GU J Sci, Part C, 8(1): 224-236 (2020)

Gazi Üniversitesi

Fen Bilimleri Dergisi

PART C: TASARIM VE TEKNOLOJİ

http://dergipark.gov.tr/gujsc

Approximation Algorithm Variants for Convex Multiobjective Optimization Problems

Firdevs ULUS

[1] *Bilkent University Faculty of Engineering, Department of Industrial Engineering, 06800, Çankaya/ANKARA*

Article Info:

Research article
Received: 12/09/2019
Revision: 10/02/2020
Accepted: 26/02/2020

Highlights

- *Multiobjective optimization.*
- *Algorithms.*
- *Numerical Analysis.*

Keywords

Multiobjective Optimization
Algorithms
Convex Optimization
Linear Programming

Graphical/Tabular Abstract

In this study, a recent objective space based convex multiobjective optimization algorithm (Algorithm 1), which generates inner and outer approximations to the whole Pareto frontier for any given error bound, is considered. The algorithm solves a Pascoletti-Serafini (PS) scalarization for every vertex of the current outer approximation and iterates by updating it. Different variants of this algorithm are proposed and their efficiencies are compared through numerical tests.

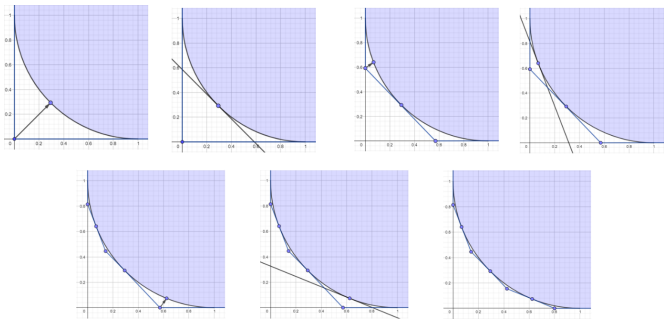


Figure A. Iterations of Algorithm 1

Purpose: Recent literature shows that Algorithm 1 works well for convex multiobjective optimization problems (MOPs) in the sense that independent of the dimension of the objective space, it returns an approximation to the whole frontier. The purpose of this study is to develop further variants to this algorithm and to compare the efficiencies.

Theory and Methods: The solution concepts for MOPs and results regarding the PS scalarization method are provided. Algorithm 1 is explained in detail. Further variants to the algorithm are proposed. Note that the variants are parametric and each $k \in \mathbb{N} \cup \{\infty\}$ describes a different variant. The variants are implemented using MATLAB. First, the performances of the variants ($k = 1,2,3,4,5,10,\infty$) are compared for randomly generated linear MOP instances; then, they are compared for a scalable (in the sense of the dimension of the objective space) nonlinear convex problem. The number of objectives is increased from 2 to 6 for each.

Results: For linear MOP instances with two and three objective functions, we observe that $k = \infty$ works the best and the performance of the algorithm improves if one decreases k as the number of the objective functions increases. For the nonlinear example however, no particular pattern is observed.

Conclusion: A convex MOP algorithm is considered and a set of variants to this algorithm is proposed. The variants are designed parametrically, one for each $k \in \mathbb{N} \cup \{\infty\}$. One recovers the original variants by setting the two extreme points ($k = 1, k = \infty$). Seven variants are implemented and tested through randomly generated linear MOPs and a scalable nonlinear convex MOP. It has been observed that the performances of the different variants differ as well. Even though no particular conclusion regarding the performances of the variants is observed for the nonlinear example, for the linear instances the algorithm works better if one decreases the value of k as the dimension of the objective space increases.

*Corresponding author, e-mail: firdevs@bilkent.edu.tr
DOI: 10.29109/gujsc.619548

GU J Sci, Part C, 8(1): 224-236 (2020)

Gazi Üniversitesi

Fen Bilimleri Dergisi

PART C: TASARIM VE TEKNOLOJİ

http://dergipark.gov.tr/gujsc

Dışbükey Çok Amaçlı Eniyileme Problemleri için Yaklaşıklama Algoritma Varyantları

Firdevs ULUS

[1]*Bilkent Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği Bölümü, 06800, Çankaya/ANKARA*

Makale Bilgisi

Araştırma makalesi
Başvuru: 12/09/2019
Düzeltme: 10/02/2020
Kabul: 26/02/2020

Anahtar Kelimeler

Çok Amaçlı Eniyileme
Algoritmalar
Dışbükey Eniyileme
Doğrusal Programlama

Öz

Bu çalışmada, dışbükey çok amaçlı eniyileme problemlerini Pareto kümeye iç ve dış yaklaşık kümeler bulmak anlamında 'çözen' bir Benson tipi algoritma ele alınmıştır. Algoritma her yinelemede o anki dış yaklaşık kümenin herhangi bir köşesi için Pascoletti-Serafini skalerizasyon modelini çözer. Çözülen bu model sayesinde köşenin Pareto kümeye yeterince yakın olup olmadığı anlaşılır. Eğer yeterince yakın değilse o anki dış yaklaşık küme bir kesit eklenerek güncellenir. Bu uygulama bir dış yaklaşık kümenin tüm köşeleri Pareto kümeye yeterince yakın oluncaya kadar tekrarlanır. Dış yaklaşık kümenin güncellemesi işlemi, Pareto kümeye yeterince yakın olmayan ilk köşe bulunduktan sonra yapılabileceği gibi tüm köşeler kontrol edildikten sonra da yapılabilmektedir. Bu seçim algoritmanın performansını etkilemektedir. Bu çalışma ile algoritmaya bu iki uç varyanta ek olarak farklı varyantlar önerilmiş ve tüm varyantların performansları bilgisayımsal testler yolu ile karşılaştırılmıştır. Rasgele türetilmiş doğrusal problemlerde, amaç uzayının boyutu arttıkça bu çalışma ile önerilen varyantların performansının daha iyi olduğu gözlenmiştir.

Keywords

Multiobjective
Optimization
Algorithms
Convex Optimization
Linear Programming

Approximation Algorithm Variants for Convex Multiobjective Optimization Problems

Abstract

We consider a Benson type algorithm to 'solve' convex multiobjective optimization problems in the sense that it generates inner and outer approximations to the Pareto frontier. In each iteration of the algorithm, a Pascoletti-Serafini scalarization is solved for an arbitrary vertex of the current outer approximation. In this way, it is possible to determine if the vertex is close enough to the Pareto frontier. If not, then the current outer approximation is updated by a cut. This procedure continues until all the vertices are close enough. The update of the outer approximation can be done right after finding the first vertex that is not close enough to the Pareto frontier; or after checking all the vertices. This choice affects the performance of the algorithm. With this study, additional variants that are different than these two extreme ones are proposed and the performances of all these variants are compared via computational tests.

1. GİRİŞ (INTRODUCTION)

Birbiri ile çelişen birden fazla amaç fonksiyonun aynı anda eniyilendiği çok amaçlı eniyileme problemlerinin birçok uygulama alanı mevcuttur (örneğin [23]) ve bu problemler literatürde çokça çalışılmıştır. Elbette çok amaçlı bir eniyileme problemini 'çözmek', tek amaç fonksiyonu olan standart bir eniyileme problemini çözmeye göre oldukça zordur. Literatürde belirli problem tipleri için geliştirilmiş çok amaçlı eniyileme problemi çözüm yaklaşımları ve algoritmalar bulunmaktadır ([8]).

Bu çalışmada, her bir amaç fonksiyonunun enküçüklendiği çok amaçlı eniyileme problemleri ele alınmıştır. Böyle bir problemde genellikle tek bir en küçük amaç fonksiyonu değeri yoktur. Bunun yerine p boyutlu amaç uzayında 'minimal' noktaların kümesini bulmak gerekmektedir. Olurlu bir noktanın amaç uzayındaki görüntüsü ancak ve ancak kendisinden 'daha küçük' başka bir olurlu nokta görüntüsü yoksa minimaldir.

