerçek tercih fonksiyonu) değerlerinin gerçek en iyi çözümün $L_\alpha^{\boldsymbol{w}}$ fonksiyonu değerinden mutlak sapmasıdır. Bununla birlikte, önerilen ve gerçek en iyi çözümlerin $L_\alpha^{\boldsymbol{w}}$ fonksiyonu değerlerinin farkı, gerçek en kötü ve gerçek en iyi çözümlerin $L_\alpha^{\boldsymbol{w}}$ fonksiyonu değerlerinin farkına oranlanarak da bir kıyaslama yapılmıştır ve bu ölçüt *oransal fark* olarak adlandırılmaktadır. Diğer bir ölçüt olarak da, *en iyiyi bulma sayısı*, yani çözülen problem örnekleri arasında algoritmaların gerçek en iyiyi önerme sayıları kullanılmaktadır.

4.3. Tartışmalar (Discussions)

Filtrelemeli ve Rassal algoritmaların mutlak fark, oransal fark ve en iyiyi bulma sayısı ölçütlerindeki performansları Tablo 2'de sunulmaktadır. 4. ve 5. sütunlarda her α değeri ve her problem kümesi için çözülen 50 farklı problem örneği için ortalama değerler yer alırken, son sütunda ise 50 problem örneğinden kaçında uygulanan durma koşulu ile en iyi çözümün bulunduğu raporlanmaktadır. Ortalama mutlak fark ve ortalama oransal fark değerlerine bakıldığında Filtrelemeli Algoritma'nın Rassal Algoritma'ya göre daha iyi performans gösterdiği gözlenmektedir. Ayrıca, özellikle Filtremeli Algoritma için bu ölçütlerdeki küçük değerler, algoritmanın önerdiği çözümün tercih fonksiyonu değerinin gerçek en iyi çözümün tercih fonksiyonu değerine oldukça yakın olduğunu göstermektedir. Yine son sütundaki değerlere bakıldığında Filtrelemeli Algoritma'nın Rassal Algoritma'ya göre daha çok kez en iyiyi bulduğu görülmektedir. Algoritmaların daha detaylı bir şekilde karşılaştırılabilmesi için başka ölçütlere de bakılmıştır. Bu ölçütlerden birisi algoritmaların Bölüm 3.2'de açıklanan durma koşuluna bağlı olarak gerçekleşen iterasyon sayılarıdır. Algoritmaların 50 farklı problem örneği için iterasyon sayılarının ortalamalarına ve standart sapmalarına bakılmıştır. Ayrıca, algoritmaların durma koşulundan bağımsız olarak gerçek en iyi çözüme ulaştıkları iterasyon sayılarının ortalamaları ve standart sapmaları da

Tablo 2. Filtrelemeli ve Rassal algoritmaların genel performansları
(General performances of the Filtering-enabled and Randomized algorithms)

α	Problem kümesi	Algoritma	Ortalama mutlak fark	Ortalama oransal fark	En iyiyi bulma sayısı
1	1	Filtrelemeli	0,0038	0,0056	45
		Rassal	0,0653	0,1273	7
	2	Filtrelemeli	0,0028	0,0060	47
		Rassal	0,1400	0,2563	2
	3	Filtrelemeli	0,0000	0,0000	50
		Rassal	0,1024	0,1830	15
2	1	Filtrelemeli	0,0027	0,0089	40
		Rassal	0,0383	0,1033	5
	2	Filtrelemeli	0,0031	0,0090	44
		Rassal	0,0849	0,1958	6
	3	Filtrelemeli	0,0011	0,0066	43
		Rassal	0,0666	0,1800	5
3	1	Filtrelemeli	0,0034	0,0120	37
		Rassal	0,0310	0,0848	7
	2	Filtrelemeli	0,0016	0,0056	45
		Rassal	0,0775	0,1823	7
	3	Filtrelemeli	0,0003	0,0015	47
		Rassal	0,0516	0,1452	9
4	1	Filtrelemeli	0,0037	0,0133	37
		Rassal	0,0285	0,0786	8
	2	Filtrelemeli	0,0019	0,0060	44
		Rassal	0,0812	0,1948	6
	3	Filtrelemeli	0,0008	0,0043	45
		Rassal	0,0452	0,1243	10
5	1	Filtrelemeli	0,0043	0,0153	36
		Rassal	0,0270	0,0721	9
	2	Filtrelemeli	0,0021	0,0069	43
		Rassal	0,0817	0,1995	6
	3	Filtrelemeli	0,0011	0,0057	45
		Rassal	0,0455	0,1233	10
6	1	Filtrelemeli	0,0043	0,0157	34
		Rassal	0,0268	0,0714	9
	2	Filtrelemeli	0,0023	0,0075	40
		Rassal	0,0804	0,1919	6
	3	Filtrelemeli	0,0008	0,0032	43
		Rassal	0,0455	0,1215	10
∞	1	Filtrelemeli	0,0034	0,0121	35
		Rassal	0,0265	0,0663	12
	2	Filtrelemeli	0,0023	0,0081	36
		Rassal	0,0789	0,1837	8
	3	Filtrelemeli	0,0017	0,0071	40
		Rassal	0,0455	0,1189	14

bulunmuştur. Son olarak da algoritmalarda baskılanan bölgede yer alıp KV'ye herhangi bir bilişsel yük getirmeden değerlendirme dışı bırakılan çözüm sayılarının ortalamaları ve standart sapmaları göz önüne alınmıştır. Algoritmaların bu ölçütlerdeki performansları Tablo 3'de raporlanmaktadır. Tablo 3'de 4., 5. ve 6. sütunlara karşılık gelen ölçütler için ortalama değerler verilirken parantez içerisinde de standart sapmalar verilmektedir. Algoritmaların durma koşuluna bağlı olarak gerçekleşen ortalama iterasyon sayıları ve standart sapmalarına bakıldığında Rassal Algoritma'nın Filtrelemeli Algoritma'ya göre daha erken durduğu görülmektedir. Ancak, 5. sütuna bakıldığında Rassal Algoritma'nın gerçek en iyi çözüme ulaştığı ortalama iterasyon sayılarının durma iterasyon sayılarından büyük olduğu görülmektedir. Diğer bir deyişle, Rassal Algoritma en iyi çözüme ulaşmadan durmaktadır. Diğer yandan, Filtrelemeli Algoritma için en iyi çözüme ulaşılan ortalama iterasyon sayıları durma iterasyon sayılarından küçüktür.

