EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

(DOKTORA TEZİ)

KESİRLER CEBİRİ

VE

UYGULAMALARI ÜZERİNE

Necla KIRCALI GÜRSOY

Tez Danışmanı: Prof. Dr. Alev FIRAT

İkinci Danışman: Prof. Dr. Urfat NURİYEV

Matematik Anabilim Dalı

Bilim Dalı Kodu : 403.01.01
Sunuş Tarihi: 08.07.2015

Bornova - İZMİR

2015

Necla KIRCALI GÜRSOY tarafından doktora tezi olarak sunulan "**Kesirler Cebiri ve Uygulamaları Üzerine**" başlıklı bu çalışma E.Ü. Lisansüstü Eğitim ve Öğretim Yönetmeliği ile E.Ü. Fen Bilimleri Enstitüsü Eğitim ve Öğretim Yönergesi'nin ilgili hükümleri uyarınca tarafımızdan değerlendirilerek savunmaya değer bulunmuş ve 08.07.2015 tarihinde yapılan tez savunma sınavında aday oy birliği/oy çokluğu ile başarılı bulunmuştur.

Jüri Üyeleri:		**İmza**
Jüri Başkanı	**:**	**.....**
Raportör Üye	**:**	**.....**
Üye	**:**	**.....**
Üye	**:**	**.....**
Üye	**:**	**.....**

EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

ETİK KURALLARA UYGUNLUK BEYANI

E.Ü. Lisansüstü Eğitim ve Öğretim Yönetmeliğinin ilgili hükümleri uyarınca Doktora Tezi olarak sunduğum "Kesirler Cebiri ve Uygulamaları Üzerine" başlıklı bu tezin kendi çalışmam olduğunu, sunduğum tüm sonuç, doküman, bilgi ve belgeleri bizzat ve bu tez çalışması kapsamında elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara atıf yaptığımı ve bunları kaynaklar listesinde usulüne uygun olarak verdiğimi, tez çalışması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını, bu tezin herhangi bir bölümünü bu üniversite veya diğer bir üniversitede başka bir tez çalışması içinde sunmadığımı, bu tezin planlanmasından yazımına kadar bütün safhalarda bilimsel etik kurallarına uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul edeceğimi beyan ederim.

08 / 07 / 2015

İmzası

Necla KIRCALI GÜRSOY

ÖZET

KESİRLER CEBİRİ VE UYGULAMALARI ÜZERİNE

KIRCALI GÜRSOY, Necla

Doktora Tezi, Matematik Anabilim Dalı
Tez Danışmanı: Prof. Dr. Alev FIRAT
İkinci Danışmanı: Prof. Dr. Urfat NURİYEV
Temmuz 2015, 103 sayfa

Günümüzde Ekonomik ve Teknik sistemlerdeki birçok Karar (verme) Problemleri Kesir-Doğrusal Programlama (KDP) modelleri şeklinde gösterilebilir. KDP problemlerinin önemli bir kısmını ise Kesir Doğrusal Boole Programlama Problemleri (KDBP) oluşturmaktadır. Bu türlü problemlerin çözümü için kesirler üzerinde "pay-pay, payda-payda" prensibi ile yapılan işlemler (koordinatsal işlemler) gerekir.

Bu tezde, kesirler üzerinde "pay-pay, payda-payda" prensibi ile yapılan koordinatsal işlemler tanımlanmış ve bu işlemlerin cebirsel ve geometrik özellikleri incelenmiştir. Bazı temel eşitsizlikler ispatlanmıştır. İnversiyon özellikleri incelenmiştir. Farklı stratejilere göre toplamlar geliştirilerek sezgisel algoritmalar için bir temel oluşturulmuştur.

Optimizasyon problemlerinden Çanta Problemi NP-zor sınıftan olduğu için çözümünde farklı sezgisel algoritmalar kullanılır. Bu nedenle, 0/1 maksimizasyon sırt çantası problemi ve 0/1 minimizasyon sırt çantası problemi için önceden oluşturulmuş algoritmalar göz önünde bulundurularak, bu problemleri çözmek için tezin içerisinde geçen cebirsel alt yapıya uygun olacak şekilde yeni algoritmalar geliştirilmiştir.

Son olarak, bu problemlerin çözümü için geliştirilmiş algoritmaların verimliliğini kıyaslamak için test problemleri oluşturularak, KNAPSACK Test Problemleri Kütüphanesi hazırlanmıştır.

Anahtar sözcükler: Kesirler Cebiri, Kesir Doğrusal Programlama, Boole Programlama, 0/1 Sırt Çantası Problemi, Minimizasyon, Maksimizasyon, Sezgisel Algoritmalar, Yaklaşım Algoritmaları, Aç gözlü(greedy) algoritmalar.

ABSTRACT

ON ALGEBRA OF FRACTIONS AND APPLICATIONS

KIRCALI GÜRSOY, Necla

PhD in Mathematics
Supervisor: Prof. Dr. Alev FIRAT
Co-Supervisor: Prof. Dr. Urfat NURİYEV
July 2015, 103 pages

Nowadays, many decision problems in economical and technical systems can be shown as Linear Fractional Programming (LFP) models. A significant part of the LFP problems consists of Linear Fractional Boole Programming problems. In order to solve the problems as such, we need operations (coordinate operations) which work on fractions with “numerator-numerator, denominator- denominator” principle.

In this thesis, coordinate operations which work on fractions with “numerator - numerator, denominator - denominator” priciple have been defined and their geometric and algebraic properties have been examined. Some basic inequalities have been proved. Inversion properties have been examined. A foundation for the heuristic algorithms have been laid by developing sums according to different strategies.

Different heuristic algorithms are used in Knapsack problems’ solution since Knapsack problem is in NP-complete class. Therefore, in order to solve these problems, new algorithms in accordance with the algebraic substructure in this thesis have been developed by previously written algorithms for 0/1 maximization knapsack problem and 0/1 minimization knapsack problem have been taken into consideration.

Finally, some test problems have been created in order to compare the efficiency of these algorithms which had been developed for the solution for these Knapsack problems and then a KNAPSACK Test Problems Library has been created.

Keywords: Algebra of Fractions, Linear Fractional Programming, Boolean Programming, 0/1 Knapsack Problem, Minimization, Maximization, Heuristic Algorithms, Approximation Algorithms, Greedy Algorithms.

TEŞEKKÜR

Matematik ve Bilgisayar Bilimleri alanlarındaki geniş bilgi ve tecrübesiyle çalışmalarıma yön veren, danışmanlarım, değerli hocalarım Sayın Prof. Dr. Alev FIRAT ve Sayın Prof. Dr. Urfat NURİYEV'e en içten teşekkürlerimi sunarım.

Eğitimime her zaman destek vererek bugünlere gelmemde üzerimde sonsuz emekleri olan Babama ve Anneme saygı ve şükranlarımı sunar, varlığıyla destek olan sevgili kardeşime; hayatımın her anında yanımda olan, varlığını her zaman yanımda hissettiren can yoldaşım, sevgili eşim Arif GÜRSOY'a sonsuz teşekkürlerimi sunarım.

Canımın içi, hayatımın anlamı, güzel meleğim oğlum Ata Orbay'a da küçücük olmasına rağmen bu süre içerisinde bizden ayrı kalarak gösterdiği fedakarlıktan dolayı teşekkür ederim, umarım büyüdüğünde beni anlayacaktır.

İÇİNDEKİLER

Sayfa

İÇİNDEKİLER (devam)

Sayfa

İÇİNDEKİLER (devam)

Sayfa

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

Çizelge Sayfa

1 GİRİŞ

Günümüzde ekonomik ve teknik sistemlerdeki birçok Karar (verme) Problemleri Kesir-Doğrusal Programlama (KDP) modelleri şeklinde gösterilebilir (Bajalinov, 2003; Chernov and Lange, 1978; Karlin, 1992; Kulinkovich, et al., 1981). Bu türlü modellerde doğrusal kısıtlar altında iki doğrusal formun (fonksiyonun) oranlarının uç(ekstremal) değerlerinin bulunması istenir.

$$Enb(veya\ Enk)\ F(x) = \frac{P(x)}{D(x)} = \frac{\sum_{j=1}^{n} p_j x_j + p_0}{\sum_{j=1}^{n} d_j x_j + d_0} \tag{1.1}$$

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, \qquad i = 1, 2, ..., m_1 \tag{1.2}$$

$$\sum_{j=1}^{n} a_{ij} x_j \ge b_i, \qquad i = m_1 + 1,\ m_1 + 2, ..., m_2 \tag{1.3}$$

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \qquad i = m_2 + 1,\ m_2 + 2, ..., m \tag{1.4}$$

$$\mathrm{x}_j \ge 0, \qquad j = 1, 2, ..., n_1 \tag{1.5}$$

$$\mathrm{x}_j \in \{0,1\}, \qquad j = 1, 2, ..., n \tag{1.5'}$$

$m_1 \le m_2 \le m$ ve $n_1 \le n$ olmak üzere *F(x)* amaç fonksiyonu (1.1) yukarıdaki (1.2-1.5) kısıtlar altında maksimize (veya minimize) edilmek istenmektedir. Yukarıdaki kısıtlara göre tanımlanmış uygun çözümlerin kümesi *S* olarak gösterilir ve $\forall x = (\mathrm{x}_1, \mathrm{x}_2, ..., \mathrm{x}_n) \in \mathrm{S}$ için $D(\mathrm{x}) \ne 0$ kabul edilir (Bajalinov, 2003).