Amaç uzayındaki bir minimal nokta çok amaçlı eniyileme problemlerinde başatlanmamış/baskın (İng. Nondominated) nokta olarak adlandırılmıştır. Karar uzayında, amaç fonksiyonu altındaki görüntüsü baskın nokta olan çözümlere etkin çözüm denilmektedir.

Çok amaçlı eniyileme problemlerinde genellikle bir etkin çözümler kümesi bulmak hedeflenir. Bazı problem tiplerinde bütün etkin çözümlerin kümesini bulmak da mümkündür. Örneğin, Evans ve Steuer ([9]) çok amaçlı doğrusal programlama problemlerinin tüm etkin noktalarını bulan bir simpleks algoritması geliştirmiştir. Daha sonra bu algoritmanın çeşitli iyileştirilmeleri yapılmıştır ([1, 2]). Ancak genellikle tüm etkin çözümleri bulmak zor ve bazen de gereksizdir. Örneğin, doğrusal problemlerde yozlaşma (İng. Degeneracy) olduğunda birçok etkin çözümün görüntüsü tek bir baskın noktaya denk gelebilmektedir. Aslında tek amaç fonksiyonu olan sıradan bir eniyileme probleminde de amaç tüm eniyi çözümler kümesini bulmak değil, bir eniyi çözüm bulmaktır. Benzer şekilde, çok amaçlı eniyileme problemlerinde de tüm etkin çözümleri bulmak yerine amaç uzayındaki tüm baskın noktalar kümesini ya da tüm baskın noktalar kümesini 'yeterince iyi' şekilde temsil edecek bir altkümesini verecek etkin çözümler altkümesi bulmak hedeflenir.

Bu çalışmada doğrusal ve dışbükey çok amaçlı eniyileme problemleri üzerinde durulacaktır. Yakın geçmişte bu problemler için motivasyonunu küme değerli eniyileme problemlerinden alan çözüm kavramları geliştirilmiştir. Buna göre 'sonlu (ε-) çözüm' verimli kümenin amaç uzayındaki baskın noktaların tamamını (ε-yaklaşıklığını) oluşturabilen bir altkümesidir ([15, 16]).

Doğrusal ve dışbükey çok amaçlı eniyileme problemleri için tasarlanmış dış yaklaşıklama algoritmalarının öncüsü Benson'ın 1998'de doğrusal problemler için önerdiği algoritmadır ([3]) ve yazında benzer düzende çalışan dış yaklaşıklama algoritmaları sıkça Benson tipi algoritmalar olarak anılmaktadır. Benson'ın önerdiği algoritmanın doğrusal problemler için geliştirilmiş farklı varyantları [22, 12, 4]'te görülebilir. Ayrıca, doğrusal problemler için geometrik çifteşlik teorisi geliştirilmiş ([15]) ve çifteş problemin yapısı kullanılarak farklı (çifteş) dış yaklaşıklama algoritmaları geliştirilmiştir ([6, 12]). Dışbükey problemler için Benson tipi bir dış yaklaşıklama algoritması [7]'de önerilmiştir. Daha az sayıda skalerizasyon modeli çözen benzer bir algoritma ve onun geometrik çifteş versiyonu ise [16]'da önerilmiştir. Önerilen bu algoritmalar probleme sonlu ε- çözüm bulmaktadır.

Bu çalışmada baz olarak alınan algoritma, [16]'da geliştirilen Benson tipi dış yaklaşıklama algoritmasıdır. [16]'da bu temel algoritmanın iki varyantı önerilmiş, bunların belirli varsayımlar altında doğru çalıştığı ispatlanmıştır. Ancak sınırlı sayıda test problemi çözülmüş ve iki varyantın verimlilik açısından kıyası yapılmamıştır. Bu çalışma ile [16]'da önerilen temel algoritmaya farklı bir varyant önerilmektedir. Bu varyant parametrik olarak tasarlanmıştır ve bu parametrenin alabileceği iki uç değer alındığında [16]'da önerilen iki varyant elde edilmektedir. Ancak parametre sonsuz farklı şekilde seçilebilmektedir, bu yüzden onu problemin kendi yapısına uygun olacak şekilde seçmeye yarayacak bir yaklaşım sunulmuştur. Bu yaklaşım kullanılarak alternatif varyantlar türetilmiş ve [16]'da önerilen iki (uç) varyant da dahil olmak üzere bu varyantlar sayısal testler kullanılarak birbirleri ile kıyaslanmıştır.

Sayısal kıyaslamalar için rastgele türetilmiş doğrusal problemler ile temel bir dışbükey çok amaçlı eniyileme problemi kullanılmıştır. Her iki problem tipi için de amaç fonksiyonu ikiden altıya kadar değiştirilmiştir ve varyantların çalışma zamanları ile çözdükleri skalerizasyon modeli sayıları bu problem kümeleri üzerinden karşılaştırılmıştır. Doğrusal problemler ele alındığında, varyantların verimliliğinin problem boyutuna bağlı olarak değişimi için bir çıkarım yapılabilirken, incelenen dışbükey örnek için genel bir çıkarım yapmak mümkün olmamıştır.

2. BAŞLANGIÇ (PRELIMINARIES)

Bir $A \subseteq \mathbb{R}^p$ kümesinin dışbükey örtüsü, sınırı ve öziçi (İng. İnterior) sırasıyla $dışb\ A$, $snr\ A$ ve $int\ A$ ile gösterilmektedir. $A \subseteq \mathbb{R}^p$ dışbükey bir küme ve $F \subseteq A$ onun dışbükey bir altkümesi olsun. Eğer, $0 < \lambda < 1$ ve $a^1, a^2 \in A$ için "$\lambda a^1 + (1-\lambda)a^2 \in F$ ancak ve ancak $a^1,\ a^2 \in F$" ilişkisi varsa, F kümesine A'nın bir *yüz*ü denir. Sıfır boyutlu bir yüze *köşe*; bir boyutlu bir yüze ise *kenar* denir ([20]).

$A, B \subseteq \mathbb{R}^p$ kapalı kümeleri arasındaki Hausdorff uzaklık,

$$H(A,B) \coloneqq \max\left\{\sup_{x\in A}\inf_{y\in B}\|x-y\|, \sup_{y\in B}\inf_{x\in A}\|x-y\|\right\}$$

olarak tanımlanır ([21]). Burada $\|*\|$, $\mathbb{R}^p$'de tanımlı bir normdur. Bu çalışmada Öklid normu olarak alınmıştır.

Bu çalışmada $y, y' \in \mathbb{R}^p$ noktalarının arasındaki tam olmayan sıralama ilişkisi aşağıdaki şekilde gösterilmiştir:

$$y \leq y' :\Leftrightarrow y_i \leq y_i', \text{her } i = 1,\dots,p \text{ için}$$
$$y < y' :\Leftrightarrow y_i < y_i', \text{her } i = 1,\dots,p \text{ için}$$
$$y \lneqq y' :\Leftrightarrow y \leq y' \text{ ve } y \neq y'.$$

Ayrıca, $\mathbb{R}^p_+ = \{y \in \mathbb{R}^p : \text{Her } i \text{ için}, y_i \geq 0\}$ ile p boyutlu uzaydaki pozitif koni, $e \in \mathbb{R}^p$ ile birlerden oluşan $(1,\dots,1)^T$ vektörü ve $e^i \in \mathbb{R}^p$ ile i'inci bileşeni bir olan birim $(0,\dots,1,\dots,0)^T$ vektörü gösterilmektedir. Makale boyunca kümeler arasındaki toplam $(+)$ işlemi Minkowski toplamını ifade etmektedir, yani, $A, B \subseteq \mathbb{R}^p$ olmak üzere $A + B \coloneqq \{a + b : a \in A,\ b \in B\}$.

3. PROBLEM TANIMI VE ÇÖZÜM YAKLAŞIMLARI (PROBLEM DEFINITION AND SOLUTION APPROACHES)

$f = (f_1,\dots,f_p)^T : \mathbb{R}^n \longrightarrow \mathbb{R}^p$ vektör değerli bir fonksiyon ve $X \subseteq \mathbb{R}^n$ dışbükey kapalı bir küme olsun. Her i için $f_i : X \longrightarrow \mathbb{R}$ dışbükey fonksiyon olmak üzere bir dışbükey çok amaçlı enküçükleme problemi

(P) $$enk\text{üçü}kle \quad f(x) \quad \text{öyleki} \quad x \in X$$

olarak yazılabilir.