Tablo 3. Filtrelemeli ve Rassal algoritmaların detaylı performansları
(Detailed performances of the Filtering-enabled and Randomized algorithms)

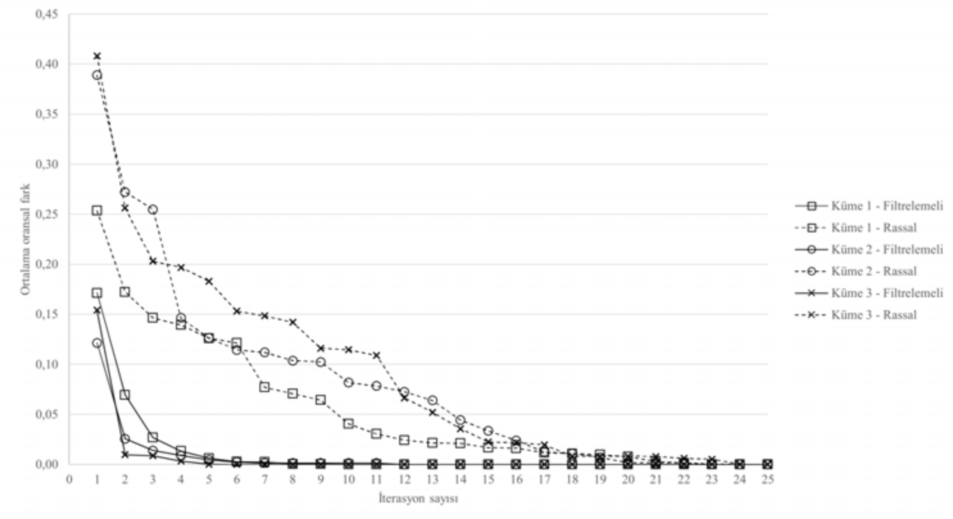
α	Problem kümesi	Algoritma	Durma iterasyon sayısı	En iyi için iterasyon sayısı	Baskılanan bölgedeki çözüm sayısı
1	1	Filtrelemeli	6,06 (1,41)	3,74 (1,71)	18,68 (12,89)
		Rassal	4,28 (1,16)	11,42 (5,79)	13,48 (7,68)
	2	Filtrelemeli	4,62 (1,03)	2,42 (1,99)	14,18 (5,52)
		Rassal	3,16 (1,18)	12,28 (6,42)	9,70 (6,54)
	3	Filtrelemeli	4,68 (1,19)	2,12 (1,15)	11,62 (7,06)
		Rassal	4,24 (1,22)	10,00 (6,33)	14,22 (10,65)
2	1	Filtrelemeli	6,08 (1,37)	4,64 (2,45)	17,20 (10,92)
		Rassal	4,08 (0,83)	11,48 (5,57)	15,04 (9,08)
	2	Filtrelemeli	4,92 (1,24)	3,36 (2,59)	13,52 (7,55)
		Rassal	3,10 (0,97)	12,52 (6,50)	13,20 (10,66)
	3	Filtrelemeli	4,96 (1,56)	3,56 (2,52)	11,86 (6,65)
		Rassal	4,44 (1,15)	12,40 (5,25)	16,24 (9,22)
3	1	Filtrelemeli	6,12 (1,24)	5,12 (2,70)	18,96 (11,42)
		Rassal	4,42 (1,54)	11,54 (5,99)	15,90 (9,68)
	2	Filtrelemeli	5,10 (1,28)	3,34 (2,41)	12,30 (7,18)
		Rassal	3,12 (1,04)	11,88 (6,81)	14,08 (12,11)
	3	Filtrelemeli	5,18 (1,55)	3,64 (2,69)	12,58 (7,20)
		Rassal	4,64 (1,16)	11,86 (5,76)	16,14 (8,42)
4	1	Filtrelemeli	6,12 (1,26)	5,10 (2,66)	19,24 (11,39)
		Rassal	4,28 (1,09)	11,24 (5,67)	18,56 (11,21)
	2	Filtrelemeli	5,26 (1,52)	3,58 (2,44)	12,52 (6,79)
		Rassal	3,00 (0,97)	12,38 (6,68)	14,92 (11,69)
	3	Filtrelemeli	5,38 (1,75)	3,94 (2,88)	14,16 (9,41)
		Rassal	4,82 (1,35)	11,12 (5,79)	17,94 (9,81)
5	1	Filtrelemeli	6,18 (1,22)	5,40 (3,49)	19,98 (12,14)
		Rassal	4,26 (0,94)	10,96 (5,50)	19,60 (11,17)
	2	Filtrelemeli	5,28 (1,54)	3,78 (2,38)	12,42 (6,70)
		Rassal	2,90 (0,84)	12,42 (6,88)	15,04 (12,20)
	3	Filtrelemeli	5,50 (1,74)	3,96 (2,83)	15,04 (10,72)
		Rassal	4,76 (1,24)	10,92 (5,56)	18,68 (10,66)
6	1	Filtrelemeli	6,14 (1,18)	5,90 (4,03)	19,90 (12,08)
		Rassal	4,26 (0,94)	10,98 (5,60)	19,50 (11,02)
	2	Filtrelemeli	5,34 (1,66)	4,14 (2,59)	13,94 (8,71)
		Rassal	2,96 (0,88)	12,34 (6,76)	17,38 (14,04)
	3	Filtrelemeli	5,58 (1,87)	4,24 (3,50)	15,66 (10,78)
		Rassal	4,82 (1,32)	11,40 (5,80)	20,90 (12,13)
∞	1	Filtrelemeli	6,12 (1,19)	5,82 (3,84)	24,28 (14,76)
		Rassal	4,52 (1,13)	10,20 (6,05)	19,46 (11,33)
	2	Filtrelemeli	5,34 (1,53)	5,20 (3,25)	18,16 (13,62)
		Rassal	3,22 (1,62)	10,56 (5,82)	24,94 (18,42)
	3	Filtrelemeli	5,80 (1,99)	4,60 (3,24)	21,66 (16,74)
		Rassal	4,96 (1,32)	9,80 (5,82)	22,40 (15,00)

Ayrıca en iyi çözüme ulaşılan ortalama iterasyon sayıları da oldukça küçüktür, Filtrelemeli Algoritma kısa sürede en iyi çözüme ulaşabilmektedir. Filtrelemeli Algoritma en iyi çözüme ulaşarak ya da en iyi çözüme yakın bir çözüm bularak durmaktadır denilebilir. Buradaki sonuçlar Tablo 2'deki ortalama mutlak fark ve ortalama oransal fark sonuçlarını da desteklemektedir. Tablo 3'deki son sütuna bakıldığında ise her iki algoritma için de baskılanan bölgedeki çözümlerin elenmesi ile KV'ye sorulan soruların önemli derecede azaldığı gözlemlenmektedir. Filtrelemeli ve Rassal algoritmaların performansları istatistiksel olarak da karşılaştırılmıştır. Bu amaçla en iyi için iterasyon sayısı, durma iterasyon sayısı ve ortalama oransal fark performans ölçütleri için t testi uygulanmıştır. Algoritmaların ölçüt değerlerindeki farkları (Filtrelemeli Algoritma ölçüt değeri - Rassal Algoritma ölçüt değeri) için bulunan %95 güven aralıkları Tablo 4'te sunulmaktadır. Her bir ölçüt için güven aralıklılarına göre daha iyi olan algoritma kazanan sütununda belirtilmiştir. Algoritmaların birbirine üstünlük sağlamadığı durumlarda ise bu sütun boş bırakılmıştır. Tablo 4'e bakıldığında en iyi için iterasyon sayısı ve ortalama oransal fark ölçütleri için Filtrelemeli Algoritma'nın Rassal Algoritma'ya göre istatistiksel olarak daha iyi performans gösterdiği görülmektedir.