KDP problemlerinin önemli bir kısmını Kesir Doğrusal Boole Programlama Problemleri (KDBP) oluşturmaktadır. KDBP, (1.5) kısıtı yerine (1.5′) kısıtnın gelmesi ile oluşmaktadır.

KDBP modelleri büyük enerji ve üretim komplekslerinin planlanmasında, bilgisayar destekli yönetim sistemleri için bilgisayar sistemlerinin yapılarının modellenmesinde ve diğer önemli uygulama alanlarında kullanılmaktadır (Arora, et al., 1977; Bajalinov, 2003; Chandra and Chandramohan**,** 1980; Chernov, 1978; Gupta and Malhotra, 1995; Kulinkovich et al., 1981).

Bu tezde, Kesir Doğrusal Boole Programlama problemlerine yönelik yeni cebirsel işlemler tanımlanmış, bu işlemler yeni çözüm yöntemlerinin geliştirilmesi ve verimli algoritmaların tasarlanmasında kullanılmıştır.

Tez, sekiz bölüm ve kaynaklar listesinden oluşmaktadır.

Birinci bölümde, tezde üzerinde çalışılan problem grubunun modelinden ve öneminden bahsedilerek problemin uygulamadaki yerine değinilmiş ve tezde yapılanlar hakkında kısa bilgi verilmiştir.

Tezin ikinci bölümünde ilk olarak cebir kavramından bahsedilerek cebir örnekleri verilmiştir. Daha sonra optimizasyon problemlerinin genel tanımı ve modeline değinerek belirli kriterlere göre yapılan sınıflandırılmalardan bahsedilmiştir. Bu bölümde, optimizasyon problemlerinin karmaşıklık sınıfları, ayrık optimizasyon problemlerinin tanımı ve çözüm yöntemleri de ele alınmıştır. Ayrıca birinci bölümde matematiksel modeli verilen Kesir Doğrusal Boole Programlama Problemlerine gerçek hayattan örnekler verilmiştir.

Tezin üçüncü bölümünde, “pay-pay, payda-payda” prensibi ile yapılan üç yeni işlem tanımlanarak onların cebirsel ve geometrik özellikleri incelenmiştir. Bu bölümde tanımladığımız cebirsel işlemleri temel alan bazı eşitsizlikler ispatlanmıştır.

Tezin dördüncü bölümünde, bu işlemlerin inversiyon özellikleri incelenerek bazı teoremler ispatlanmıştır. Farklı stratejilere dayanan toplamlar tanımlanarak, bunların kıyaslanması yapılmıştır. Bu stratejilere dayanan bazı teorem ve önermeler ispatlanmıştır.

Tezin beşinci bölümünde, sırt çantası problemi tanımlanarak farklı kriterlere göre sınıflandırmaları anlatılmıştır. 0/1 değişkenli tek boyutlu maksimizasyon sırt çantası probleminin çözümü için sezgisel stratejiler araştırılmıştır. Bu stratejilere göre algoritmalar oluşturulmuş ve bu algoritmaların kombinasyonları incelenmiştir.

Tezin altıncı bölümünde, 0/1 sırt çantası probleminin maksimizasyon ve minimizasyon versiyonları ele alınmıştır. Sırt çantası probleminin bilinen garanti değerli algoritmalarına değinilerek 0/1 minimizasyon problemi için iki yeni algoritma önerilmiş ve ikinci algoritmanın garanti değerinin $\Delta = 2$ olduğu gösterilmiştir.

Tezin yedinci bölümünde, beşinci ve altıncı bölümlerde önerilen yeni algoritmalarla hesaplama denemelerinin yapılabilmesi için farklı özelliklere sahip 120 adet test problemi oluşturulmuştur. Bu algoritmaların bulduğu sonuçlar, WinQSB programı ile elde edilmiş optimum sonuç ile kıyaslanmıştır. Ayrıca bu problemler ve sonuçları içinde bulunduran KNAPSACK Test Problemleri Kütüphanesi hazırlanmıştır (http://fen.ege.edu.tr/~urfat/knapsack/). Bu problemler maksimizasyon ve minimizasyon olmak üzere iki başlık altında erişime açılmıştır.

Sonuç bölümünde ise, yapılan araştırmalar özetlenerek, elde edilen sonuçlar paylaşılmıştır.

2 ÖNBİLGİLER

2.1 Cebirler Üzerine

Tanım 2.1 (Hungerford, 1974) *C*, *F* cismi üzerinde bir vektör uzayı ve $.: C \times C \to C$ bir dönüşüm olsun. Eğer bu $.: C \times C \to C$ dönüşümü her $x, y, z \in C$ her $\alpha \in F$ için aşağıdaki şartları sağlıyor ise *C* ye (*F* üzerinde) bir *cebir* denir:

1. $\alpha(x.y) = (\alpha x).y = x.(\alpha y)$
2. $x.(y+z) = x.y + x.z$ ve $(x+y).z = x.z + y.z$
3. $(x.y).z = x.(y.z)$

Cebirin tanımında geçen *F* cisminin $\mathbb{R}$ veya $\mathbb{C}$ olması halinde *C*' ye sırası ile *reel veya kompleks cebir* denir. *C* bir cebir ve her $x, y \in C$ için $x.y = y.x$ ise *C'* ye değişmeli veya *komütatif cebir* denir. *C*' nin çarpma işlemine göre etkisiz elemanı varsa, yani her $x \in C$ için $x.e = e.x = x$ olacak şekilde bir $e \in C$ var ise C' ye *birimli cebir* ve *e*' ye de *birim eleman* denir.

Tanım 2.2 C bir cebir ve C' de bir $\|.\|$ normu tanımlanmış olsun. Yani kısaca $(C, \|.\|)$ bir normlu uzay ve aynı zamanda bir cebir olsun. Bu norm her $x, y \in C$ için $\|xy\| \leq \|x\|\|y\|$ şartını sağlıyorsa ve C' nin birim elemanlı olması halinde de $\|e\| = 1$ ise C' ye *normlu cebir* denir.

Tanım 2.3 (Whitesitt, 1962) (B,+,.) sistemi aşağıdaki şartları sağlıyor ise *Boole cebiri* denir:

1. Her $a, b \in B$ için $a+b = b+a$ ve $a.b = b.a$
2. B kümesinde, $+$ işlemi için 0, $.$ işlemi için 1 birim elemanları vardır
3. Her $a, b, c \in B$ için $a.(b+c) = a.b + a.c$ ve $a+(b.c) = (a+b)(a+c)$
4. Her $a \in B$ için $a + a' = 1$ ve $a.a' = 0$ olacak şekilde $a' \in B$ vardır.

2.2 Optimizasyon Problemleri

Optimizasyon problemleri, karar değişkenlerine bağlı en uygun çözümü araştırılan problemlerdir. Optimizasyon problemlerinde bir fonksiyonun (*amaç fonksiyonu*) maksimizasyonu veya minimizasyonu amaçlanır. Örneğin, bir turun maliyetinin minimizasyonu, bir projeden elde edilecek kazancın maksimizasyonu gibi amaçlar karar değişkenlerine bağlı fonksiyonlarla ifade edilir. Belirtilen karar değişkeni *x* ile gösterildiğinde amaç fonksiyonu $f(x)$ ile gösterilir (Cura, 2008). Genel bir optimizasyon problemi aşağıdaki gibi ifade edilebilir:

Amaç fonksiyonu:

$$Enb \ (\text{veya} \ Enk) \ f(x) \tag{2.1}$$

Kısıtlar:

$$g_i(x) \geq b_i, \qquad i = \overline{1,m} \tag{2.2}$$

$$h_j(x) = c_j, \qquad j = \overline{1,n} \tag{2.3}$$

$$p_k(x) \leq d_k, \qquad k = \overline{1,s} \tag{2.4}$$

$$x \in J \tag{2.5}$$

Bir problemi matematiksel olarak (2.1 – 2.5) ifadelerinde olduğu gibi tanımlayan soyut modellere *matematiksel model* denir. Bu matematiksel modelde, (2.2 – 2.4) modelin kısıtlarını, (2.5) modelin karar değişkeni *x*'i göstermektedir. *x* karar değişkeni herhangi bir *J* kümesinin elemanıdır. (2.2 – 2.5) kısıtları altında (2.1) ifadesi, modelin amaç fonksiyonunu göstermektedir (Cura, 2008).

Bir optimizasyon probleminin amaç ve kısıt fonksiyonları doğrusal fonksiyon ise bu probleme *doğrusal programlama problemi,* bu fonksiyonlardan herhangi biri doğrusal değil ise *doğrusal olmayan programlama problemi* denir. Karar değişkenleri negatif olmayan tamsayı değerler alan doğrusal programlama

problemine *tamsayılı programlama problemi*, aksi halde *tamsayılı olmayan programlama problemi* denir (Karaboğa, 2004).

Optimizasyon problemlerinin başka bir sınıflandırması ise *sürekli (continuous) optimizasyon* ve *ayrık (combinatorial) optimizasyon* problemleri şeklindedir (Nemhauser and Woolsey, 1998; Woolsey, 1998). Sürekli optimizasyonda karar değişkenleri gerçel sayılar kümesinde değer alabilir. Ayrık niceliklerin optimal olarak düzenlenmesi, gruplanması, sıralanması veya seçilmesi problemi *ayrık (discrete) optimizasyon problemi* olarak adlandırılır. Ayrık optimizasyonda karar değişkenleri önceden tanımlanmış değerler alabilirler. Bir *x* karar değişkeninin tam sayı değerler alması veya sadece sıfır veya bir değerler alması örnek olarak verilebilir (Karaboğa, 2004; Cura, 2008).