Olurlu kümenin f altındaki görüntüsü $Y \coloneqq f(X) = \{f(x) : x \in X\}$ olsun. $\mathcal{P} \coloneqq f(X) + \mathbb{R}^p_+$ kümesine (P) probleminin *üst görüntü kümesi* denmektedir. Üst görüntü kümesi dışbükeydir; ayrıca, eğer amaç fonksiyonu sürekli ise kapalıdır.

Tanım 3.1 ([13]): Amaç uzayındaki iki nokta y ve y' için $y \leq y'$ ise y *noktası* y' *noktasına baskındır*; $y < y'$ ise y *noktası* y' *noktasına güçlü baskındır* denir. Y kümesi içinde y noktasına baskın olan (kendisinden başka) bir nokta yoksa y noktası *baskın* noktadır. Benzer şekilde, y noktasına güçlü baskın olan bir nokta yoksa y noktası *zayıf baskın* noktadır. Amaç fonksiyonu altındaki görüntüsü (zayıf) baskın olan bir $x \in X$ çözümüne *(zayıf) etkin çözüm* denir.

Bir dışbükey çok amaçlı eniyileme problemi için bütün (zayıf) baskın noktalar üst görüntü kümesinin sınırında yer almaktadır ([7]). (P) problemini çözmekle kastedilen, (zayıf) etkin çözümler kümesi bulmaktır. Bulunan çözüm kümesinin tüm zayıf baskın noktalar kümesini ($f(X) \cap snr\,\mathcal{P}$) 'kaliteli' şekilde temsil etmesi beklenir. Bu çalışmada, motivasyonunu küme değerli eniyileme problemlerinden alan aşağıdaki çözüm kavramı kullanılacaktır.

Tanım 3.2 ([16]): Sonlu sayıda (zayıf) etkin çözümden oluşan $\bar{X} \subseteq X$ altkümesi $\mathcal{P} \subseteq dışb\left(f(\bar{X}) + \mathbb{R}^p_+\right) - \varepsilon\{e\}$ ilişkisini sağlıyorsa $\bar{X}$'e *sonlu (zayıf)* ε*-çözüm* denir.

Bu tanım, üst görüntü kümesine aşağıdaki gibi içerden ve dışardan yaklaşık çokyüzlüler türetmektedir:

$$dışb\left(f(\bar{X}) + \mathbb{R}^p_+\right) \subseteq \mathcal{P} \subseteq dışb\left(f(\bar{X}) + \mathbb{R}^p_+\right) - \varepsilon\{e\}.$$

Bu çok yüzlüler arasındaki Hausdorff uzaklık tam olarak $\varepsilon\|e\| = \varepsilon\sqrt{p}$ kadardır. Eğer problem doğrusal ise, yani, $C \in \mathbb{R}^{p\times n}, A \in \mathbb{R}^{n\times m}, b \in \mathbb{R}^m$ olmak üzere $f(x) = Cx$ ve $X = \{x \in \mathbb{R}^n : Ax \leq b\}$ şeklinde verilmişse, o zaman üst görüntü kümesi kesin olarak hesaplanabilmektedir. Bir diğer deyişle, $\varepsilon = 0$ alınabilmektedir. Elbette, doğrusal olmayan dışbükey problemlerde bu mümkün değildir. Ancak, eğer dışbükey problemin olurlu kümesi tıkız (İng. Compact) ise her $\varepsilon > 0$ için bir sonlu ε-çözüm vardır ([16]).

Yazında etkin ya da zayıf etkin çözümlerin bulunabilmesi için farklı yaklaşımlar bulunmaktadır. Yaygın bir yaklaşım (P) probleminden türetilmiş, tek amaç fonksiyonlu ve çözüldüğünde (zayıf) etkin çözüm bulan bir eniyileme modeli, diğer adı ile skalerizasyon modeli, çözmektir. Skalerizasyon modellerinde genel olarak problem parametreleri haricinde model parametreleri vardır ve bu model parametreleri değiştirilerek farklı (zayıf) baskın noktalar bulmak mümkündür. Yakın zamanda bu modellerin kıyaslandığı bir çalışma Kasımbeyli v.d. tarafından yazına kazandırılmıştır [14].

Doğrusal ve dışbükey problemler için sıkça kullanılan iki skalerizasyon modeli, ağırlıklı toplam ve Pascoletti-Serafini skalerizasyonlarıdır. Ağırlıklı toplam skalerizasyonunda model parametresi olarak ağırlık vektörü ($w \in \mathbb{R}^p$) bulunur:

(P(w)) $$enk\text{üçük}le \quad w^T f(x) \quad \text{ö}yleki \quad x \in X.$$

Eğer $w \in \mathbb{R}^p_+$ ise modelin eniyi çözümü (P) problemi için zayıf etkin çözümdür; eğer her i için $w_i > 0$ ise o zaman etkin çözümüdür ([13]).

Çok amaçlı eniyileme problemlerinde *ideal nokta*, $y^I \in \mathbb{R}^p$, her amaç fonksiyonunun diğerlerinden bağımsız olarak alabileceği en iyi değerin oluşturduğu vektördür, matematiksel olarak her $i = 1, \dots, p$ için $y_i^I = \inf_{x \in X} f_i(x)$ olarak tanımlanabilir. Dolayısı ile ideal noktanın i'yinci bileşeni (P(e^i)) skalerizasyon modelinin eniyi değeridir.

Pascoletti-Serafini skalerizasyon modelinde biri referans noktası ($v \in \mathbb{R}^p$) ve biri de yön vektörü ($d \in \mathbb{R}^p$) olmak üzere iki parametre vardır:

(P(v, d)) $$enk\text{üçük}le \quad z \quad \text{ö}yleki \quad x \in X, \quad f(x) \leq v + zd.$$

Eğer $d \neq 0 \in \mathbb{R}^p$ ise modelin eniyi çözümü, (P) problemi için zayıf etkin çözümdür ([19, 5]). Bu çalışma boyunca $d = e$ olarak alınacaktır. Pascoletti-Serafini modelinin Lagrange çifteş problemi

(D(v, d)) $$enb\text{üyük}le \quad \left(\inf_{x \in X} w^T f(x) - w^T v\right) \quad \text{ö}yleki \quad w \geq 0, \; w^T d = 1$$

olarak yazılabilir. Çifteş problemin eniyi çözümü kullanılarak üst görüntü kümesi $\mathcal{P}$'ye bir destekleyici (İng. Supporting) hiperdüzlem bulmak mümkündür.

Teorem 3.3 ([16]): Eğer X kümesinin öziçi boş küme değilse o zaman hem (P(v, e)) hem de (D(v, e)) probleminin eniyi çözümü, (x^*, z^*) ve w^*, vardır ve aralarında güçlü çifteşlik ilişkisi geçerlidir. Ayrıca, $h := \{y \in \mathbb{R}^p : (w^*)^T y = (w^*)^T f(x^*)\}$ hiperdüzlemi üst görüntü kümesi $\mathcal{P}$'yi $f(x^*)$ noktasında destekler ve $H := \{y \in \mathbb{R}^p : (w^*)^T y \geq (w^*)^T f(x^*)\}$ yarıuzayı $\mathcal{P}$'yi içerir.

4. BENSON TİPİ BİR YAKLAŞIKLAMA ALGORİTMASI (A BENSON TYPE APPROXIMATION ALGORITHM)

Dışbükey çok amaçlı eniyileme problemlerine Tanım 3.2'de verildiği gibi zayıf ε-çözüm üreten temel ve (geometrik) çifteş algoritmalar [16]'da verilmiştir. Bu çalışmada, amaç uzayını baz alan ve Benson tipi bir algoritma olan temel algoritma üzerinde durulacaktır.