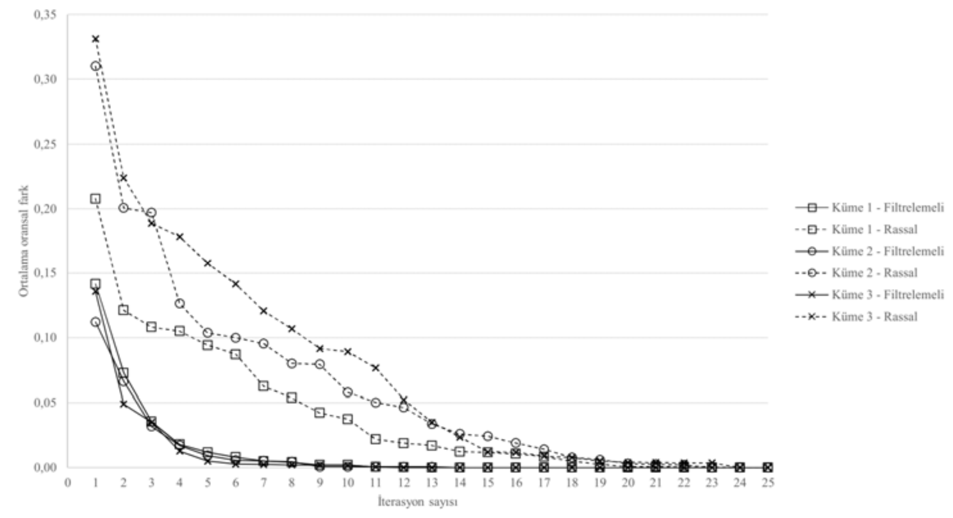
Tablo 4. Filtrelemeli ve Rassal algoritmaların istatistiksel karşılaştırması
(Statistical comparison of the Filtering-enabled and Randomized algorithms)

α	Problem kümesi	Ortalama oransal fark		Durma iterasyon sayısı		En iyi için iterasyon sayısı	
		%95 Güven Aralığı	Kazanan	%95 Güven Aralığı	Kazanan	%95 Güven Aralığı	Kazanan
1	1	(-0,1517; -0,0916)	Filtrelemeli	(1,268; 2,292)	Rassal	(-9,391; -5,969)	Filtrelemeli
	2	(-0,2949; -0,2057)	Filtrelemeli	(1,020; 1,900)	Rassal	(-11,762; -7,958)	Filtrelemeli
	3	(-0,2375; -0,1285)	Filtrelemeli	(-0,038; 0,918)		(-9,705; -6,055)	Filtrelemeli
2	1	(-0,1195; -0,0692)	Filtrelemeli	(1,550; 2,450)	Rassal	(-8,558; -5,122)	Filtrelemeli
	2	(-0,2339; -0,1397)	Filtrelemeli	(1,376; 2,264)	Rassal	(-11,136; -7,184)	Filtrelemeli
	3	(-0,2212; -0,1256)	Filtrelemeli	(-0,025; 1,065)		(-10,483; -7,197)	Filtrelemeli
3	1	(-0,0952; -0,0503)	Filtrelemeli	(1,145; 2,255)	Rassal	(-8,274; -4,566)	Filtrelemeli
	2	(-0,2217; -0,1319)	Filtrelemeli	(1,516; 2,444)	Rassal	(-10,580; -6,500)	Filtrelemeli
	3	(-0,1878; -0,0997)	Filtrelemeli	(-0,003; 1,083)		(-10,015; -6,425)	Filtrelemeli
4	1	(-0,0872; -0,0432)	Filtrelemeli	(1,373; 2,307)	Rassal	(-7,908; -4,372)	Filtrelemeli
	2	(-0,2344; -0,1433)	Filtrelemeli	(1,752; 2,768)	Rassal	(-10,810; -6,790)	Filtrelemeli
	3	(-0,1602; -0,0799)	Filtrelemeli	(-0,061; 1,181)		(-9,003; -5,357)	Filtrelemeli
5	1	(-0,0788; -0,0348)	Filtrelemeli	(1,486; 2,354)	Rassal	(-7,391; -3,729)	Filtrelemeli
	2	(-0,2383; -0,1470)	Filtrelemeli	(1,886; 2,874)	Rassal	(-10,700; -6,580)	Filtrelemeli
	3	(-0,1588; -0,0764)	Filtrelemeli	(0,140; 1,340)	Rassal	(-8,719; -5,201)	Filtrelemeli
6	1	(-0,0775; -0,0339)	Filtrelemeli	(1,456; 2,304)	Rassal	(-7,019; -3,141)	Filtrelemeli
	2	(-0,2301; -0,1386)	Filtrelemeli	(1,850; 2,910)	Rassal	(-10,250; -6,150)	Filtrelemeli
	3	(-0,1593; -0,0773)	Filtrelemeli	(0,116; 1,404)	Rassal	(-9,065; -5,255)	Filtrelemeli
∞	1	(-0,0754; -0,0331)	Filtrelemeli	(1,140; 2,060)	Rassal	(-6,400; -2,360)	Filtrelemeli
	2	(-0,2214; -0,1299)	Filtrelemeli	(1,494; 2,746)	Rassal	(-7,238; -3,482)	Filtrelemeli
	3	(-0,1522; -0,0714)	Filtrelemeli	(0,168; 1,512)	Rassal	(-7,076; -3,324)	Filtrelemeli