Kesir doğrusal programlama problemi'nde doğrusal kısıtlar altında iki doğrusal fonksiyonun oranlarının uçsal değerlerinin bulunması istenir. Kesir doğrusal programlama problemleri aşağıdaki gibi modellenebilir:

Amaç Fonksiyonu:

$$Enb(veya\ Enk)\ F(x) = \frac{P(x)}{D(x)} = \frac{\sum_{j=1}^{n} p_j x_j + p_0}{\sum_{j=1}^{n} d_j x_j + d_0} \tag{2.6}$$

Kısıtlar:

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, \qquad i = 1,2,....,m_1 \tag{2.7}$$

$$\sum_{j=1}^{n} a_{ij} x_j \ge b_i, \qquad i = m_1 + 1,\ m_1 + 2,....,m_2 \tag{2.8}$$

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \qquad i = m_2 + 1,\ m_2 + 2,....,m \tag{2.9}$$

$$x_j \geq 0 \qquad j = 1, 2, ..., n_1 \qquad (2.10)$$

$m_1 \leq m_2 \leq m$ ve $n_1 \leq n$ olmak üzere *F(x)* amaç fonksiyonu yukarıdaki kısıtlar altında maksimize (veya minimize) edilmek istenmektedir. Yukarıdaki kısıtlara göre tanımlanmış uygun çözümlerin kümesi *S* olarak gösterilir ve $\forall x = (x_1, x_2, ..., x_n) \in S$ için $D(x) \neq 0$ kabul edilir (Bajalinov, 2003).

Bilinmeyen değişkenlerin kümesi tamsayılardan oluşan kesir doğrusal programlama problemlerine *tamsayılı kesir doğrusal programlama problemleri* denir. *0-1 kesir doğrusal programlama problemleri* tamsayılı kesir doğrusal programlama problemlerinin bir alt sınıfıdır ve bilinmeyen değişkenlerin uygun kümesinin elemaları 0 veya 1 dir.

Bir optimizasyon problemi, optimizasyon versiyonu ve tanıma versiyonu olma üzere iki türlü ifade edilebilir. Bir problemin optimizasyon versiyonunda amaç fonksiyonunun en iyi değere sahip uygun çözümü aranmaktadır ve her uygun çözüm bir aday çözümdür. Tanıma versiyonunda ise, amaç fonksiyonunun verilen bir sayısal sınır *B*'den daha iyi maliyete sahip çözüm olup olmadığı aranmaktadır. Tanıma problemlerinin cevapları "evet" veya "hayır" dır. Örneğin, Gezgin Satıcı Probleminin optimizasyon versiyonunda herhangi tepe kombinasyonu bir uygun çözüm iken en kısa uzunluklu tur (tepe kombinasyonu) aranırken Tanıma versiyonunda verilen bir sayısal sınır *B*'den daha iyi maliyete sahip çözüm olup olmadığı aranmaktadır (Garey and Johnson, 1979; Cormen et al., 2001).

2.2.1 Problemlerin karmaşıklık sınıfları

Bir problemin algoritmik çözümlerinin sınıflandırılması, algoritmaların yürütülmesi için gerekli işlemlerin sayısı temel alınarak yapılabilir ve buna *algoritmik karmaşıklık* denir. Karmaşıklık, algoritmanın çalıştırılacağı bilgisayarın işlemcisinden bağımsız olarak değerlendirilmelidir (Nabiyev, 2011).

Hesaplama karmaşıklığı, bir algoritmanın ne kadar hızlı çalışacağı ve bilgisayarın belleğinde ne kadar yer kullanacağına dair bilgiler verir.

Algoritmanın girdi boyutu veya problem boyutu, girişi tanımlamak için gerekli sembollerin sayısı ile bağıntılıdır. Bir algoritmanın hesaplama karmaşıklığı, hesaplamayı yapmak için gerekli zamanı ölçen *zamansal karmaşıklık* değerlendirilmesi ve hesaplama için gerekli bellek alanının ölçümü olarak *yersel karmaşıklık* olmak üzere iki açıdan incelenmektedir. Genelde, zaman karmaşıklığı sadece karmaşıklık olarak isimlendirilmektedir (Nabiyev, 2011).

Çalışma zamanının üst sınırı olan *büyük O* (Big *O*) notasyonu, en kötü durumdaki çalışma zamanının belirlenmesinde kullanılır ve büyük *O* notasyonu bir fonksiyonun asimptotik üst sınırını belirtir (Nabiyev, 2011).

Örneğin, n^2-4n+8 fonksiyonunda n'nin büyük değerleri için n^2 'nin dışındaki toplananlar önemsenmez ve algoritma karmaşıklığı $c=1$ olmak üzere $c\cdot O\left(n^2\right)$, yani $O\left(n^2\right)$ şeklinde yazılır. Algoritma karmaşıklığı fonksiyonlarına örnek olarak, ikili arama için $O(\log n)$, doğrusal arama için $O(n)$, matris çarpımı işlemleri için $O\left(n^3\right)$, hızlı sıralama algoritması için $O(n\cdot\log n)$ fonksiyonları verilebilir. Bu problemlerin hepsinde $g(x)$ fonksiyonu polinomiyal karakter taşır (Nabiyev, 2011; Cormen et al., 2009).

2.2.1.1 P, NP, NP-tamlık, NP-zorluk

Bir algoritmanın zaman karmaşıklığı, belli boyutlardaki girdi değerleri için harcanan basit hesapsal adımların sayısını veren bir fonksiyondur. n girdi verisinin boyutu olmak üzere, bir algoritmanın çalışma zamanı üstten herhangi bir $P(n)$ polinomu ile sınırlı ise buna *polinom sınırlı algoritma* denir ve bu algoritmalarla çözülebilen tüm problemler *P sınıfı* olarak tanımlanır. P sınıfındaki problemler deterministik makinelerle çözülebilmektedir (Nabiyev, 2012).

Bir problemin karmaşıklığı polinomiyal biçimde tanımlanabilirse, bu problemin sonlu sayıda adımla polinomiyal zamanda çözülebileceği söylenebilir. Polinomiyal algoritmaları çalıştıran sanal modeller *deterministik Turing makineleridir* (DTM) ve P sınıfı, DTM ile polinomiyal zamanda çözülebilen problemler sınıfıdır. Polinomiyal zamanda deterministik olmayan Turing

makineleriyle (NDTM) çözülebilen problemler ise *NP* (Non-deterministic Polinomial) *sınıfını* oluşturur (Nabiyev, 2011; Garey and Johnson, 1979). P sınıfı NP sınıfının bir alt kümesi olarak düşünülebilir, $P \subset NP$. Çünkü NDTM, DTM ile çözülebilen tüm problemleri çözebilir (Cormen et al., 2009).

P sınıfı, polinomiyal zamanda çözülebilen problemlerden oluşmaktadır. Bu problemler, *k* bir sabit ve *n* problem girdisinin boyutu olmak üzere $O(n^k)$ zamanda çözülebilen problemlerdir. NP sınıfı, polinomiyal zamanda doğrulanabilen problemlerden oluşur. Yani, problemin herhangi bir çözüm sertifikası verildiğinde, problemin girdi boyutuna bağlı olarak bu sertifikanın polinomiyal zamanda doğrulanabildiği problemler sınıfıdır. Bu tür problemleri polinomiyal zamanda doğrulayabilen algoritmalara *deterministik olmayan algoritma* denir.

Bir Π probleminin *NP-tam sınıftan* olabilmesi için, öncelikle bu Π probleminin NP sınıftan olduğu ve sonrasında da NP sınıftaki her problemin polinomiyal zamanda bu probleme indirgenebileceği gösterilmelidir (Cormen et al., 2009):

1. $\Pi \in NP$ ve

2. $\forall \Pi' \in NP$ için $\Pi' \leq_p \Pi$.

Bu adımlardan sadece ikincisi sağlanırsa Π problemi *NP-zor sınıftandır* denir.

2.2.2 Ayrık optimizasyon problemlerinin çözüm yöntemleri

Ayrık optimizasyon problemlerine örnek olarak bazı sıralama, çizelgeleme, paketleme ve programlama türünden optimizasyon problemleri verilebilir. Bu problemlerin çözüm yöntemleri kesin ve yaklaşık çözüm yöntemleri olarak ikiye ayrılmaktadır. Kesin çözüm yöntemleri optimal çözümü garanti etmesine rağmen hesaplama zamanı ve bellek gereksinimi nedeniyle makul sınırlar içerisinde gerçekleştirilemezler. Yaklaşık çözüm yöntemleri ise optimal çözüme ulaşmasalar

da optimale makul sınırlarda yaklaşarak kısa zamanda çözüm üretebilirler (Horowitz and Sahni, 1978).

2.2.2.1 Kesin yöntemler

Bu sınıftaki bazı yöntemler aşağıdaki gibi sayılabilir:

- Sayımlama (Enumeration),
- Kesen düzlemler (Cutting planes),
- Dinamik programlama (Dynamic Programming),
- Dal-sınır (Branch and bound),

Sayımlama

Sayma metodu tüm olasılıkları saymaya dayanan bir yöntemdir. Bulunan tüm olasılıklar için bir amaç fonksiyonu değeri hesaplanır ve sonunda bunlar arasından en iyi olan çözüm seçilir. Bu yöntem az değişkene sahip problemler için kullanışlı olmasına karşın, değişken sayısı arttıkça incelenen olasılıklar üstel biçimde artar. Örneğin n değişkenli sırt çantası problemini ele alalım; burada incelenmesi gereken durum sayısı 2^n dir. Yani n değeri arttıkça işlem yoğunluğu artar ve çok büyük n değerleri için şu andaki mevcut bilgisayarlarla çözüm süresi yüz yıllar alabilir.