Temel algoritmanın çalışma mantığı kısaca şu şekilde özetlenebilir: Algoritma başlangıçta üst görüntü kümesi $\mathcal{P}$'yi kapsayan bir çokyüzlü bulur. Bu ilk dışyaklaşık çokyüzlü, genelde, y^I ideal nokta olmak üzere $\mathcal{P}^0 := y^I + \mathbb{R}^p_+$ olarak alınmaktadır. Elbette $\mathcal{P}^0$ kümesinin tek köşesi ideal noktadır.

Algoritma k'yinci yinelemede o andaki dışyaklaşık çokyüzlü $\mathcal{P}^k$'nin köşelerini bulur. Her bir v^k köşesi için (P(v^k, e)) Pascoletti-Serafini skalerizasyon modelini çözerek bu köşenin üst görüntü kümesine e yönündeki 'uzaklığını' hesaplar. Eğer bu uzaklık istenilen hata payı ε'dan büyükse o zaman Teorem 3.3 kullanılarak $\mathcal{P}$'yi kapsayan ama v^k'yi dışında bırakan H yarıuzayı bulunur. Bir sonraki yineleme için geçerli olacak dışyaklaşık çokyüzlü $\mathcal{P}^{k+1} = \mathcal{P}^k \cap H$ olarak güncellenir. Eğer bir dışyaklaşık çokyüzlünün tüm köşeleri için hesaplanan uzaklık ε'dan küçükse algoritma sonlanır. Algoritmanın sözde-kodu aşağıda verilmiştir:

Algoritma 4.1: (P) problemi için temel yaklaşıklama algoritması

Başlangıç:

Her $i = 1, \dots, p$ için (P(e^i)) modelini çöz, eniyi çözüm x^i olsun.

$\bar{X} = \{x^1, \dots, x^p\}$, $y^I = \left(f_1(x^1), \dots, f_p(x^p)\right)^T$, $\mathcal{P}^0 := y^I + \mathbb{R}^p_+$, $k = 0$.

Ana döngü:

1. M $= \mathbb{R}^p$ olsun.
2. $\mathcal{P}^k$'nin köşelerini bul, bu küme $\mathcal{V}^k$ olsun.
3. $j = 1$ olsun.
4. Eğer $j > \left|\mathcal{V}^k\right|$ ise Adım 5'e git. Eğer $j \leq \left|\mathcal{V}^k\right|$ ise
   a. $\mathcal{V}^k$'nın j'nci elemanı v olsun. (P(v, e)) ve (D(v, e)) problemlerini çöz, sırasıyla (x^*, z^*) ve w^* eniyi çözümlerini bul.
   b. $\bar{X} \leftarrow \bar{X} \cup \{x^*\}$ olarak güncelle.
   c. Eğer $z^* > \varepsilon$ ise
      i. $M \leftarrow M \cap \{y \in \mathbb{R}^p : (w^*)^T y \geq (w^*)^T f(x^*)\}$
      ii. (Varyant 1): Adım 5'e ilerle
      (Varyant 2): $j \leftarrow j + 1$ olarak güncelle ve Adım 4'e git.
   d. Eğer $z^* \leq \varepsilon$ ise $j \leftarrow j + 1$ olarak güncelle ve Adım 4'e git.
5. Eğer M $\neq \mathbb{R}^p$ ise $\mathcal{P}^{k+1} \leftarrow \mathcal{P}^k \cap M$, $k \leftarrow k + 1$ olarak güncelle ve Adım 2'ye git. Eğer M $= \mathbb{R}^p$ ise dur: $\bar{X}$ bir sonlu (zayıf) ε-çözümdür.

Algoritma, başlangıçta p adet (her e^i için) ağırlıklı toplam skalerizasyonu çözerek, problemin ideal noktası y^I'yı bulur. Çözülen skalerizasyon sonucu bulunan eniyi değerler, zayıf etkin çözümler olarak $\bar{X}$ kümesine eklenir. İlk dış yaklaşık küme $\mathcal{P}^0 := y^I + \mathbb{R}^p_+$ olarak tanımlanır ve yineleme sayısı $k = 0$ olarak atanır.

Her yinelemede ilk adım o anki dış yaklaşık kümenin köşelerini bulmaktır. İlk yineleme için bunun sadece y^I noktası olduğu açıktır. Ancak, daha sonraki yinelemelerde bu aşamada bir köşe sıralaması (İng. Vertex enumeration) problemi, yani sonlu sayıda yarıuzayın kesişimi olarak tanımlanmış bir çokyüzlünün (bkz. Adım 5 ve Adım 4.c.i.) köşelerinin hesaplanması problemini, çözmek gerekmektedir. Bunun için geliştirilmiş farklı algoritmalar mevcuttur. Bu çalışmada Löhne ve Weißing tarafından geliştirilmiş MATLAB uyumlu *bensolve* programı kullanılmıştır ([17, 18]).

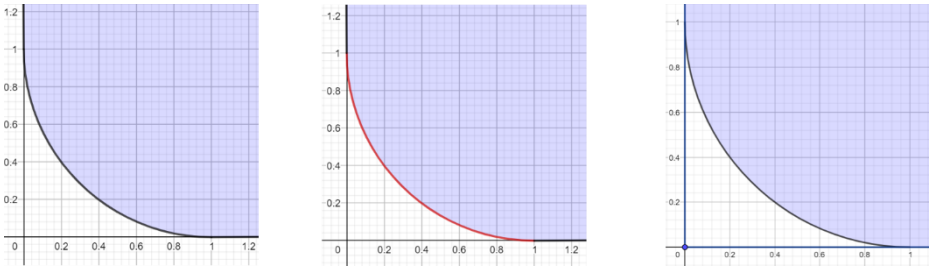
Algoritmanın k'yinci yinelemede bulduğu köşelerin kümesi $\mathcal{V}^k$ ve bunların sayısı $\left|\mathcal{V}^k\right|$ olsun. Adım 4'te, bulunan tüm bu köşeler sırasıyla ele alınmaktadır. Her bir köşe (v) için (P(v, e)) ve (D(v, e)) problemleri çözülerek sırasıyla (x^*, z^*) ve w^* eniyi çözümleri bulunur. x^* bir zayıf etkin çözüm olarak $\bar{X}$ kümesine eklenir. Eğer $z^* \leq \varepsilon$ ise incelenen v köşesinin üst görüntü kümesi $\mathcal{P}$'ye e yönündeki uzaklığı ε'dan küçük olduğu için algoritma (eğer varsa), sıradaki köşeyi ele alır. Eğer $z^* > \varepsilon$ ise v köşesi $\mathcal{P}$'ye yeterince yakın olmadığı için o anki dış yaklaşık kümenin ($\mathcal{P}^k$'nin) güncellenmesi gerekmektedir.

Güncellenme aşamasında algoritmanın iki varyantı bulunmaktadır. İlk varyantta $M = H = \{y \in \mathbb{R}^p : (w^*)^T y \geq (w^*)^T f(x^*)\}$ olarak hesaplanır ve ilk kez uzak bir köşe bulunduğunda Adım 5'e ilerlenerek dış yaklaşık küme güncellenir. Böylece $\mathcal{P}^{k+1}$ kümesi, $\mathcal{P}^k$ kümesinin tek bir yarıuzayla kesiştirilmesi sonucu elde edilir. Algoritma Adım 2'ye dönerek bu kümenin köşelerini bulur.

İkinci varyantta ise her yinelemede, mutlaka her köşe için (P(v, e)) modeli çözülür. Herhangi bir köşe için $z^* > \varepsilon$ olarak bulunmuşsa elde edilen yarıuzay direkt olarak $\mathcal{P}^k$ kümesi ile kesiştirilmek yerine, M kümesi ile kesiştirilerek saklanır. Tüm köşeler için bu işlem tamamlandığında, Adım 5'te $\mathcal{P}^k$ kümesi bulunan tüm yarıuzaylarla, tek seferde kesiştirilerek güncellenir. Algoritma yine Adım 2'ye dönerek köşe sıralaması problemi çözer.