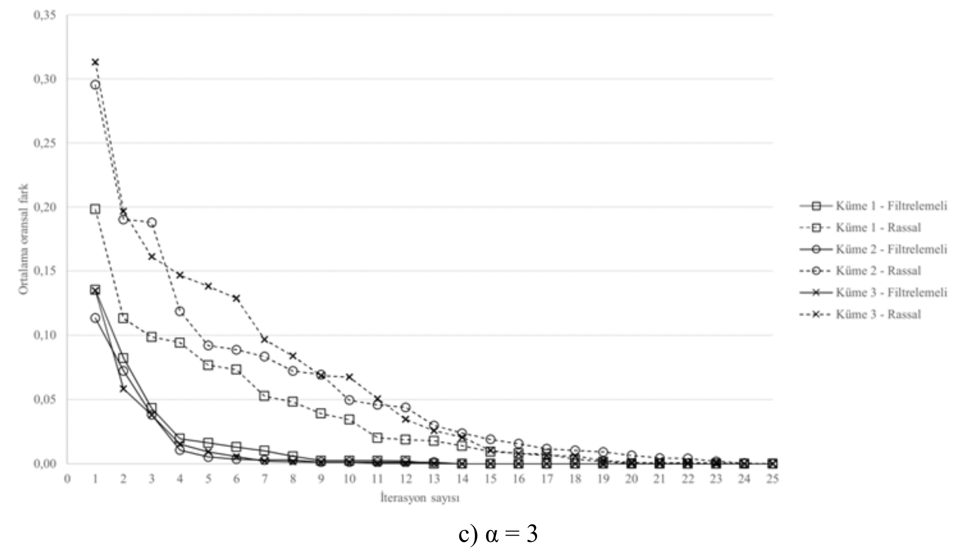
Durma iterasyon sayısı için ise çoğu problemde kazanan Rassal Algoritma olmakla birlikte, bazı problemlerde iki algoritma arasında istatistiksel olarak anlamlı bir fark görülmektedir. Sonuç olarak Filtrelemeli Algoritma'nın Rassal Algoritma'ya göre üstün olduğu söylenebilir. Filtrelemeli Algoritma için 1 (Problem Kümesi 2, ağırlık vektörü 21, α=∞) ve Rassal Algoritma için 2 (Problem Kümesi 2, ağırlık vektörü 21, α=∞ ve Problem Kümesi 3, ağırlık vektörü 26, α=∞) olmak üzere toplamda 3 problem örneğinde kayıtsızlık durumu gözlenmiştir. Bunlardan Problem Kümesi 2, ağırlık vektörü 21, α=∞ problem örneğinde her iki algoritma için de kayıtsızlık durumu gerçek en iyi ile başka bir çözüm arasında oluşurken, Problem Kümesi 3, ağırlık vektörü 26, α=∞ problem örneğinde kayıtsızlık durumu iki ara çözüm arasında oluşmuştur. Algoritmaların iterasyonlar boyunca buldukları çözümlerin en iyi çözüme yakınlığını görmek üzere Şekil 1 oluşturulmuştur.


Ortalama oransal fark
İterasyon sayısı
Küme 1 - Filtrelemeli
Küme 1 - Rassal
Küme 2 - Filtrelemeli
Küme 2 - Rassal
Küme 3 - Filtrelemeli
Küme 3 - Rassal

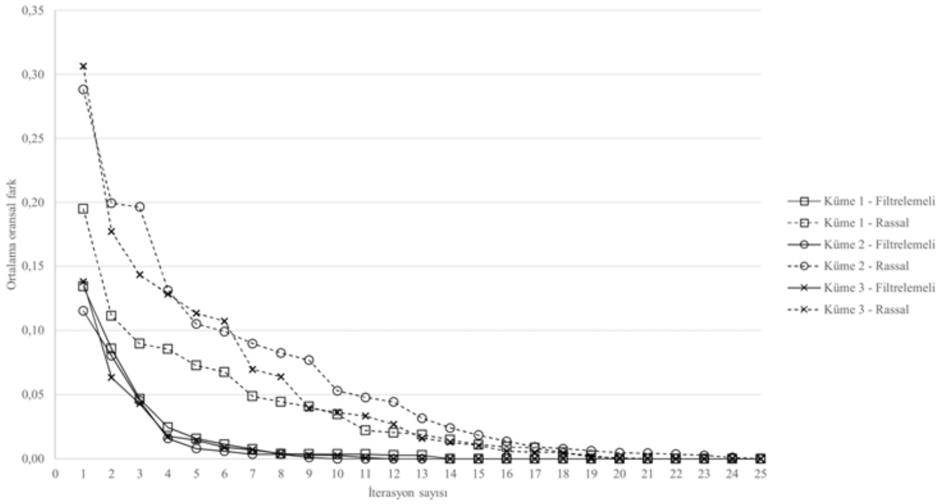
a) α = 1


Ortalama oransal fark
İterasyon sayısı
Küme 1 - Filtrelemeli
Küme 1 - Rassal
Küme 2 - Filtrelemeli
Küme 2 - Rassal
Küme 3 - Filtrelemeli
Küme 3 - Rassal

b) α = 2

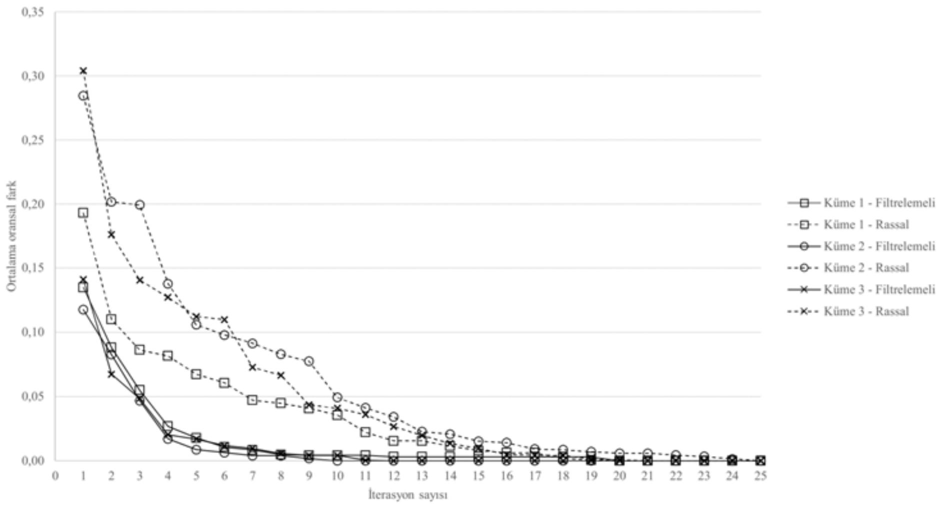

Ortalama oransal fark
İterasyon sayısı
Küme 1 - Filtrelemeli
Küme 1 - Rassal
Küme 2 - Filtrelemeli
Küme 2 - Rassal
Küme 3 - Filtrelemeli
Küme 3 - Rassal