Kesme yöntemi

Kesme yöntemi tamsayılı doğrusal problemleri çözmek için ilk önerilmiş yöntemdir. Yöntemin şeması ilk defa Dantzig tarafından açıklanmış, sonrasında ise Gomory tarafından geliştirilmiştir.

Kesme yöntemi, tamsayılı programlama probleminin doğrusal programlama problemine gevşetilmesine dayanır. Yani tamsayılı bir problemdeki tüm kısıtlar olduğu gibi ele alınırken, değişkenlerin tamsayı olması kısıtı kaldırılır. Bu şekilde

doğrusal programlama metotlarından her hangi biri kullanılarak sonuç bulunur; bulunan sonuç tamsayı ise sorun yoktur; değilse tamsayı olmayan çözümleri dışarıda bırakacak ve aynı zamanda tüm tamsayı çözümleri koruyacak yeni bir kısıt (kesme) eklenir. Bu işlemlere optimal tamsayı çözüm bulunana dek devam edilir.

Kesme yöntemlerinin genelde düzensiz olduğu bilinmektedir; ayrıca tüm verilerin simpleks tablo şeklinde saklanması büyük boyutlu problemler için sorun yaratmaktadır.

Dinamik programlama

Kesin çözüm yöntemlerinin içinde dinamik programlama özel bir yere sahiptir. Bu yöntem, örtüşen alt problemler ve optimal alt yapı özelliklerini sergiler; dolayısıyla toplam işlem sayısını ve kullanılan zamanı azaltır. Dinamik programlama algoritmalarının karmaşıklığı teorik olarak oldukça kolay hesaplanabilir ve bu değer reel karmaşıklığa yakındır (Cormen et al., 2009). Ancak rekürsif bir yöntem olması dolayısıyla kısıt ve değişken sayısının çok olması durumunda ya da çok büyük katsayılar için etkili değildir.

Dal-sınır yöntemi

Dal-sınır algoritması ise optimal çözüme ulaşmak için sistemli bir şekilde uygun çözümleri saymaya dayanır. Ancak sayma yöntemindeki gibi tüm olasılıklar değil, olasılıkların daha küçük bir kısmı incelenir, ümit vermeyen bazı olasılıklar sayma aşamasında kesip atılır. Dal-sınır algoritmasının avantajı hızlı olması, verilmiş her hangi bir tamsayılı problemin özelliklerinin göz önünde bulundurulması ve yöntemin içinde başka tamsayılı metotların kullanılmasına imkan vermesidir. Hesaplama sürecinde ortaya çıkan çözümler genellikle iyi yaklaşık çözümlerdir. Ancak bir dezavantajı bilgisayar belleğine daha çok gerek duymasıdır. Ayrıca değişken sayısının artması hesaplama süresini üstel olarak artırır ve bulunan çözümün optimal çözüm olduğunu ispatlamak çalışma zamanından daha fazla zaman gerektirebilir (Papadimitriou and Steiglitz, 1982).

2.2.2.2 Yaklaşık yöntemler

Yaklaşık yöntemler, optimizasyonda kesin yöntemlerin tersine optimal çözümü garanti etmezler. Bununla birlikte;

- Verilen problemin kesin çözüm yönteminin olmadığı durumlarda tek seçenek yaklaşık çözümlerdir.

- Uzun zamanda bulunan kesin çözümdense kısa zamanda bulunan yaklaşık çözüm daha değerlidir.

- Bilinen kesin çözüm yöntemleri yeteri kadar iyi değil; çünkü çoğu problem çözümü için zorluklarla karşılaşıyor ise,

- Kesin çözüm yönteminin getirdiği yoğun hesaplama ve çözüm zamanı söz konusu ise yaklaşık yöntemler tercih edilebilir.

Sezgiseller

Bilgisayar bilimlerinde sezgisel algoritma, ele alınan problemin özelliğine göre tasarlanmış bir algoritmadır ve çözümün doğruluğunun kanıtlanabilir olup olmadığını göz ardı eder. Ancak genellikle en iyi mümkün çözüme makul ölçüde yaklaşarak oldukça hızlı çözümler üretir. Sezgiseller en kötü durum düşünüldüğünde çok başarısız performans gösterebilirler; hatta kötü bir problem örneği seçildiğinde optimumdan çok uzak bir sonuç döndürebilir ve/veya üstel çalışma zamanı gerektirebilir. Bununla beraber iyi sezgiseller bir problemin çoğu örneğinde en iyi yaklaşım algoritmalarının performansını geride bırakabilirler (Ausiello et al., 1999).

Literatürde yer alan sezgisellerin çoğu ya yinelemeli olarak tek bir uygun çözüm inşa eden "*yapıcı sezgisel*" ya da uygun çözümlerin bir kümesini inceleyip en iyi sonucu döndüren "*sayma sezgiseli*" dir. Yapıcı sezgiseller güçlü olarak problem bağımlıdırlar ve genellikle polinom zaman gerektirirler. Diğer yandan sayma sezgiselleri incelenen küme çok büyük olduğunda üstel çalışma zamanı gerektirebilirler.

Greedy algoritmalar, Yerel arama (Local search), Genetik algoritmalar, Tabu araması (Tabu search) ve Karınca kolonisi (Ant colony) en iyi bililen bazı sezgisellerdir.

Tezde Greedy algoritma ve Yerel arama yöntemleri kullanıldığı için aşağıda bu sezgiseller detaylı olarak anlatılmıştır.

i) Greedy algoritmalar

Sezgisel bir yöntem olan Greedy türündeki algoritmalar tasarlama kolaylığı ve optimum çözüme sıkça iyi yaklaşımlar vermesi bakımından oldukça kullanışlıdır. Eğer verilen bir problem sınıfı için tasarlanan greedy algoritmasının optimal değeri ürettiği ispatlanabilirse daha hızlı olan greedy türündeki bu algoritmanın diğer optimizasyon yöntemlerine tercih edilmesi kaçınılmazdır. Greedy algoritmalar her zaman olmasa da bazı problemler için optimal çözümü verirler (Cormen et al., 2009). Bu yöntem oldukça güçlü bir yöntem olup problemlerin geniş bir sınıfı için iyi çalışırlar. Örneğin, Dijkstra' nın en kısa yol algoritması, Chvátal' ın küme örtüsü sezgiseli ve Kruskal algoritması iyi bilinen greedy algoritmalarından bazılarıdır.

Genel olarak greedy algoritmalarının özellikleri aşağıdaki şekilde verilebilir:

- Bir seferde tek bir karar verir.

- Karar verirken yerel bilgiyi kullanır.

- Kararı verirken o an için en fazla faydayı almaya bakar; o nedenle açgözlü olarak adlandırılır. Başka bir deyişle global optimal çözüme götürebilir umuduyla yerel optimal seçimler yapar (Cormen et al., 2009).

- Açgözlülüğü hızlı karar vermeyi gerektir, herhangi bir adımda çözüm kümesinde elde edilen bir değer takip eden adımlarda değiştirilmez. Yani greedy algoritmaları asla seçimlerini tekrardan düşünmez.

Greedy yöntemi başlangıç olarak nesneleri bazı kriterlere göre sıraya koyar ve boş kümeden başlayarak çözüm kümesini genişletir. Yani yukarıda verilen ilk özellik kullanılır: Tek seferde tek nesne için karar verilir. Eğer sondaki çözüm uygun oluyorsa nesne o anki geçerli çözüme eklenir; aksi halde bir daha düşünülmemek üzere elenir.

Geedy algoritmasının çalışma zamanı başlangıçta n tane nesnenin sıralanması ve n tane uygunluk testi nedeniyle $O(n)+O(n\log n)=O(n\log n)$' dir (Martello and Toth, 1990). Ayrıca yaklaşık çözümün kalitesi başlangıçtaki sıralamaya bağlıdır. Açıktır ki her problem örneği için her zaman optimal bir sıralama vardır; ancak hesaplama açısından zor bir problemin tüm örnekleri için böyle bir sıralamayı polinom zamanda bulmak pek olası değildir. Bununla birlikte bazı durumlarda, greedy yönteminin iyi yaklaşık çözümler bulmasını sağlayacak basit sıralamalar bulunabilir (Ausiello et al., 1999).

Greedy algoritmalardan yararlanarak Çanta Problemleri için garanti değerli algoritmalar oluşturulabilir (Kellerer et al., 2004). Büyük boyutlu problemlerin yaklaşık çözümü için Balas ve Zemel (1980) tarafından karmaşıklığı $O(n)$ olan bir yöntem geliştirilmiştir.

ii) Yerel arama

Yerel arama algoritmaları bazı diğer algoritmalar tarafından bulunmuş bir başlangıç çözümle çalışmaya başlar ve daha iyi bir komşu çözüme taşınma yoluyla yinelemeli olarak geçerli çözümü geliştirir. Çözümü daha fazla geliştirebilecek hiç bir çözüm bulunamadığı takdirde algoritma bir yerel optimumda sonlanır (Aarts and Lenstra, 1997).

Özel bir problem için bir yerel arama algoritması elde etmek için başlangıç uygun çözümü bulan bir algoritma ve bir komşuluk yapısına ihtiyaç vardır (Her hangi bir y uygun çözümü için y'nin tüm komşularını bulmak gerekmez; ancak tek bir tane iyileştirici komşu bulunması yeterlidir.). Açıktır ki NP-zor problemler için optimal çözüme ulaşmayı sağlayan bir komşuluk yapısı polinom zamanda

bulunamaz; bu durumda yaklaşık çözüm üreten yerel arama algoritmaları araştırılır.