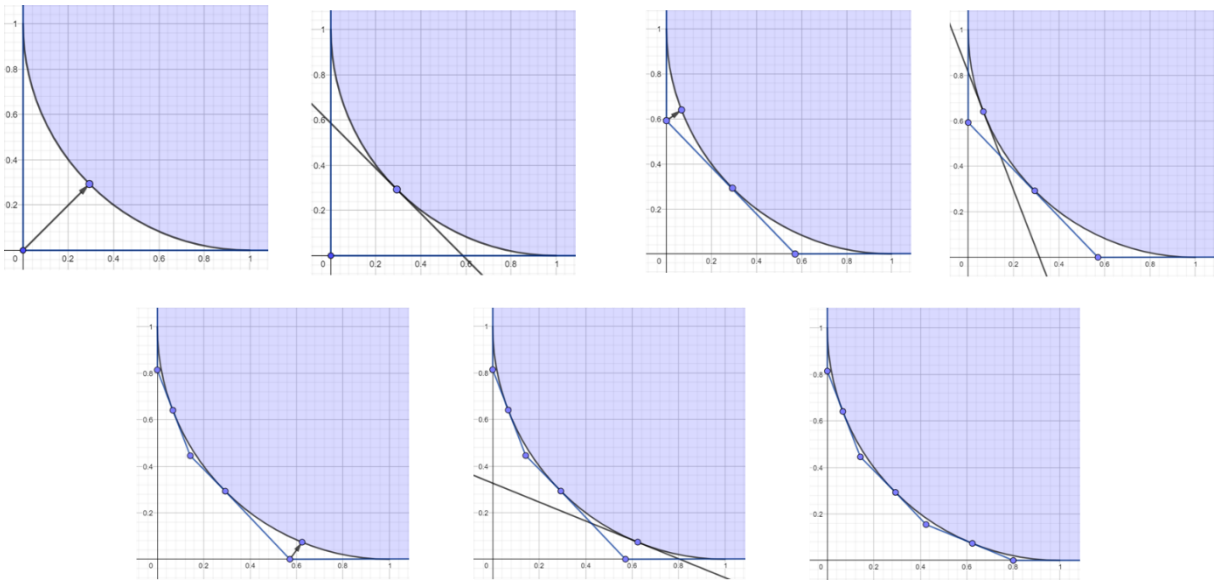
Herhangi bir yinelemede bulunan tüm köşeler üst görüntü kümesine yeterince yakın olarak bulunmuşsa M kümesi güncellenmeden kalır ve algoritma sonlanır. Aşağıda basit bir örnek üzerinde algoritma görsellerle anlatılmıştır.

Örnek 4.2: (P) problemi için $f(x) = x$ ve $X = \{x \in \mathbb{R}^2_+ : (x_1 - 1)^2 + (x_2 - 1)^2 \leq 1\}$ olsun. Bu durumda $n = p = 2$'dir. Problemin üst görüntü kümesi, baskın noktaları ve ideal noktası Şekil 1'de verilmiştir.



Şekil 1. *Örnek 4.2 için sırası ile üst görüntü kümesi (mavi küme), baskın noktalar kümesi (kırmızı çizgi) ve ideal nokta (koyu mavi nokta ile gösterilmiş* 0 *noktası)*

Şekil 2'de ise Algoritma 4.1 Varyant 1'in çalışma basamakları gösterilmiştir. Figür 1 ilk döngüde çözülen Pascoletti-Serafini skalerizasyonunu, Figür 2 bu model sonucu çifteş çözüm kullanılarak bulunan destekleyici hiperdüzlemi gösterir. İlk yineleme sounucunda Figür 3 elde edilmiştir. İkinci yinelemeden sonra Figür 5 ve üçüncü yinelemeden sonra Figur 7 elde edilir. Bu figürdeki tüm köşeler üst görüntü kümesine yeterince yakın olduğu için Algoritma sonlanır. Şekil 2 kullanılarak Varyant 2'nin çalışma basamakları gösterilmek istense, Figür 5'in Figür 3 üzerine ve Figür 6'nın Figür 4 üzerine taşınması gerekirdi. Bu durumda Figür 7, ikinci döngü sonunda elde edilmiş olurdu.



Şekil 2*. Algoritma 1 (Varyant 1)'in Örnek 4.2 üzerinde uygulama adımları (İlk satır soldan sağa Figür 1-4 ve ikinci satır soldan sağa Figür 5-7)*

Algoritma 4.1'in doğru çalışarak probleme bir zayıf ε-çözüm verdiği bilinmektedir ([16]). [16]'da sunulan sınırlı sayıdaki (doğrusal olmayan) dışbükey sayısal örnekte algoritmanın ilk varyantının ikinci varyanta göre aynı ya da daha az sayıda skalerizasyon modeli çözerek durduğu gözlemlenmiştir. Bu örneklerden iki amaç fonksiyonuna sahip olanlarda her iki varyant da aynı şekilde çalışırken, üç ve dört amaç fonksiyonu olan örneklerde Varyant 1 daha verimlidir. Elbette, sadece bu örneklere bakarak Varyant 1'in her zaman daha iyi (ya da en azından Varyant 2 kadar iyi) bir verimlilikte çalışacağı çıkarımını yapmak mümkün değildir.

Aslında, Varyant 1 ve 2'nin farklı yönlerden daha güçlü olma potansiyelleri vardır. Varyant 1'de, dış yaklaşık küme güncellendiğinde henüz kontrol edilmemiş bazı köşelerin, bu güncelleme sonucunda elenmesi mümkündür. Elbette bu durumda bu köşeler için Pascoletti-Serafini modeli çözülmeyecektir. Varyant 2'nin bir avantajı ise tüm köşeler kullanılarak güncelleme yapıldığı için dış yaklaşık kümenin, üst

görüntü kümesine, daha 'dengeli' yakınsamasıdır. Diğer bir deyişle, bu varyantta dış yaklaşık kümenin üst görüntü kümesine olan Hausdorff uzaklığı her yinelemede mutlaka azalmaktadır. Bunun sayesinde, algoritma erken durdurulduğunda elde edilen dış yaklaşık kümenin, üst görüntü kümesini her bölümünde daha iyi temsil etmesi beklenir. Ayrıca, Varyant 2'de genel olarak çok daha az sayıda köşe sıralaması problemi çözülmesi beklenir. Elbette, bu problemlerde, o anki dış yaklaşık küme, aynı anda birden fazla sayıda yarıuzay ile kesiştirilmektedir. Dolayısı ile çözülen köşe sıralaması problemi Varyant 1'e göre yapısal olarak daha zordur.

5. BENSON TİPİ ALGORİTMAYA FARKLI BİR VARYANT (A DIFFERENT VARIANT FOR THE BENSON TYPE ALGORITHM)

Bu bölümde Algoritma 4.1'e parametrik olarak tasarlanmış bir varyant önerilmektedir. Bu varyant parametre seçimine göre Varyant 1 ve 2'yi kapsamaktadır. Varyantın Algoritma 4.1'den farklı olduğu adım 4.c. adımıdır. Bir önceki bölümde açıklandığı gibi algoritmada o anki dış yaklaşık küme ya ilk kez uzak bir köşeye denk gelindiğinde (Varyant 1) ya da tüm köşeler tarandıktan sonra (Varyant 2) güncellenmektedir. Bunun yerine, uzaklık için ikinci bir kontrol yapılmasını öneriyoruz. Buna göre, belirlenmiş bir $\tilde{\varepsilon} \geq \varepsilon$ için, Pascoletti-Serafini modeli çözülerek bulunan uzaklığın, $\tilde{\varepsilon}$'den büyük olup olmadığına bakılmalı; eğer büyükse o anki dış yaklaşık küme güncellenmeli, değilse diğer köşelere bakmaya devam edilmelidir. Algoritmanın bu varyantına ait 4.c. adımının sözde-kodu aşağıda verilmiştir.