c) α = 3



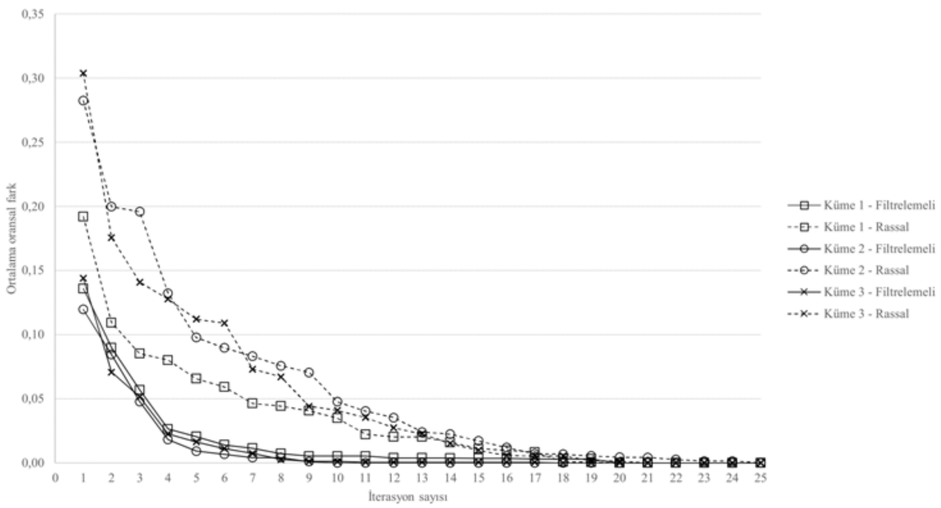
Ortalama oransal fark
İterasyon sayısı
Küme 1 - Filtrelemeli
Küme 1 - Rassal
Küme 2 - Filtrelemeli
Küme 2 - Rassal
Küme 3 - Filtrelemeli
Küme 3 - Rassal

d) α = 4



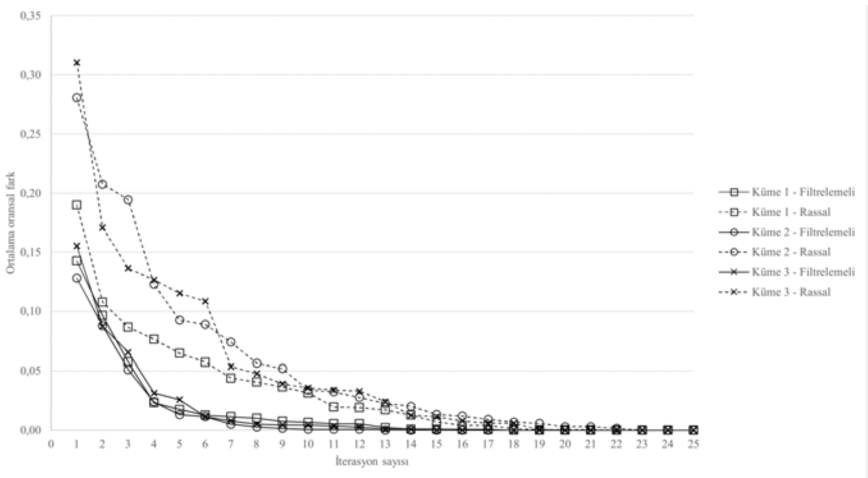
Ortalama oransal fark
İterasyon sayısı
Küme 1 - Filtrelemeli
Küme 1 - Rassal
Küme 2 - Filtrelemeli
Küme 2 - Rassal
Küme 3 - Filtrelemeli
Küme 3 - Rassal

e) α = 5



Ortalama oransal fark
İterasyon sayısı
Küme 1 - Filtrelemeli
Küme 1 - Rassal
Küme 2 - Filtrelemeli
Küme 2 - Rassal
Küme 3 - Filtrelemeli
Küme 3 - Rassal

f) α = 6



g) $\alpha = \infty$

Şekil 1. Farkli α değerleri için algoritmaların iterasyonlar boyunca ortalama oransal fark değerleri
(The average proportional difference values of the algorithms through iterations for different α values)

Bu şekilde algoritmaların iterasyonlar boyunca önerdikleri çözümlerin oransal fark değerlerinin değişimi görselleştirilmiştir. Şekil 1 incelendiğinde iterasyonlar boyunca algoritmaların önerdikleri çözümlerin tercih fonksiyonu değerlerinin en kötü ve en iyi çözümlerin fonksiyon değerlerinin farkına oranla nasıl değiştiği gözlemlenebilmektedir. Tüm α değerleri ve tüm problem kümeleri için her iterasyonda Filtrelemeli Algoritma'nın ortalama oransal fark değerinin Rassal Algoritma'ya göre oldukça düşük olduğu görülmektedir. Yukarıda da tartışıldığı gibi Filtrelemeli Algoritma'nın hızlı bir şekilde iyi çözümlere ulaştığı görülmektedir. Filtrelemeli Algoritma'nın Rassal Algoritma'ya göre üstün performans gösterdiği görüldükten sonra Filtrelemeli Algoritma literatürdeki başka bir yöntem ile de kıyaslanmıştır. Kıyaslama için seçilen yöntem Korhonen vd.'de [27] KV'nin tercih ettiği çözümü bulmak üzerine geliştirilmiş olan başka bir algoritmadır. Yöntemi geliştiren araştırmacıların isimleri (Korhonen, Wallenius ve Zionts) dolayısıyla bu algoritma KWZ Algoritması şeklinde adlandırılacaktır. Kıyaslama için KWZ Algoritması'nın seçilme sebepleri bu algoritmanın da KV'nin en çok tercih ettiği çözümü bulmaya yönelik olması, Filtrelemeli Algoritma'da olduğu gibi enbüyükleme tipinde içbükeyimsi fonksiyonlar kullanması ve literatürde yaygın olarak kabul edilen bir algoritma olmasıdır. KWZ Algoritması iterasyonlar boyunca KV'ye ikili karşılaştırmalar yaptırmaktadır, KV'nin tercihlerini doğrusal bir tercih fonksiyonuna göre yaptığı varsayılmaktadır. Her iterasyonda KV mevcut en iyi çözümü komşu çözümleriyle kıyaslamakta ve eğer daha çok tercih edilen bir komşu çözüm bulunursa mevcut en iyi çözüm güncellenmektedir. KV tercihleri sonrasında oluşan baskılanan bölgedeki çözümler Filtrelemeli Algoritma'daki gibi değerlendirmeden çıkarılmaktadır. Algoritma ilerledikçe KV tercihleri olursuzluğa yol açarsa bu durumla eski KV cevaplarının silinmesiyle başa çıkılmaktadır. KWZ Algoritması'nın sonuçlarını Filtrelemeli Algoritma ile kıyaslanabilir yapmak için KWZ Algoritması'na da benzer durma koşulları uygulanmıştır. Ayrıca KWZ ile Filtrelemeli algoritmaların KV'ye sordukları soruların farklı olması sebebiyle Filtrelemeli Algoritma'nın her iterasyonunda kaç ikili karşılaştırma yapıldığı hesaplanarak sonuçlar raporlanmıştır. KV'nin yaptığı her ikili karşılaştırma bir soru olarak sayılmıştır. Filtrelemeli Algoritma'nın uygulandığı 1050 probleme KWZ Algoritması da uygulandığında elde edilen karşılaştırmalı sonuçlar Tablo 5'te sunulmuştur. Tablo 5'te iki algoritma için de ortalama oransal fark değerlerinin oldukça düşük olduğu gözlemlenirken, algoritmaların ortalama durma soru sayıları ve standart sapmalarına bakıldığında Filtrelemeli Algoritma'nın KWZ Algoritması'na göre oldukça erken durduğu görülmektedir. En iyi için soru sayısı değerlerinin de iki algoritma için benzer olduğu görülmektedir. Ayrıca, algoritmalar Tablo 6'te istatistiksel olarak da karşılaştırılmaktadır. Algoritmaların ölçüt değerleri farkları (Filtrelemeli Algoritma ölçüt değeri - KWZ Algoritması ölçüt değeri) için bulunan %95 güven aralıkları Tablo 6'te sunulmaktadır.