Yaklaşım algoritmaları

Önemli uygulama alanlarında karşılaşılanlar da dahil olmak üzere çoğu tamsayılı optimizasyon problemi NP-zor problemdir. Dolayısıyla $P \neq NP$ olduğu sürece bu problemler için kesin çözüm aramak sadece zaman kaybıdır. Bu nedenle de polinom zamanlı algoritmalar kullanarak yaklaşık çözümler üretmek bilgisayar bilimleri ve matematiğin ilgi çekici konularından biri haline gelmiştir. NP-zor problemler kesin çözümler bulmak için elverişli değillerse de yakın optimal çözümler bulunabilir. Dolayısıyla yaklaşım algoritmalarının tasarımı aslında kesin çözüm veren algoritmaların tasarım işleminden pek de farklı değildir (Vazirani, 2001).

Tanım 2.2.2.1 Bir $\varepsilon - yaklaşık\ algoritması,\ Z^*$ optimal değeriyle verilen bir maksimizasyon [minimizasyon] probleminin her bir örneği için $Z^*(1-\varepsilon) \leq Z$ [minimizasyon için $Z \leq (1+\varepsilon)Z^*$] olacak şekilde bir Z çözüm değeri veren algoritmadır.

Bir yaklaşım algoritması için akla gelen en önemli sorulardan biri problemin doğru yaklaşılabilirliği; yani polinom zamanda ulaşılabilen en iyi performans oranıdır. Çeşitli problemler için yaklaşım algoritmaları, oran kalitesi ya da en kötü durum hata sınırı bakımından birbirlerinden oldukça farklıdır. Eğer bir algoritma her hangi bir ε sabitine göre $\varepsilon -$ yaklaşık oluyorsa bu algoritma iyi olarak düşünülür. Yalnız karşılaşılan bir zorluk, elde edilmiş bir yaklaşım sonucunun daha da geliştirilip geliştirilemeyeceğinin belirlenmesi ve geliştirilirse ne boyutta olduğunu belirlemektir (Hochbaum, 1997).

Performans analizi ve garanti değer

Bir problem için tasarlanan bir algoritmanın çözüme ne kadar yaklaştığı, ne kadar çalışma zamanı ve belleğe ihtiyaç duyduğu bazı analizler sonucu belirlenir. Optimal çözüm üreten her algoritmanın performansı aynı derecede iyi olmadığı

açıktır. Bununla birlikte sadece yaklaşık sonuç veren bir algoritma daha iyi bir hesaplama zamanıyla bu açığını kapatmalıdır.

Temel olarak, algoritmaları nitelendirmenin üç yolu vardır. İlk yaklaşım, algoritmayı farklı problem örneklerine uygulayıp sonuçları karşılaştırmaktır. Bu yöntemde yazılım, donanım ve uygulama kalitesi gibi faktörlerin çalışmayı etkilememesi için çok fazla özen gösterilmelidir. Ayrıca uygun test problemlerinin seçimi de büyük bir sorundur. İkinci yol, algoritmanın ortalama çalışma zamanını araştırmaktır. Burada gerçek hayattan örnekleri yansıtan uygun bir olasılıksal model bulmak başlı başına özveri gerektirir. Karşılaşılan teknik güçlüklerin yanı sıra sonuçların anlamlı olabilmesinin problemin boyutunun yeterince büyük olmasıyla ya da algoritmanın yeterince çok sayıda problem örneğine uygulanmasıyla bağlantılı olması da bir dezavantajdır.

Bir algoritmanın performansını değerlendirmede kullanılan üçüncü ve en yaygın olan yöntem çalışma zamanının *en kötü durum* analizidir. Bu yöntemde verilen bir problem ailesi için algoritmanın bulduğu çözümün en kötü durumda optimalden en fazla ne kadar farklı olduğu belirlenir (Fisher, 1980). Burada problemin giriş parametrelerinden bağımsız öyle bir Δ değeri bulunur ki verilmiş ailedeki her bir bireysel maksimizasyon [minimizasyon] problemi için *f,* algoritmanın bulduğu çözüm değeri; *R,* optimal çözüm değeri olmak üzere $\Delta \leq f/R$ [$\Delta \geq f/R$] olur. Δ değeri genellikle algoritmanın *garanti değeri* olarak adlandırılır.

2.3 Bazı Kesir Doğrusal Programlama Problemleri

İnsan faaliyetlerinin çeşitli kollarında ve özellikle ekonomi için doğrusal programlama uygulamaları iyi bilinirken Kesir doğrusal programlamanın uygulamaları daha az bilinir. Ancak bütün gerçek hayat problemlerini doğrusal modelleme ile tanımlamak mümkün olamayabilir. Aşağıda bazı gerçek hayat problemlerinin kesir doğrusal programlama ile modellenmesi verilecektir.

2.3.1 Temel ekonomi problemi

Temel Ekonomi Problemi, bir fabrikada üretim esnasında toplam karı maksimum, toplam maliyeti de minimum olacak şekilde hangi üründen ne kadar üreteceğine karar (verme) problemidir (Bajalinov, 2003).

Temel Ekonomi Probleminin tanımlanması için aşağıdaki gösterimler kullanılmaktadır: Bir fabrikada *n* farklı ürün imal edilmektedir.

p_j , j. ürünün birim başına düşen karı;

p_0 , hacimden bağımsız olarak büyüklüğün sabit karı;

d_j , j. ürünün birim başına düşen maliyeti,

d_0 , fabrikada bir şey üretilmezse dahi ödenmesi gereken ve fabrikanın üretim işleminden bağımsız sabit masrafı;

b_i , fabrikanın mevcut *i.* kaynağının hacmi;

a_{ij} , j. ürünün imalat için birim başına düşen *i.* kaynağın masraf kotası;

x_j, $j = 1,2,...,n$ olmak üzere *j.* ürünün bilinmeyen üretim hacmi (karar değişkeni) olsun.

Bu durumda Temel Ekonomi Problemi aşağıdaki gibi tanımlanabilir:

Toplam kâr

$$P(\mathrm{x}) = \sum_{j=1}^{n} p_j x_j + p_0 \qquad (2.3.1)$$

Üretimin Toplam Maliyeti

$$D(\mathrm{x}) = \sum_{j=1}^{n} d_j x_j + d_0 \qquad (2.3.2)$$

Amaç Fonksiyonu:

$$Q(\mathrm{x}) = \frac{P(\mathrm{x})}{D(\mathrm{x})} = \frac{\sum_{j=1}^{n} p_j x_j + p_0}{\sum_{j=1}^{n} d_j x_j + d_0} \rightarrow max \qquad (2.3.3)$$

Fabrika, *toplam kar/toplam maliyet* oranını yani amaç fonksiyonunu maksimum yapmak için aşağıdaki kısıtlar altında her bir *j* ürününden kaç tane üreteceğine karar vermelidir.

Kısıtlar:

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, \qquad i = 1, 2, ..., m \qquad (2.3.4)$$

$$x_j \ge 0\,, \qquad j = 1, 2, ..., n \qquad (2.3.5)$$

Burada, (2.3.4) kısıtı *i*. kaynak için *j*. üründen kaynağın hacmini aşmayacak şekilde üretim yapılabilmesini, (2.3.5) kısıtı ise *j*. ürünün üretim hacminin negatif olamayacağını kontrol eder.

2.3.2 Deniz ulaşım problemi

Deniz Ulaşım Probleminde, A limanından *n* tip malı B limanına taşıyacak olan C kapasiteli gemileri, yükleyeceğimizi kabul edelim. Problem, ulaşım maliyetinin birim değeri başına kazanabilecek kârı maksimum yapmak üzere, her tipte maldan ne kadar yükleyeceğimizi belirlemektir (Bajalinov, 2003).

Deniz Ulaşım Probleminin tanımlanması için aşağıdaki gösterimler kullanılacaktır. $j = 1, 2, ..., n$ olmak üzere,

U_j, j. ürünün mevcut maksimum miktarını,

p_j, j. ürünün birim başına kazanılacak karını;

d_j, j. ürünün ulaşım maliyetini,

w_j, j. ürünün birim ağırlığını,

x_j, j. ürünün yüklenecek miktarını göstersin.

Bu durumda Deniz Ulaşım Problemi aşağıdaki gibi tanımlanabilir:

Amaç Fonksiyonu:

$$\frac{toplam\ kar}{toplam\ maliyet} = \frac{\sum_{j=1}^{n} p_j x_j}{\sum_{j=1}^{n} d_j x_j} \rightarrow max \qquad (2.3.6)$$

Kısıtlar:

$$\sum_{j=1}^{n} w_j x_j \leq C \qquad (2.3.7)$$

$$0 \leq x_j \leq U_j, \qquad j = 1, 2, ..., n \qquad (2.3.8)$$

Burada, (2.3.7) kısıtı gemi kapasitesini aşmayacak şekilde j. üründen yükleme yapılmasını, (2.3.8) kısıtı ise j. üründen en çok mevcut maksimum miktarı kadar yüklenebileceğini kontrol eder.