Algoritma 5.1: (P) problemi için temel yaklaşıklama algoritmasına farklı bir varyant:

4. c. Eğer $z^* > \varepsilon$ ise
   i. $M \leftarrow M \cap \{y \in \mathbb{R}^p : (w^*)^T y \geq (w^*)^T f(x^*)\}$
   ii. Eğer $z^* < \tilde{\varepsilon}$ ise $j \leftarrow j + 1$ olarak güncelle ve Adım 4'e git.
   iii. Eğer $z^* \geq \tilde{\varepsilon}$ ise Adım 5'e ilerle.

Kolayca anlaşılacağı gibi eğer $\tilde{\varepsilon} = \varepsilon$ olarak alınırsa her zaman 4.c.iii. geçerli olur ve Algoritma 4.1'de verilen Varyant 1 elde edilir. Buna karşılık eğer $\tilde{\varepsilon}$ yeterince büyük seçilirse her zaman 4.c.ii. geçerli olur ve Algoritma 4.1'de verilen Varyant 2 elde edilir. Bu iki uç varyantın yanı sıra, $\tilde{\varepsilon}$'yi farklı şekillerde seçmek, algoritmanın farklı varyantları olmasını sağlamaktadır. Ayrıca, algoritma $\tilde{\varepsilon} \geq \varepsilon$ olarak alındığı sürece doğru çalışır ve [16]'da gösterilmiş olduğu gibi probleme bir zayıf ε-çözüm verir. Ancak seçilen $\tilde{\varepsilon}$ değeri algoritmanın performansını etkileyeceğinden, bu parametrenin nasıl seçileceği önemlidir.

Bu çalışmada $\tilde{\varepsilon}$ seçimi için problemin kendi yapısı kullanılmıştır. Bu prosedürü anlatmadan önce şu gözlemleri yapalım:

Açıklama 5.2: (a) Eğer bir dışbükey çokyüzlü (A) başka bir dışbükey kümeyi (B) kapsıyorsa, bu iki küme arasındaki Hausdorff uzaklık $H(A,B) := \sup_{x \in A} \inf_{y \in B} \|x - y\|$ olarak yazılabilir. Elbette, burada $\inf_{y \in B} \|x - y\|$, x noktasının B kümesine uzaklığıdır. Eğer $H(A,B) = \inf_{y \in B} \|\bar{x} - y\| > 0$ ifadesini sağlayan bir $\bar{x} \in A$ varsa o zaman $\bar{x}$, A'nın bir köşesi olarak alınabilir. **(b)** Ayrıca, eğer $A \supseteq C \supseteq B$ ifadesini sağlayan bir C kümesi varsa o zaman, $H(A,B) \geq H(A,C)$ doğrudur.

Algoritma boyunca elde edilen dış yaklaşık kümeler $\mathcal{P}^0 \supseteq \mathcal{P}^1 \supseteq \cdots \supseteq \mathcal{P}^k \supseteq \mathcal{P}^{k+1} \supseteq \cdots \supseteq \mathcal{P}$ ilişkisini sağladığı için Açıklama 5.2.'ye göre algoritma boyunca ele alınacak tüm köşeler içinde üst görüntü kümesine en uzak olanı $\mathcal{P}^0$ kümesinin tek köşesi olan ideal noktadır. Burada dikkat edilmesi gereken, algoritma içinde bir köşenin üst görüntü kümesine olan Öklid uzaklığı değil, bunun yerine Pascoletti-Serafini skalerizasyonu çözülerek, köşeye kaç e eklenirse ilk kez üst görüntü kümesine çarpacağı hesaplanmaktadır. Elbette üst görüntü kümesine ilk kez değen nokta $(v + z^* e)$ ile köşe (v) arasındaki uzaklık, köşenin üst görüntü kümesine olan Öklid uzaklığından daha büyük olacaktır. Yine de bu iki mesafe arasında bir ilişki olduğu açıktır.

Çalışmada $\tilde{\varepsilon}$, çözülen ilk Pascoletti-Serafini modelinin (ki bu (P(y^I, e)) modelidir) eniyi değerine bağlı olarak seçilmektedir. Bu modelin eniyi değeri z^I olsun. Bu durumda ilerleyen yinelemelerde Pascoletti-Serafini modeli çözüldüğünde eniyi değer olarak z^I'dan daha küçük değerler elde edilmesi beklenebilir.

(Elbette bu her zaman doğru olmak zorunda değildir.) Bu gözlemden faydalanarak algoritmanın önerdiğimiz varyantında $k \geq 1$ olmak üzere farklı k sayıları için $\tilde{\varepsilon} = \frac{z^I}{k}$ olarak alınmıştır.

6. BİLGİSAYIMSAL TESTLER (COMPUTATIONAL TESTS)

Algoritma 4.1'de önerilen iki varyant ile bu çalışmada önerilen varyant, doğrusal çok amaçlı eniyileme problemleri ile özel yapılı dışbükey çok amaçlı eniyileme problemleri üzerinde test edilmiştir. Bütün test problemleri için $k = 1, 2, 3, 4, 5, 10, \infty$ olarak alınmak üzere toplamda yedi varyant kıyaslanmıştır. $k = 1$ durumu Algoritma 4.1'de verilen Varyant 1'e ve $k = \infty$ durumu Varyant 2'e denk gelmektedir. Algoritma ve tüm varyantları MATLAB (R2019a versiyonu) kullanılarak uygulanmıştır. Skalerizasyon modelleri için çözücü olarak CVX kullanılmıştır ([10, 11]). Testler sistem özellikleri Intel(R) Core(TM) i7-4790 CPU@ 3.60GHz, 4.00GB, x64 Windows 10 olan bir bilgisayar kullanılarak yapılmıştır.

6.1. Doğrusal Problemler

Algoritma varyantlarının çalışma performansları öncelikle rastgele oluşturulmuş doğrusal problemler üzerinde test edilmiştir. Bilindiği gibi p boyutlu bir doğrusal çok amaçlı eniyileme problemi $C \in \mathbb{R}^{p \times n}, A \in \mathbb{R}^{n \times m}$ ve $b \in \mathbb{R}^m$ olmak üzere (P) probleminde $f(x) = Cx$ ve $X = \{x \in \mathbb{R}^n : Ax \leq b\}$ alınarak yazılabilir. Rastgele testler için A ve C matrislerinin her bileşeni ortalama değeri $\mu = 0$ ve varyans değeri $\sigma^2 = 100$ olan bağımsız normal dağılımlar kullanılarak, b vektörünün her bileşeni ise aralığı $[0,10]$ olan bağımsız düzgün dağılımlar kullanılarak türetilmiştir. Nümerik karmaşıklıkları engellemek için bu matris ve vektörlerin her bileşeni, kendisine en yakın tamsayıya yuvarlanmıştır. Bu şekilde rastgele bir problem türetildikten sonra, ilk olarak, problemin olurlu ve sınırlı (ideal noktanın her bileşeninin sonlu olması bağlamında) olduğu kontrol edilmiş; bu koşulları sağlıyorsa test problemi olarak alınmıştır.

Bu çalışmada sayısal testler için $p = 2, 3, 4, 5, 6$ boyutlu örnekler türetilmiştir. İki amaçlı problemler ($p = 2$) için $n = 10$ ve $m = 20$ olarak alınmış, diğer tüm problem kümeleri için ise $n = 5$ ve $m = 10$ olarak alınmıştır. 2-5 boyutlu problem kümeleri 20şer, 6 boyutlu problem kümesi ise 10 adet olurlu ve sınırlı doğrusal problemden oluşmaktadır. Tüm problemler için $\varepsilon = 10^{-5}$ olarak alınmıştır. Tablo 1'de her bir problem kümesi için yedi varyantın ortalama çalışma süreleri (saniye cinsinden) görülmektedir. Buna göre Varyant 1 ($k = 1$) tüm problem kümelerinde en kötü sonucu vermiştir. İki boyutlu örnekler için diğer tüm örnekler benzer performans gösterirken, $p = 3, 4, 5, 6$ boyutlu örnekler için sırası ile $k = \infty, 10, 3, 5$ varyantları en iyi sonucu vermiştir.