Tablo 6'te ortalama oransal fark ölçütünde bazı problemlerde KWZ Algoritması kazansa da pek çok problemde algoritmaların performansları arasında istatistiksel olarak anlamlı bir fark çıkmamıştır. Bunun yanında durma soru sayısında, tüm problemler için Filtrelemeli Algoritma KWZ Algoritması'na göre üstündür. Bu ölçüt için güven aralıklarına bakıldığında, Filtrelemeli Algoritma için durma soru sayısının KWZ Algoritması'na göre oldukça küçük olduğu görülmektedir. En iyi için soru sayısı ölçütünde ise Filtrelemeli Algoritma'nın üstün olduğu bir örnek dışında, algoritmaların performansları benzerdir.

5. SİMGELER (SYMBOLS)

5.1. Kısaltmalar (*Abbreviations*)

ÇKKV : Çok Kriterli Karar Verme
KV : Karar Verici
KWZ : Korhonen, Wallenius, Zionts Algoritması
THE : Times Higher Education
UTA : Utility Additive

Tablo 5. Filtrelemeli ve KWZ algoritmalarının karşılaştırması (Comparison of Filtering-enabled and KWZ algorithms)

α	Problem kümesi	Algoritma	Ortalama oransal fark	En iyiyi bulma sayısı	Durma soru sayısı	En iyi için soru sayısı
1	1	Filtrelemeli	0,0056	45	22,24 (5,62)	12,96 (6,85)
		KWZ	0,0000	50	39,18 (10,01)	12,56 (10,32)
	2	Filtrelemeli	0,0060	47	16,48 (4,11)	7,68 (7,96)
		KWZ	0,0000	50	36,48 (7,64)	6,08 (5,68)
	3	Filtrelemeli	0,0000	50	16,72 (4,74)	6,48 (4,62)
		KWZ	0,0000	50	35,76 (9,07)	8,14 (7,93)
2	1	Filtrelemeli	0,0089	40	22,32 (5,47)	16,56 (9,79)
		KWZ	0,0000	50	43,78 (11,05)	14,48 (11,88)
	2	Filtrelemeli	0,0090	44	17,68 (4,97)	11,44 (10,34)
		KWZ	0,0000	50	36,28 (8,39)	11,22 (9,04)
	3	Filtrelemeli	0,0066	43	17,84 (6,26)	12,24 (10,07)
		KWZ	0,0002	48	38,12 (11)	13,96 (12,10)
3	1	Filtrelemeli	0,0120	37	22,48 (4,96)	18,48 (10,80)
		KWZ	0,0001	49	41,04 (10,76)	17,42 (14,03)
	2	Filtrelemeli	0,0056	45	18,40 (5,13)	11,36 (9,65)
		KWZ	0,0000	50	36,92 (8,73)	12,36 (9,60)
	3	Filtrelemeli	0,0015	47	18,72 (6,19)	12,56 (10,77)
		KWZ	0,0000	50	40,68 (11,53)	15,34 (13,94)
4	1	Filtrelemeli	0,0133	37	22,48 (5,02)	18,40 (10,64)
		KWZ	0,0002	49	42,14 (12,02)	19,84 (14,13)
	2	Filtrelemeli	0,0060	44	19,04 (6,09)	12,32 (9,77)
		KWZ	0,0000	50	38,24 (9,95)	13,84 (10,86)
	3	Filtrelemeli	0,0043	45	19,52 (6,99)	13,76 (11,53)
		KWZ	0,0012	48	42,48 (11,66)	16,3 (14,44)
5	1	Filtrelemeli	0,0153	36	22,72 (4,89)	19,6 (13,95)
		KWZ	0,0021	47	40,12 (10,69)	21,36 (15,07)
	2	Filtrelemeli	0,0069	43	19,12 (6,16)	13,12 (9,50)
		KWZ	0,0000	50	38,94 (10,29)	13,60 (11,62)
	3	Filtrelemeli	0,0057	45	20,00 (6,96)	13,84 (11,31)
		KWZ	0,0016	47	43,02 (10,35)	17,52 (14,50)
6	1	Filtrelemeli	0,0157	34	22,56 (4,71)	21,60 (16,11)
		KWZ	0,0028	46	41,02 (10,94)	22,80 (17,26)
	2	Filtrelemeli	0,0075	40	19,36 (6,64)	14,56 (10,35)
		KWZ	0,0000	49	38,60 (10,65)	16,90 (15,30)
	3	Filtrelemeli	0,0032	43	20,32 (7,50)	14,96 (13,99)
		KWZ	0,0044	45	41,22 (10,47)	18,74 (17,73)
∞	1	Filtrelemeli	0,0121	35	22,48 (4,76)	21,28 (15,37)
		KWZ	0,0057	44	41,04 (13,14)	24,64 (18,36)
	2	Filtrelemeli	0,0081	36	19,36 (6,13)	18,80 (12,98)
		KWZ	0,0006	47	37,12 (9,67)	18,28 (16,13)
	3	Filtrelemeli	0,0071	40	21,20 (7,96)	16,40 (12,96)
		KWZ	0,0068	42	42,40 (11,18)	24,88 (21,12)

Tablo 6. Filtrelemeli ve KWZ algoritmalarının istatistiksel karşılaştırması
(Statistical comparison of the Filtering-enabled and KWZ algorithms)