2.3.3 Sırt çantalı gezgin satıcı problemi

Gezgin satıcı problemi mevcut şehirlere birer kez uğrayıp tekrar başlangıç şehrine en kısa yoldan ulaşmayı hedefleyen bir problemken sırt çantası problemi toplam değeri maksimum yapmak için mevcut ürünlerden hangilerinin sınırlı kapasiteli sırt çantasına alınacağı ile ilgilenmektedir. Bu problemde, farklı ürün tipleri mevcut şehirlerde bulunabilmektedir. Ancak farklı şehirlerden alınan aynı ürün farklı faydalar sağlayabilmektedir. Gezgin satıcının sırt çantası kapasitesi yolculuk boyunca sabittir. Satıcı uğradığı her şehirden bazı ürünler almakta ve yolculuğa başladığı şehirde hepsini birden faydaya (örneğin kâra) dönüştürmektedir. Bu varsayımlar altında problem, uzaklık birimi başına düşen fayda miktarını (örneğin TL/km) maksimize edecek şekilde rotanın belirlenmesi ve hangi tip üründen hangi şehirden kaçar tane alınacağına ve satılacağına karar verilmesi problemidir. Kurulan model, doğrusal olmayan karma tamsayılı programlama modelidir (Çetin vd., 2007).

Sırt Çantalı Gezgin Satıcı Probleminin tanımlanması için aşağıdaki gösterimler kullanılmaktadır:

$i, j = 1, 2, ..., N$, şehirlerin indeksini,

$k = 1, 2, ..., \mathrm{K}$, şehirlerde bulunabilen ürünlerin indeksini,

c_{ij}, i şehrinden j şehrine olan uzaklığı,

w_k, k tipi ürün tarafından kullanılan mevcut kaynak miktarını (örneğin ürünün birim ağırlığını),

W , gezgin satıcının sahip olduğu mevcut kaynağı (örneğin taşıyabileceği ağırlık), W_i, i şehrinde kullanılması gereken en az kaynak miktarını, f_{ik}, k tipi ürünün i şehrinden alınmasıyla başlangıç şehrinde elde edilecek birim fayda (örneğin birim kar), y_{ik}, i şehrinden alınması gereken k tipi ürün sayısı

(tamsayı), u_i alt turların oluşmasını engelleyen ve turların uygun olmasında kullanılan ekstra değişkenleri ve karar değişkenleri de

$$x_{ij} = \begin{cases} 1, \; i \; şehrinden \; j \; şehrine \; gidilecekse \\ 0, \; aksi \; halde \end{cases}$$

olmak üzere Sırt Çantalı Gezgin Satıcı Problemi aşağıdaki gibi modellenir:

Amaç Fonksiyonu:

$$\frac{\sum_{i=1}^{N}\sum_{k=1}^{K} f_{ik} y_{ik}}{\sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij} x_{ij}} \rightarrow max \qquad (2.3.9)$$

Kısıtlar:

$$\sum_{i=1}^{N} x_{ij} = 1, \qquad j = 1,2,...,N \qquad (2.3.10)$$

$$\sum_{j=1}^{N} x_{ij} = 1, \qquad i = 1,2,...,N \qquad (2.3.11)$$

$$u_i - u_j + N x_{ij} \leq N - 1, \;\; i \neq j, \;\; i, j = 2,3,...,\mathrm{N} \qquad (2.3.12)$$

$$\sum_{k=1}^{K} w_k \sum_{i=1}^{N} y_{ik} \leq W, \qquad i = 1,...,N \qquad (2.3.13)$$

$$\sum_{k=1}^{k} w_k y_{ik} \geq W_i, \qquad i = 1,2,...,N \qquad (2.3.14)$$

$$x_{ij} \in \{0,1\}, \qquad (2.3.15)$$

$$y_{ik} \geq 0 \;\; ve \; tamsayı, \qquad (2.3.16)$$

$$u_i \geq 0, \qquad i = 1,...,N \tag{2.3.17}$$

Burada, amaç fonksiyonu belirsiz duruma düşmemektedir çünkü her şehirden alınması gereken bazı ürünler bulunmaktadır. Ayrıca, c_{ii} değerleri diğer uzaklıklara göreceli olarak çok büyük bir M değerine eşitlenerek, aynı şehirden aynı şehire gitmek yasaklanmıştır. Matematik programlama modeli, tüm şehirlere sadece birer kez uğrayıp gerekli ürünleri kapasite dahilinde alıp yolculuğa başladığı şehirde tüm ürünleri faydaya dönüştürme varsayımı altında geliştirilmiştir. Böylece, uzaklık birimi (örneğin km) başına düşen fayda miktarı (örneğin kâr) amaç fonksiyonu (2.3.9) ile maksimize edilmektedir. Amaç fonksiyonunun pay kısmı şehirlerden alınıp yolculuğa başlangıç şehrinde satılan ürünlerden elde edilen faydayken, payda kısmı da gezgin satıcının kat ettiği toplam mesafeyi vermektedir. Amaç fonksiyonunun maksimize edilmesi için, toplam faydanın mümkün olduğunca büyüklenmesi ve toplam mesafenin de mümkün olduğunca küçüklenmesi gerekmektedir. (2.3.10) kısıtları ile her şehre bir kez gelinmesi garantilenirken (2.3.11) kısıtları ile de her şehirden bir kez ayrılınması garantilenmektedir. (2.3.12) kısıtları, x_{ij} değişkenlerinin oluşturduğu herhangi bir *alt turun* uygun olmamasını ve söz konusu değişkenlerin oluşturduğu tam bir *tur*un da uygun olmasını gerektirmektedir. Kısıt (2.3.13) ile gezgin satıcının sırt çantasına alacağı ürünlerin sırt çantasının kapasitesini aşmaması gerektiği sağlanmaktadır. (2.3.14) kısıtları ile her şehirde sırt çantası kapasitesinin bazı miktarlarının kullanılması gerektiği vurgulanmaktadır. (2.3.15) kısıtı, ilgili karar değişkenlerinin ikili sayılar olduğunu, (2.3.16) kısıtı bu değişkenlerin negatif olmayan tamsayı ve (2.3.17) kısıtı da sürekli değişkenler olmaları gerektiğini belirtmektedir.

3 KESİRLER CEBİRİ VE ÖZELLİKLERİ

3.1 Kesirler Cebiri

Tezin giriş kısmındada bahsettğimiz gibi KDBP modelleri günümüzdeki bir çok problemlemin matematiksel modelidir. Tezin bu bölümünde, KDBP'nin çözümünde kullanılan ve "pay-pay, payda-payda" prensibi ile yapılan üç yeni işlem tanımlanarak bu işlemlere ait özelliklere ve ispatlara yer verilmiştir.

Tanım 3.1.1 $[-\infty,+\infty]$ genişletilmiş reel sayılar kümesi olmak üzere, $\Theta=\left\{ f \mid f=\frac{a}{b};\ a, b\in\mathbb{R} \right\}$ kümesi üzerinde $\oplus, \otimes$ sembolleri ile gösterilen iki ikili işlem ve $\odot$ sembolü ile gösterilen birli işlem aşağıdaki gibi tanımlanmıştır:

$\lambda\in\mathbb{R}$ ve $f_1=\frac{a_1}{b_1},\ f_2=\frac{a_2}{b_2}\in\Theta$ olmak üzere

$$f_1\oplus f_2=\frac{a_1}{b_1}\oplus\frac{a_2}{b_2}=\frac{a_1+a_2}{b_1+b_2}$$

$$f_1\otimes f_2=\frac{a_1}{b_1}\otimes\frac{a_2}{b_2}=\frac{a_1a_2}{b_1b_2}$$

$$\lambda\odot f_1=\lambda\odot\frac{a_1}{b_1}=\frac{\lambda a_1}{\lambda b_1}.$$

Not 3.1.1 Nikitin and Nuriyev (1982), $\oplus, \odot$ işlemlerini $\mathbb{Z}^+$ kümesi üzerinde aşağıdaki gibi tanımlamışlardır:

$$f_1\oplus f_2=\frac{a_1}{b_1}\oplus\frac{a_2}{b_2}=\frac{a_1+a_2}{b_1+b_2}$$

$$\lambda\odot f_1=\lambda\odot\frac{a_1}{b_1}=\frac{\lambda a_1}{\lambda b_1}$$

Teorem 3.1.1 $(\Theta,\oplus)$ sistemi bir abel gruptur.

İspat Θ nın her $f_1=\frac{a_1}{b_1},\ f_2=\frac{a_2}{b_2},\ f_3=\frac{a_3}{b_3}$ elemanları için,

$$\begin{aligned}(f_1\oplus f_2)\oplus f_3&=\left(\frac{a_1}{b_1}\oplus\frac{a_2}{b_2}\right)\oplus\frac{a_3}{b_3}=\left(\frac{a_1+a_2}{b_1+b_2}\right)\oplus\frac{a_3}{b_3}\\&=\frac{(a_1+a_2)+a_3}{(b_1+b_2)+b_3}=\frac{a_1+(a_2+a_3)}{b_1+(b_2+b_3)}=\frac{a_1}{b_1}\oplus\frac{(a_2+a_3)}{(b_2+b_3)}\\&=\frac{a_1}{b_1}\oplus\left(\frac{a_2}{b_2}\oplus\frac{a_3}{b_3}\right)=f_1\oplus(f_2\oplus f_3)\end{aligned}$$

olur. Buradan $\oplus$ işleminin birleşmeli olduğu görülür. Θ nın her $f=\frac{a}{b}$ elemanı için, $\frac{a}{b}\oplus\frac{0}{0}=\frac{a+0}{b+0}=\frac{a}{b}=\frac{0+a}{0+b}=\frac{0}{0}\oplus\frac{a}{b}$ olur. Buradan Θ kümesinin birim elemanın $e_\Theta=\frac{0}{0}$ olduğu elde edilir. Θ nın her $f=\frac{a}{b}$ elemanı için, $\frac{a}{b}\oplus\frac{x}{y}=\frac{0}{0}$ olduğunu kabul edelim. Bu durumda $\frac{a+x}{b+y}=\frac{0}{0}$ elde edilir. Buradan, $a+x=0$ ve $b+y=0$ olup $x=-a$ ve $y=-b$ bulunur. Θ nın her f elemanı için $f=\frac{a}{b}$ nin tersi $-f=-\left(\frac{a}{b}\right)=\frac{-a}{-b}$ olarak elde edilir. Ayrıca $\odot$ nın tanımından $\frac{-a}{-b}=\frac{(-1)a}{(-1)b}=(-1)\odot\frac{a}{b}$ olduğu görülür Θ nın her $f_1=\frac{a_1}{b_1},\ f_2=\frac{a_2}{b_2}$ elemanları için, $f_1\oplus f_2=\frac{a_1}{b_1}\oplus\frac{a_2}{b_2}=\frac{a_1+a_2}{b_1+b_2}=\frac{a_2+a_1}{b_2+b_1}=\frac{a_2}{b_2}\oplus\frac{a_1}{b_1}=f_2\oplus f_1$ olur.