Çalışma zamanları üzerinden, varyantları farklı bir şekilde daha kıyaslamak için, problem kümelerindeki her bir örnek için her varyantın o örneği en kısa zamanda çözen varyanttan yüzde kaç daha yavaş çözdüğü ve o örneği en uzun zamanda çözen varyanttan yüzde kaç daha hızlı çözdüğü hesaplanmıştır. Daha sonra, her problem kümesi için bu yüzdelerin ortalamaları alınarak Tablo 2 oluşturulmuştur. Daha detaylı açıklamak gerekirse, her bir problem yedi farklı varyantla çözüldüğünde çözüm süreleri $T_1, \dots, T_\infty$ olarak kaydedilir. Burada T_i, $(k = i)$ varyantının problemi çözme süresidir. Daha sonra her k için $\left(\frac{T_k - \min_i T_i}{\min_i T_i} \times 100\right)$ ve $\left(\frac{\max_i T_i - T_k}{\max_i T_i} \times 100\right)$ değerleri hesaplanmış ve son olarak o problem kümesindeki tüm problemler üzerinden ortalama alınmıştır.

Tablo 1. *Rastgele oluşturulmuş doğrusal problemler için algoritma varyantlarının ortalama çalışma süreleri (sn)*

p \ k	1	2	3	4	5	10	∞
2	4,05	3,98	3,96	3,97	3,97	3,96	3,96
3	6,34	6,08	6,13	6,06	6,06	6,06	**5,99**
4	21,20	18,23	18,11	18,12	18,21	**17,44**	17,88
5	48,26	36,43	**35,86**	36,64	36,08	37,07	38,64
6	477,43	114,53	148,82	**90,86**	114,61	181,09	112,76

Tablo 2. *Rastgele oluşturulmuş doğrusal problemler için varyantların çalışma sürelerinin birbiri ile problem bazında kıyası*

En iyi varyanttan ne kadar yavaş?							
p \ k	1	2	3	4	5	10	∞
2	2,67	1,92	1,61	1,76	1,85	1,51	**1,28**
3	8,87	6,05	6,94	5,86	5,42	5,11	**4,92**
4	22,16	10,91	10,01	9,33	10,31	**4,87**	7,40
5	50,59	11,68	10,21	11,66	**9,47**	17,59	20,13
6	407,18	33,50	74,39	**4,66**	30,52	106,02	28,91
En kötü varyanttan ne kadar hızlı?							
p \ k	1	2	3	4	5	10	∞
2	2,31	2,98	3,29	3,17	3,02	3,36	**3,58**
3	2,84	5,35	4,58	5,48	5,87	6,13	**6,31**
4	6,04	12,83	13,72	14,48	13,69	**17,54**	15,78
5	6,01	27,22	27,74	27,29	**28,67**	24,76	23,40
6	2,90	68,23	59,57	**76,17**	70,12	50,78	71,65

Tablo 2 incelendiğinde iki ve üç amaç fonksiyonu olan problemler için Varyant 2 $(k = \infty)$'nin en iyi sonucu verdiği ve problem boyutu arttıkça k değerini küçültmenin daha etkili olduğu gözlenmiştir: Sırası ile $p = 4, 5, 6$ boyutlu örnekler için $k = 10, 5, 4$ varyantları en iyi sonucu vermiştir.

6.2. Doğrusal Olmayan Bir Örnek

(P) problemi için $f(x) = x$ ve $X = \{x \in \mathbb{R}^n_+ : \sum_{i=1}^{n}(x_i - 1)^2 \leq 1\}$ olsun. Bu durumda $p = n$'dir. Eğer $p = 2$ olarak alınırsa Örnek 4.2 elde edilmiş olur. Bu problem, p boyutlu birim küre örneği olarak adlandırılabilir.

Bu problemlerde Pareto kümeyi tam olarak veren bir kesin çözüm bulunması mümkün değildir. Bu yüzden hata oranı ε seçimi algoritmanın çalışma hızını fazlaca etkileyecektir. Testler için farklı ε değerleri belirlenmiştir. Bu değerler belirlenirken iki noktaya dikkat edilmiştir. Birincisi, çalışma sürelerinin çok fazla uzamasını engellemek için problemin boyu arttıkça hata payı da artırılmıştır. Ayrıca, özellikle problemin boyutu büyüdükçe, ya skalerizasyon modelleri için kullanılan çözücü (CVX) ya da köşe sıralaması problemini çözmek için kullanılan program (*bensolve*) hata verebilmektedir. Testlerde kullanılan ε değerleri, her iki programın da hata vermeden çözüme ulaştığı en küçük değerler olarak alınmaya çalışılmıştır.

Bu problemlerde, CVX çözücüsü bazen skalerizasyon modelini çözse dahi çözümün kesin olmadığına dair bir uyarı gönderebilmektedir. Bu uyarı, iki boyutlu örnekler de dahil olmak üzere, çözülen tüm birim küre örneklerinde, bazı köşeler için çözülen Pascoletti-Serafini modellerinde görülmüştür. Bu çalışmada CVX için 'yüksek kesinlik' modu tercih edilmiştir. Buna göre, çalıştırılan skalerizasyon modeli için hata payı 10^{-12}'den küçükse model kesin bir şekilde çözülmüş sayılır; hata payı 10^{-12} ile 10^{-6} arasındaysa kesin olmayan bir şekilde çözülmüş kabul edilir. Bu hata payları bizim çok amaçlı problem için aldığımız ε değerlerinden çok daha küçük olduğu için, sonuçların hala geçerli olduğu düşünülebilir. Ancak, algoritma için çifteş çözümün de o anki dış yaklaşık kümeyi güncellemek için kullanıldığı gözden kaçırılmamalıdır.

Dolayısı ile bu aşamada kesin olmayan bir çözüm algoritmanın genel yapısını etkiyebilme potansiyeline sahiptir.

Tablo 3. *Birim küre örneği için algoritma varyantlarının çalışma süreleri (sn)*

p | ε \ k		1	2	3	4	5	10	∞
2	10^{-4}	33,31	28,47	30,50	28,59	28,72	29,07	28,39
2	10^{-5}	127,79	106,35	106,37	106,22	107,16	107,25	106,33
3	0,1	8,07	9,94	9,86	9,92	8,44	**7,93**	11,72
3	0,01	74,29	69,37	69,47	69,31	75,66	82,92	**60,90**
4	0,5	16,35	16,31	16,40	16,38	16,35	16,31	**13,87**
4	0,15	**39,79**	88,64	79,40	69,05	76,93	**39,75**	67,43
5	1	29,56	18,25	18,51	18,23	18,21	18,18	**13,76**
5	0,5	86,33	88,63	85,12	85,14	85,63	84,92	**43,92**
6	1	148,16	148,42	148,02	147,17	148,49	148,39	173,95

Tablo 3'te ilk kolon amaç fonksiyonu sayısını, ikinci kolon ise bu problem için alınan hata payı oranını göstermektedir. Altı boyutlu problem hariç tüm problemlerde iki farklı hata payı alınarak varyantların performansının hata payı değiştiğinde nasıl bir değişiklik gösterdiğine de bakılmıştır.

Tablo 3 incelendiğinde doğrusal problemlerde olduğu gibi genel bir sonuç çıkarmanın mümkün olmadığı gözlenmiştir. İki boyutlu problemde her iki hata payı değeri için de Varyant 1 en kötü performansı gösterirken diğer tüm varyantlar benzer performans göstermiştir. Üç boyutlu problemde ise hata payı değiştirildiğinde, en iyi ve en kötü performansı gösteren varyantlar yer değiştirmektedir. Tek başına bu gözlem bile genel bir çıkarım yapmanın doğru olmayacağını göstermektedir. Dört boyutlu problemde, hata payı büyük olduğunda bariz bir kazanan yokken, hata payı küçüldüğünde $\boldsymbol{k = 1}$ ve $\boldsymbol{k = 10}$ varyantlarının diğerlerine göre daha iyi sonuç verdiği görülmektedir. Beş boyutlu problemde her iki hata payı değeri için de Varyant 2 daha iyi çalışırken altı boyutlu problemde aynı varyant en kötü performansı göstermiştir. Tüm problemlere bakıldığında $\boldsymbol{k = 1, 2, 10, \infty}$ varyantları zaman zaman en iyi ya da en kötü performansı göstermiştir. Ancak $\boldsymbol{k = 3, 4, 5}$ varyantları hiçbir zaman en iyi ya da en kötü performansa sahip olmamıştır.