α	Problem kümesi	Ortalama oransal fark		Durma soru sayısı		En iyi için soru sayısı	
		%95 Güven Aralığı	Kazanan	%95 Güven Aralığı	Kazanan	%95 Güven Aralığı	Kazanan
1	1	(0,00026; 0,01094)	KWZ	(-20,17; -13,71)	Filtrelemeli	(-3,08; 3,88)	
	2	(-0,00138; 0,01343)		(-22,45; -17,55)	Filtrelemeli	(-1,15; 4,35)	
	3	-		(-21,93; -16,15)	Filtrelemeli	(-4,24; 0,92)	
2	1	(0,00256; 0,01532)	KWZ	(-24,94; -17,98)	Filtrelemeli	(-2,24; 6,40)	
	2	(0,00002; 0,01793)	KWZ	(-21,34; -15,86)	Filtrelemeli	(-3,64; 4,08)	
	3	(-0,00074; 0,01342)		(-23,84; -16,72)	Filtrelemeli	(-6,14; 2,70)	
3	1	(0,00335; 0,02046)	KWZ	(-21,90; -15,22)	Filtrelemeli	(-3,91; 6,03)	
	2	(-0,00030; 0,01143)		(-21,37; -15,67)	Filtrelemeli	(-4,82; 2,82)	
	3	(-0,00035; 0,00331)		(-25,65; -18,27)	Filtrelemeli	(-7,73; 2,17)	
4	1	(0,00355; 0,02275)	KWZ	(-23,34; -15,98)	Filtrelemeli	(-6,41; 3,53)	
	2	(-0,00016; 0,01213)		(-22,48; -15,92)	Filtrelemeli	(-5,62; 2,58)	
	3	(-0,00123; 0,00741)		(-26,79; -19,13)	Filtrelemeli	(-7,73; 2,65)	
5	1	(0,00208; 0,02435)	KWZ	(-20,72; -14,08)	Filtrelemeli	(-7,52; 4,00)	
	2	(0,00043; 0,01331)	KWZ	(-23,19; -16,45)	Filtrelemeli	(-4,69; 3,73)	
	3	(-0,00236; 0,01056)		(-26,53; -19,51)	Filtrelemeli	(-8,85; 1,49)	
6	1	(0,00217; 0,02345)	KWZ	(-21,82; -15,10)	Filtrelemeli	(-7,83; 5,43)	
	2	(0,00095; 0,01406)	KWZ	(-22,77; -15,71)	Filtrelemeli	(-7,53; 2,85)	
	3	(-0,00656; 0,00430)		(-24,52; -17,28)	Filtrelemeli	(-10,12; 2,56)	
∞	1	(-0,00418; 0,01687)		(-22,51; -14,61)	Filtrelemeli	(-10,08; 3,36)	
	2	(0,00132; 0,01362)	KWZ	(-20,98; -14,54)	Filtrelemeli	(-5,29; 6,33)	
	3	(-0,00888; 0,00935)		(-25,06; -17,34)	Filtrelemeli	(-15,45; -1,51)	Filtrelemeli

6. SONUÇLAR (CONCLUSIONS)

Gerçek hayatta sıklıkla karşılaşılan ÇKKV problemlerinde birden fazla kriterin varlığı pek çok olası çözümü de beraberinde getirir. Farklı KV'ler aynı problemdeki kriterler için farklı tercihlere sahip olabilir ve bunun sonucu olarak farklı çözümleri kendileri için en iyi olarak belirleyebilirler. Bu nedenle KV'yi tercih ettiği bir çözümü bulması konusunda desteklemek önemlidir. Bu çalışmada, ÇKKV problemleri için KV'nin en çok tercih ettiği çözüme yakınsamayı hedefleyen etkileşimli bir algoritma, Filtrelemeli Algoritma, tasarlanmıştır. Filtrelemeli Algoritma'da her iterasyonda KV'ye birtakım çözümler sunularak KV'nin tercih bilgisi toplanmaktadır. Bu bilgiler doğrultusunda ardışık iterasyonlar boyunca çözüm uzayı daraltılmakta ve sonunda iyi bir çözümün bulunması hedeflenmektedir. Algoritmada, her iterasyonda KV'ye sunulan çözümlerin çözüm uzayını iyi temsil etmesi amacıyla bir filtreleme yöntemi kullanılmaktadır. KV her iterasyonda sınırlı sayıda çözümü değerlendirmekte ve tercih bilgisi olarak sadece bu çözümler arasından en çok tercih ettiğini belirtmektedir. Bu açıdan KV bilişsel yükü düşük kararlar vermektedir. Ayrıca, KV'nin tercihleri gerçek karar verici davranışlarına uygun ve esnek bir tercih fonksiyonu olan ağırlıklı L_α fonksiyonu ile modellenmiştir.

Filtrelemeli Algoritma'nın performansını değerlendirmek için Rassal Algoritma geliştirilmiştir. Rassal Algoritma'da KV'ye sorulacak olan sorular filtreme yöntemi kullanılmadan rassal bir şekilde seçilmektedir. Böylelikle uygulanan filtreleme yönteminin soru seçimine ve performansa etkisi değerlendirilmiştir. Bu amaçla Times Higher Education'ın yıllık olarak yaptığı üniversite sıralama verileri kullanılarak beş kriterli üç farklı üniversite kümesi ile deneyler yapılmıştır. Algoritmalar farklı performans ölçütleri bağlamında değerlendirilmiştir. Ayrıca algoritmaların performansları istatistiksel olarak da karşılaştırılmıştır. Deneylerin sonuçları Filtrelemeli Algoritma'nın Rassal Algoritma'ya göre daha üstün olduğunu göstermektedir. Filtrelemeli Algoritma hızlı bir şekilde en iyi çözüme yakınsamaktadır. Ayrıca, Filtrelemeli Algoritma literatürde benzer amaçlar için geliştirilmiş olan KWZ Algoritması ile de karşılaştırılmıştır. Bulunan çözümlerin kalitesi açısından algoritmaların benzer performans gösterdiği gözlenirken, Filtrelemeli Algoritma'nın KWZ Algoritması'na göre çok daha az soru sorarak iyi çözümlere ulaştığı görülmektedir.

Gelecek çalışmalar olarak, en iyi çözüme yakınsamayı daha da hızlandıracak ek süreçler veya farklı işleyişler düşünülebilir. KV'ye sunulacak olan çözüm kümesini belirlemek için farklı sezgisel yaklaşımlar geliştirilebilir. Gerçek KV'lerin Filtrelemeli Algoritma'nın sunduğu çözümlerden ne derecede memnun kaldığı üzerine laboratuvar çalışmaları yapılabilir.

KAYNAKLAR (REFERENCES)