Sonuç olarak, $(\Theta,\oplus)$ sistemi bir abel gruptur. ■

Not 3.1.2 Her $f,\ g\in\Theta$ için $f\oplus(-g)$ yerine $f\ominus g$ alacağız.

Teorem 3.1.2 $(\Theta,\oplus,\otimes)$ sistemi bir halkadır.

İspat Teorem 3.1.1 den $(\Theta,\oplus)$ sistemi bir abel grup olduğunu biliyoruz. Θ nın her $f_1=\frac{a_1}{b_1},\ f_2=\frac{a_2}{b_2},\ f_3=\frac{a_3}{b_3}$ elemanları için,

$$(f_1\otimes f_2)\otimes f_3=\left(\frac{a_1}{b_1}\otimes\frac{a_2}{b_2}\right)\otimes\frac{a_3}{b_3}=\left(\frac{a_1a_2}{b_1b_2}\right)\otimes\frac{a_3}{b_3}=\frac{(a_1a_2)a_3}{(b_1b_2)b_3}=\frac{a_1(a_2a_3)}{b_1(b_2b_3)}=\frac{a_1}{b_1}\otimes\left(\frac{a_2a_3}{b_2b_3}\right)$$
$$=\frac{a_1}{b_1}\otimes\left(\frac{a_2}{b_2}\otimes\frac{a_3}{b_3}\right)=f_1\otimes(f_2\otimes f_3)$$

olur. Buradan Θ kümesi üzerinde $\otimes$ işlemi birleşmelidir. Θ nın her $f_1=\frac{a_1}{b_1},\ f_2=\frac{a_2}{b_2},\ f_3=\frac{a_3}{b_3}$ elemanları için,

$$(f_1\oplus f_2)\otimes f_3=\left(\frac{a_1}{b_1}\oplus\frac{a_2}{b_2}\right)\otimes\frac{a_3}{b_3}=\left(\frac{a_1+a_2}{b_1+b_2}\right)\otimes\frac{a_3}{b_3}=\frac{(a_1+a_2)a_3}{(b_1+b_2)b_3}=\frac{(a_1a_3)+(a_2a_3)}{(b_1b_3)+(b_2b_3)}$$
$$=\left(\frac{a_1a_3}{b_1b_3}\right)\oplus\left(\frac{a_2a_3}{b_2b_3}\right)=\left(\frac{a_1}{b_1}\otimes\frac{a_3}{b_3}\right)\oplus\left(\frac{a_2}{b_2}\otimes\frac{a_3}{b_3}\right)=(f_1\otimes f_3)\oplus(f_2\otimes f_3)$$

olur. Böylece $\otimes$ işleminin $\oplus$ işlemi üzerine sağdan dağılma özelliğinin olduğu görülür. Ayrıca $(\mathbb{R},\cdot)$ değişmeli olduğundan $\otimes$ işleminin $\oplus$ işlemi üzerine soldan dağılma özelliği de sağlanır.

Sonuç olarak, $(\Theta,\oplus,\otimes)$ sistemi bir halkadır. ■

Tanım 3.1.2 Θ kümesinin elemanlarının değerine bu kesrin *modülü* denir ve $f\in\Theta$ için f nin modülü $|f|$ şeklinde gösterilir. k bir doğal sayı olmak üzere $\frac{kp}{kq}$ sayısı $\frac{p}{q}$ ile aynı sayıyı gösterir. Örneğin, $1/3,\ 2/6,\ 3/9\ldots$ sayıları aynı değeri, $1/3$ ü temsil ederler (Sergeyev, 2003). Yani $\left|\frac{1}{3}\right|=\left|\frac{2}{6}\right|=\left|\frac{3}{9}\right|=\ldots=0,\bar{3}$.

Not 3.1.3 $f=\frac{m}{n}\in\Theta$ olsun. O zaman f nin modülü, $|f|=\begin{cases}+\infty, & m>0,\ n=0\\ -\infty, & m<0,\ n=0\end{cases}$ olarak tanımlıdır. Ayrıca $m=0=n$ ise f nin modülü tanımsızdır.

Tanım 3.1.3 Θ kümesinin her f, g elemanı için $f\sim g\Leftrightarrow|f|=|g|$ ile tanımlanan "$\sim$" bağıntısı Θ kümesi üzerinde bir denklik bağıntısıdır.

İspat Θ kümesinin her f elemanı için $|f|=|f|$ olduğundan "$\sim$" bağıntısı yansımalıdır. Θ kümesinin her f, g elemanları için $|f|=|g|$ ise $|g|=|f|$ olduğundan $g\sim f$ elde edilir. Dolayısıyla "$\sim$" bağıntısı simetriktir. Θ kümesinin her f, g, h elemanları için $f\sim g$ iken $|f|=|g|$ ve $g\sim h$ iken $|g|=|h|$ olduğundan $|f|=|g|=|h|$ bulunur. Buradan $|f|=|h|$ yani $f\sim h$ elde edilir. Dolayısıyla "$\sim$" bağıntısı geçişmelidir. ■

Tanım 3.1.4 Θ kümesinin her f elemanı için f nin "$\sim$" bağıntısına göre denklik sınıfı $\Theta_f=\{\ g\ |\ |f|=|g|\}$ ile tanımlıdır.

Özellik 3.1.1 Her $\lambda\in\mathbb{R}\setminus\{0\}$ için $\lambda\odot f\in\Theta_f$ dir.

İspat Θ kümesinin her $f=\frac{a}{b}$ elemanı ve her $\lambda\in\mathbb{R}\setminus\{0\}$ için $|\lambda\odot f|=\left|\frac{\lambda a}{\lambda b}\right|=\left|\frac{a}{b}\right|=|f|$ olur. Θ_f nin tanımından $\lambda\odot f\in\Theta_f$ olduğu görülür. ■

Özellik 3.1.2 Her $g, h\in\Theta_f$ ve her $\lambda\in\mathbb{R}\setminus\{0\}$ için $|g\oplus h|=|f|$ ve $|\lambda\odot f|=|f|$ dir.

İspat $f=\frac{a}{b}\in\Theta$ olsun. $g\in\Theta_f\Rightarrow|f|=|g|$

$$\Rightarrow g=\frac{ka}{kb}\quad \exists k\in\mathbb{R}\setminus\{0\}$$

dır. $h \in \Theta_f \Rightarrow |f| = |h|$

$$\Rightarrow h = \frac{ma}{mb} \quad \exists m \in \mathbb{R} \setminus \{0\}$$

dır. Özellik 3.2.1 den

$$|g \oplus h| = \left|\frac{ka}{kb} \oplus \frac{ma}{mb}\right| = \left|\frac{ka+ma}{kb+mb}\right| = \left|\frac{(k+m)a}{(k+m)b}\right| = \left|\frac{a}{b}\right| = |f|$$

olur. ■

Özellik 3.1.3 Θ_f kümesi $\oplus$ işlemine göre Θ kümesinin bir alt grubudur.

İspat $\Theta_f \neq \varnothing$ dır, çünkü her $f \in \Theta$ için $|f| = |f|$ olduğundan $f \in \Theta_f$ tir. Her $g, h \in \Theta_f$ için Özellik 3.1.2 den $|g \oplus h| = |f|$ olduğundan $g \oplus h \in \Theta_f$ tir. Her $h \in \Theta_f$ için $|f| = |h|$ olduğundan $f = \frac{a}{b}$ için $h = \frac{ma}{mb}$ olacak şekilde en az bir $m \in \mathbb{R} \setminus \{0\}$ vardır. Teorem 3.1.1 den h elemanının $\oplus$ işlemine göre tersi $-h = \frac{-ma}{-mb} = (-m) \odot \frac{a}{b}$ dir. Özellik 3.1.2 den $|-h| = \left|(-m) \odot \frac{a}{b}\right| = \left|\frac{a}{b}\right| = |f|$ olur. Böylece $h \in \Theta_f$ için $-h$ da Θ_f nin bir elemanıdır. ■

Özellik 3.1.4 $\Theta'_f = \left\{g' \mid g' = \lambda \odot f,\ \lambda \in \mathbb{R}\right\}$ olmak üzere $\Theta_f = \Theta'_f$ dir.

İspat Her $g \in \Theta_f$ için $|g| = |f|$ olduğundan $f = \frac{a}{b}$ için $g = \frac{ma}{mb}$ olacak şekilde en az bir $m \in \mathbb{R} \setminus \{0\}$ vardır. Dolayısıyla $g = m \odot \frac{a}{b}$ olduğundan $g \in \Theta'_f$ dir, yani $\Theta_f \subseteq \Theta'_f$ dir.