7. SONUÇ (CONCLUSION)

Bu çalışmada, dışbükey çok amaçlı eniyileme problemleri için geliştirilmiş bir dış yaklaşıklama algoritması ele alınmış ve algoritmanın daha önce geliştirilmiş iki varyantına ek olarak yeni varyantlar geliştirilmiştir. Geliştirilen yeni varyantlar parametrik olarak ve alınan parametrenin uç değerleri ile daha önce geliştirilmiş iki varyant elde edilecek şekilde tasarlanmıştır. Çalışmada ayrıca, parametrik olarak tasarlanan bu varyantlar için parametre seçimin nasıl yapılabileceğine dair bir öneri getirilmiş ve buna göre oluşturulan beş yeni varyant bilgisayımsal testler için kullanılmıştır.

Testler için öncelikle rastgele türetilmiş doğrusal örnekler kullanılmıştır. Amaç fonksiyonu sayısı ikiden altıya kadar artırılmış ve her boyut için problem kümeleri oluşturulmuştur. Bu problemler tüm varyantlar ile çözdürülerek varyantların çözüm zamanları ile çözdükleri toplam skalerizasyon sayıları kıyaslanmıştır. Doğrusal problemler için daha önce geliştirilmiş ve bu çalışmada Varyant 1 olarak adlandırılan varyantın, en kötü performansı verdiği gözlemlenmiştir. Buna karşılık iki ve üç boyutlu problemlerde, yine daha önce geliştirilmiş ve bu çalışmada Varyant 2 olarak adlandırılan varyantın, diğerlerinden daha iyi çalıştığı gözlemlenmiştir. Problem boyutu arttıkça bu çalışmada geliştirilmiş ara varyantların, daha önceden geliştirilmiş bu iki varyanta göre daha iyi sonuç verdiği gözlemlenmiştir. Bu yüzden, doğrusal problemler için ve özellikle de problem boyutu artırıldığında, geliştirilen varyantların ümit verici olduğunu söylemek mümkündür.

Doğrusal problemlere ek olarak, problemin yapısını değiştirmeden, amaç fonksiyonu ve karar değişkeni sayısının istenildiği kadar artırılabileceği bir doğrusal olmayan dışbükey problem incelenmiştir. Bu problem tipi için de amaç sayısı ikiden altıya kadar artırılmıştır. Bu kez, hata payları da problemin boyutuna

göre farklı değerler alacak şekilde seçilmiştir. Bu problemler için, varyantların çözüm süreleri ve çözdükleri skalerizasyon modeli sayıları kıyaslandığında, genel bir çıkarım yapmak mümkün olmamıştır.

Çalışmada doğrusal olmayan problemler için de rastgele örnekler türetilip varyantların performansına bakmak, bu tip problemler için daha geçerli sonuçların elde edilmesini sağlayabilir. Ancak, bu çalışmada ele alınan tek doğrusal olmayan örnek bile, basit yapısına rağmen, zaman zaman çözülememiştir. Bunun sebebi ya CVX çözücüsünün ya da *bensolve* programının hata vermesidir. Dolayısı ile kullanılan bu çözücü ve programın sorunsuz şekilde çalıştığı örnekleri rastgele türetmek epey güçtür. Daha güçlü eniyileme ve köşe sıralaması çözücüleri ile böyle bir çalışma yapmak ileride mümkün olabilir. Örneğin, rastgele türetilmiş dışbükey ikinci dereceden çok amaçlı eniyileme problemleri için varyantların kıyaslaması yapılabilir.

TEŞEKKÜR (ACKNOWLEDGMENTS)

Bu çalışmada önerilen algoritma varyantların MATLAB kodlarının hazırlanmasında emeği geçen İrem Nur Keskin'e teşekkürlerimi sunarım.

KAYNAKLAR (REFERENCES)

[1] Armand,P.: Finding all maximal efficient faces in multiobjective linear programming. Mathematical Programming 61, 357 – 375 (1993)

[2] Armand, P. and Malivert, C.; Determination of the efficient set in multiobjective linear programming. Journal of Optimization Theory and Applications 70, 467-489 (1991)

[3] Benson, H.P.: An outer approximation algorithm for generating all efficient extreme points in the outcome set of a multiple objective linear programming problem. Journal of Global Optimization 13, 1-24 (1998)

[4] Csirmaz, L.: Using multiobjective optimization to map the entropy region. Computational Optimization and Applications, 63(1):45 – 67, 2016.

[5] Eichfelder, G.: Adaptive Scalarization Methods in Multiobjective Optimization (Springer, Berlin, 2008).

[6] Ehrgott, M., Löhne, A., Shao, L.: A dual variant of Benson's outer approximation algorithm. Journal of Global Optimization 52(4), 757–778 (2012)

[7] Ehrgott, M., Shao, L., Schöbel, A.: An approximation algorithm for convex multi-objective programming problems. Journal of Global Optimization 50(3), 397–416 (2011)

[8] Ehrgott, M., Wiecek, M. M.: Multiobjective programming. In: Figueira, J., Greco, S., Ehrgott, M., (eds.) Multiple Criteria Decision Analysis: State of the Art Surveys. Springer Science + Business Media, Berlin, 667–722 (2005)

[9] Evans, J.P., Steuer, R.E.: A revised simplex method for multiple objective programs. Mathematical Programming 5(1), 54–72 (1973)

[10] Grant, M. and Boyd, S.: CVX: Matlab software for disciplined convex programming, version 2.0 beta. http://cvxr.com/cvx (September 2013)

[11] Grant, M. and Boyd, S.: Graph implementations for nonsmooth convex programs, Recent Advances in Learning and Control (a tribute to M. Vidyasagar), Blondel, V., Boyd, S. and Kimura, H. editors, 95-110, Lecture Notes in Control and Information Sciences, Springer (2008).

[12] Hamel, A.H., Löhne, A., Rudloff, B.: Benson type algorithms for linear vector optimization and applications. Journal of Global Optimization 59(4), 811–836 (2014)

[13] Jahn, J.: Vector Optimization: Theory, Applications, and Extensions. Springer, Berlin (2004)

[14] Kasimbeyli, R., Ozturk, Z. K., Kasimbeyli, N., Yalcin, G. D., & Erdem, B. I. (2019). Comparison of some scalarization methods in multiobjective optimization. Bulletin of the Malaysian Mathematical Sciences Society, 42(5), 1875-1905.

[15] Löhne, A.: Vector Optimization with Infimum and Supremum. Springer, Berlin (2011)

[16] Löhne, A., Rudloff, B., Ulus, F.: Primal and dual approximation algorithms for convex vector optimization problems. Journal of Global Optimization 60(4), 713–736 (2014)

[17] Löhne, A., Weißing, B.: The vector linear program solver Bensolve -- notes on theoretical background, European Journal of Operational Research 260(3), 807-813 (2017)

[18] Löhne, A., Weißing, B.: BENSOLVE: A free VLP solver, version 2.0.0.alpha (2014)

[19] Pascoletti, A., Serafini, P.: Scalarizing vector optimization problems. Journal of Optimization Theory and Applications, 42(4), 499–524 (1984)

[20] Rockafellar, R.T.: Convex Analysis. Princeton University Press, Princeton (1970)

[21] Rockafellar, R.T., Wets, R. J-B.: Variational Analysis. Springer, Berlin (2009)

[22] Shao, L., Ehrgott, M.: Approximately solving multiobjective linear programmes in objective space and an application in radiotherapy treatment planning. Mathematical Methods of Operations Research 68(2), 257–276 (2008)

[23] Uçar, U. Ü., İşleyen, S. K., Demir, Y.: Üniversite Ders Çizelgeleme Probleminin Bulanik Ahp ve Çok Amaçli Karişik Tam Sayili Matematiksel Modelle Çözümü. Gazi University Journal of Science Part C 3(3): 513-523 (2015)