1. Özel B. ve Özyörük B., Supplier Selection With Fuzzy Axiomatic Design, Journal of the Faculty of Engineering and Architecture of Gazi University, 22 (3), 415-423, 2007.
2. Dağdeviren M. ve Erarslan E, Supplier Selection Using Promethee Sequencing Method, Journal Of The Faculty Of Engineering And Architecture Of Gazi University, 23 (1), 69-75, 2008.
3. Akyüz G., Bulanık Vikor Yöntemi Ile Tedarikçi Seçimi, Atatürk Üniversitesi Iktisadi Ve Idari Bilimler Dergisi, 26 (1), 197-214, 2012.
4. Yurdakul M. ve İpek A.Ö., Development of a Decision Support System to Use in the Selection of Material Handling Systems, Journal of the Faculty of Engineering and Architecture of Gazi University, 20 (2), 171-181, 2005.
5. Aksoy E, Ömürbek N, Karaatlı M, AHP Temelli Multimoora Ve Copras Yöntemi Ile Türkiye Kömür Işletmeleri'nin Performans Değerlendirmesi, Hacettepe Üniversitesi Iktisadi Ve Idari Bilimler Fakültesi Dergisi, 33 (4), 1-28, 2015.
6. Temiz I. ve Erol S., Multiobjective Genetic Algorithm for Fuzzy Flowshop Scheduling Problem, Journal of the Faculty of Engineering and Architecture of Gazi University, 22 (4), 855-862, 2007.
7. Roy B., Paradigm and challenges, Multiple Criteria Decision Analysis: State of the Art Surveys, Springer, Berlin, Almanya, 19-39, 2016.
8. Keeney R.L. ve Raiffa H., Decisions with Multiple Objectives: Preferences and Value Tradeoffs, Wiley, New York, A.B.D., 1976.
9. Wakker P.P., Additive Representations of Preferences: A New Foundation of Decision Analysis, Springer, Dordrecht, Hollanda, 1989.
10. Jacquet-Lagrèze E. ve Siskos Y., Assessing a set of additive utility functions for multicriteria decision-making, the UTA method, European Journal of Operational Research, 10 (2), 151-164, 1982.
11. Siskos Y., Grigoroudis E., Matsatsinis N.F., UTA methods, Multiple Criteria Decision Analysis: State of the Art Surveys, Springer, Berlin, Almanya, 315-362, 2016.
12. Greco S., Mousseau V., Słowinski R., Ordinal regression revisited: multiple criteria ranking using a set of additive value functions, European Journal of Operational Research, 191 (2), 416-436, 2008.
13. Kadzinski M. ve Tervonen T., Robust multi-criteria ranking with additive value models and holistic pair-wise preference statements, European Journal of Operational Research, 228 (1), 169-180, 2013.
14. Angilella S., Greco S., Lamantia F., Matarazzo B., Assessing non-additive utility for multicriteria decision aid, European Journal of Operational Research, 158 (3), 734-744, 2004.
15. Marichal J.L. ve Roubens M., Determination of weights of interacting criteria from a reference set, European Journal of Operational Research, 124 (3), 641-650, 2000.
16. Benabbou N., Gonzales C., Perny P., Viappiani P., Minimax regret approaches for preference elicitation with rank-dependent aggregators, EURO Journal on Decision Processes, 3 (1-2), 29-64, 2015.
17. Salo A.A. ve Hämäläinen R.P., Preference ratios in multiattribute evaluation (prime)-elicitation and decision procedures under incomplete information, IEEE Transactions on Systems, Man, and Cybernetics - Part A, 31 (6), 533-545, 2001.
18. Sarabando P. ve Dias L.C., Simple procedures of choice in multicriteria problems without precise information about the alternatives' values, Computers & Operations Research, 37 (12), 2239-2247, 2010.
19. Ahn B.S. ve Park K.S., Comparing methods for multiattribute decision making with ordinal weights, Computers & Operations Research, 35 (5), 1660-1670, 2008.
20. Branke J., Corrente S., Greco S., Gutjahr W.J., Using indifference information in robust ordinal regression, International Conference on Evolutionary Multi-

Criterion Optimization, Guimarães-Portekiz, 205-217, 29 Mart-1 Nisan 2015.
21. Branke J., Corrente S., Greco S., Gutjahr W., Efficient pairwise preference elicitation allowing for indifference, Computers & Operations Research, 88, 175-186, 2017.
22. Karakaya G. ve Köksalan M., An interactive approach for bi-attribute multi-item auctions, Annals of Operations Research, 245 (1-2), 97-119, 2016.
23. Pirlot M. ve Vincke P., Semiorders: Properties, Representations, Applications, Springer Science & Business Media, 2013.
24. Silberberg E. ve Suen W.C., The structure of economics: A mathematical analysis, McGraw-Hill, New York, A.B.D., 2001.
25. Nicholson W. ve Snyder C., Microeconomic theory: Basic Principles and extensions, Thomson/South-Western, Ohio, A.B.D., 2008.
26. Ulu C. ve Köksalan M., An interactive approach to multicriteria sorting for quasiconcave value functions, Naval Research Logistics, 61 (6), 447-457, 2014.
27. Korhonen P., Wallenius J., Zionts S., Solving the Discrete Multiple Criteria Problem using Convex Cones, Management Science, 30(11), 1336-1345, 1984.
28. Karakaya G. ve Köksalan M., An interactive approach for multi-attribute auctions, Decision Support Systems, 51, 299-306, 2011.
29. Karakaya G., Köksalan M., Ahipaşaoğlu S.D., Interactive algorithms for a broad underlying family of preference functions, European Journal of Operational Research, 265, 248-262, 2018.
30. Lokman B. ve Köksalan M., Finding highly preferred points for multi-objective integer programs, IIE Transactions, 46 (11), 1181-1195, 2014.
31. Lokman B., Köksalan M., Korhonen P.J., Wallenius J., An interactive algorithm to find the most preferred solution of multi-objective integer programs, Annals of Operations Research, 245 (1), pp. 67-95, 2016.
32. Bickerstaffe G. ve Ridgers B., Ranking of business schools, Journal of Management Development, 26 (1), 61-66, 2007.
33. Peters K., Business school rankings: content and context, Journal of Management Development, 26 (1), 49-53, 2007.
34. Keeney R.L., See K.E., von Winterfeldt D., Evaluating Academic Programs: With Applications to U.S. Graduate Decision Science Programs, Operations Research, 54 (5), 813-828, 2006.
35. Liu N.C. ve Cheng Y., The Academic Ranking of World Universities, Journal Higher Education in Europe, 30 (2), 127-136, 2005.
36. Millot B., International rankings: Universities vs. higher education systems, International Journal of Educational Development, 40, 156-165, 2015.
37. Hazelkorn E., Rankings and the Reshaping of Higher Education: The Battle for World-Class Excellence, Palgrave Macmillan, New York, A.B.D., 2015.
38. Köksalan M. ve Tuncer C., A DEA-Based Approach to Ranking Multi-Criteria Alternatives, International Journal of Information Technology & Decision Making, 8 (1), 29-54, 2009.
39. Köksalan M., Büyükbaşaran T., Özpeynirci Ö., Wallenius J., A Flexible Approach to Ranking with an Application to MBA Programs, European Journal of Operational Research, 201, 470-476, 2010.
40. R. Steuer R.E., Multiple Criteria Optimization: Theory, Computation, and Application, John Wiley & Sons, New York, A.B.D., 314-318, 1986.
41. Köksalan M., Multiple criteria decision making with discrete alternatives, Doktora Tezi, State University of New York, Buffalo, 1984.
42. Chinneck J.W., Feasibility and Infeasibility in Optimization, Springer, New York, A.B.D., 89-210, 2008.
43. Times Higher Education, World University Rankings 2018, https://www.timeshighereducation.com/world-university-rankings/2018/world-ranking#!/page/0/length/25/sort_by/rank/sort_order/asc/cols/stats. Yayın tarihi Eylül 2017. Erişim tarihi Mart 6, 2018.
44. Steuer R.E., Multiple Criteria Optimization: Theory, Computation, and Application, John Wiley & Sons, New York, A.B.D., 326-330, 1986.