$g' \in \Theta'_f$ olsun. O zaman $g' = \lambda \odot \frac{a}{b}$ olacak şekilde en az bir $\lambda \in \mathbb{R} \setminus \{0\}$, $f \in \Theta$ vardır. Özellik 3.2.3 ten $|g'| = |\lambda \odot f| = |f|$ dir. Buradan $g' \in \Theta'_f$ elde edilir, yani $\Theta'_f \subseteq \Theta_f$ dir.

Sonuç olarak, her iki kapsamadan $\Theta_f = \Theta'_f$ bulunur. ■

Özellik 3.1.5 $i = 1, ..., m \in \mathbb{N}$ olmak üzere her $f_i = \frac{a_i}{b_i} \in \Theta$ için $\bigoplus_{i=1}^{m} f_i = f_1 \frac{b_1}{b_1 + b_2 + ... + b_m} + f_2 \frac{b_2}{b_1 + b_2 + ... + b_m} + ... + f_m \frac{b_m}{b_1 + b_2 + ... + b_m}$ şeklinde yazılır.

İspat Tümevarım ispat tekniğini kullanarak $m = 2$ için doğru olduğunu gösterelim. Θ nın her $f_1 = \frac{a_1}{b_1}$, $f_2 = \frac{a_2}{b_2}$ elemanları için $f_1 \oplus f_2 = \frac{a_1}{b_1} \oplus \frac{a_2}{b_2} = \frac{a_1 + a_2}{b_1 + b_2}$ olur.

$$f_1 \oplus f_2 = f_1 \frac{b_1}{b_1 + b_2} + f_2 \frac{b_2}{b_1 + b_2} = \frac{a_1}{b_1} \frac{b_1}{b_1 + b_2} + \frac{a_2}{b_2} \frac{b_2}{b_1 + b_2}$$

$$= \frac{a_1}{b_1 + b_2} + \frac{a_2}{b_1 + b_2} = \frac{a_1 + a_2}{b_1 + b_2}$$

olduğundan istenilen eşitlik elde edilir.

Herhangi bir $k \geq 2$ için eşitliğin doğru olduğunu kabul edelim. Yani $\bigoplus_{i=1}^{k} f_i = f_1 \frac{b_1}{b_1 + b_2 + ... + b_k} + f_2 \frac{b_2}{b_1 + b_2 + ... + b_k} + ... + f_k \frac{b_k}{b_1 + b_2 + ... + b_k}$ olsun.

$k+1$ için doğruluğunu gösterelim.

$$\bigoplus_{i=1}^{k} f_i \oplus f_{k+1} = f_1 \frac{b_1}{b_1+b_2+\ldots+b_k} + f_2 \frac{b_2}{b_1+b_2+\ldots+b_k} + \ldots + f_k \frac{b_k}{b_1+b_2+\ldots+b_k} \oplus \frac{a_{k+1}}{b_{k+1}}$$

$$= \frac{a_1}{b_1} \frac{b_1}{b_1+b_2+\ldots+b_k} + \frac{a_2}{b_2} \frac{b_2}{b_1+b_2+\ldots+b_k} + \ldots + \frac{a_k}{b_k} \frac{b_k}{b_1+b_2+\ldots+b_k} \oplus \frac{a_{k+1}}{b_{k+1}}$$

$$= \frac{a_1+a_2+\ldots+a_k+a_{k+1}}{b_1+b_2+\ldots+b_k+b_{k+1}}$$

$$= f_1 \frac{b_1}{b_1+b_2+\ldots+b_k+b_{k+1}} + \ldots + f_{k+1} \frac{b_{k+1}}{b_1+b_2+\ldots+b_k+b_{k+1}}$$

elde edilir. ■

Teorem 3.1.3 $(\Theta, \oplus)$ Abel grubu $\mathbb{R}$ reel sayılar cismi üzerinde bir vektör uzayıdır.

İspat Her $\lambda \in \mathbb{R}$ ve her $f \in \Theta$ için,

$$\mathbb{R} \times \Theta \to \Theta$$

$$(\lambda, f) \mapsto \lambda \odot f$$

bağıntısı iyi tanımlıdır, çünkü $\lambda_1, \lambda_2 \in \mathbb{R}$ ve $f_1, f_2 \in \Theta$ için $(\lambda_1, f_1) = (\lambda_2, f_2)$ durumunda $\lambda_1 = \lambda_2$ ve $f_1 = f_2$ dir. Buradan $\lambda_1 \odot f_1 = \lambda_2 \odot f_1 = \lambda_2 \odot f_2$ elde edilir.

Her $\lambda \in \mathbb{R}$ ve her $f_1 = \frac{a}{b},\ f_2 = \frac{c}{d} \in \Theta$ için,

$$\lambda \odot (f_1 \oplus f_2) = \lambda \odot \left(\frac{a}{b} \oplus \frac{c}{d}\right) = \lambda \odot \left(\frac{a+c}{b+d}\right) = \frac{\lambda(a+c)}{\lambda(b+d)} = \frac{\lambda a + \lambda c}{\lambda b + \lambda d} = \frac{\lambda a}{\lambda b} \oplus \frac{\lambda c}{\lambda d}$$

$$= \left(\lambda \odot \frac{a}{b}\right) \oplus \left(\lambda \odot \frac{c}{d}\right) = (\lambda \odot f_1) \oplus (\lambda \odot f_2)$$

olur. Her $\lambda_1, \lambda_2 \in \mathbb{R}$ ve her $f = \frac{a}{b} \in \Theta$ için,

$$(\lambda_1 + \lambda_2) \odot f = (\lambda_1 + \lambda_2) \odot \frac{a}{b} = \frac{(\lambda_1 + \lambda_2)a}{(\lambda_1 + \lambda_2)b} = \frac{\lambda_1 a + \lambda_2 a}{\lambda_1 b + \lambda_2 b} = \frac{\lambda_1 a}{\lambda_1 b} \oplus \frac{\lambda_2 a}{\lambda_2 b}$$

$$=\left(\lambda_1 \odot \frac{a}{b}\right)\oplus\left(\lambda_2 \odot \frac{a}{b}\right)=(\lambda_1 \odot f)\oplus(\lambda_2 \odot f)$$

olur. Her $\lambda_1,\lambda_2 \in \mathbb{R}$ ve her $f=\frac{a}{b}\in\Theta$ için

$$\lambda_1 \odot(\lambda_2 \odot f)=\lambda_1 \odot\left(\lambda_2 \odot \frac{a}{b}\right)=\lambda_1 \odot\left(\frac{\lambda_2 a}{\lambda_2 b}\right)=\frac{\lambda_1(\lambda_2 a)}{\lambda_1(\lambda_2 b)}=\frac{(\lambda_1\lambda_2)a}{(\lambda_1\lambda_2)b}$$

$$=(\lambda_1\lambda_2)\odot\frac{a}{b}=(\lambda_1 \odot \lambda_2)\odot f$$

olur. Son olarak her $f=\frac{a}{b}\in\Theta$ için $1_{\mathbb{R}} \odot f=1_{\mathbb{R}} \odot\frac{a}{b}=\frac{1a}{1b}=\frac{a}{b}=f$ dir. ■

Teorem 3.1.4 $(\Theta, \oplus, \odot, \otimes)$ sistemi Reel sayılar cismi üzerinde bir cebirdir.

İspat Teorem 3.1.2 den $(\Theta,\oplus,\otimes)$ sistemi bir halka ve Teorem 3.1.3 ten $(\Theta,\oplus,\odot)$ sisteminin $\mathbb{R}$ reel sayılar cismi üzerinde bir vektör uzayı olduğunu biliyoruz. Ayrıca $\lambda\in\mathbb{R}$ ve $f_1=\frac{a_1}{b_1}$, $f_2=\frac{a_2}{b_2}\in\Theta$ için,

$$\lambda \odot(f_1 \otimes f_2)=\lambda \odot\left(\frac{a_1}{b_1}\otimes\frac{a_2}{b_2}\right)=\lambda \odot\left(\frac{a_1a_2}{b_1b_2}\right)=\frac{\lambda(a_1a_2)}{\lambda(b_1b_2)}=\frac{a_1\lambda a_2}{b_1\lambda b_2}$$

$$=\frac{\lambda a_1}{\lambda b_1}\otimes\frac{a_2}{b_2}=\left(\lambda\odot\frac{a_1}{b_1}\right)\otimes\frac{a_2}{b_2}=(\lambda\odot f_1)\otimes f_2$$

ve

$$\lambda \odot(f_1 \otimes f_2)=\lambda \odot\left(\frac{a_1}{b_1}\otimes\frac{a_2}{b_2}\right)=\lambda \odot\left(\frac{a_1a_2}{b_1b_2}\right)=\frac{\lambda(a_1a_2)}{\lambda(b_1b_2)}=\frac{\lambda a_1a_2}{\lambda b_1b_2}$$

$$= \frac{a_1}{b_1} \otimes \left(\frac{\lambda a_2}{\lambda b_2} \right) = \frac{a_1}{b_1} \otimes \left(\lambda \odot \frac{a_2}{b_2} \right) = f_1 \otimes (\lambda \odot f_2)$$

sağlanır. Böylece $(\Theta, \oplus, \odot, \otimes)$ sistemi $\mathbb{R}$ üzerinde bir cebirdir. ■

3.2 Kesirler Cebirinin Geometrik Yorumu

Θ nın her $f = \frac{a}{b}$ elemanına $\mathbb{R}^2$ düzleminde bir $\vec{t} = \{b, a\}$ vektörü karşılık getirebiliriz. Bu durumda bir önceki bölümde Θ kümesinde tanımlamış olduğumuz işlemler vektörel işlemlere karşılık gelir. Aşağıda ifade edilen α açıları biribirinden farklı açıları göstermektedir.

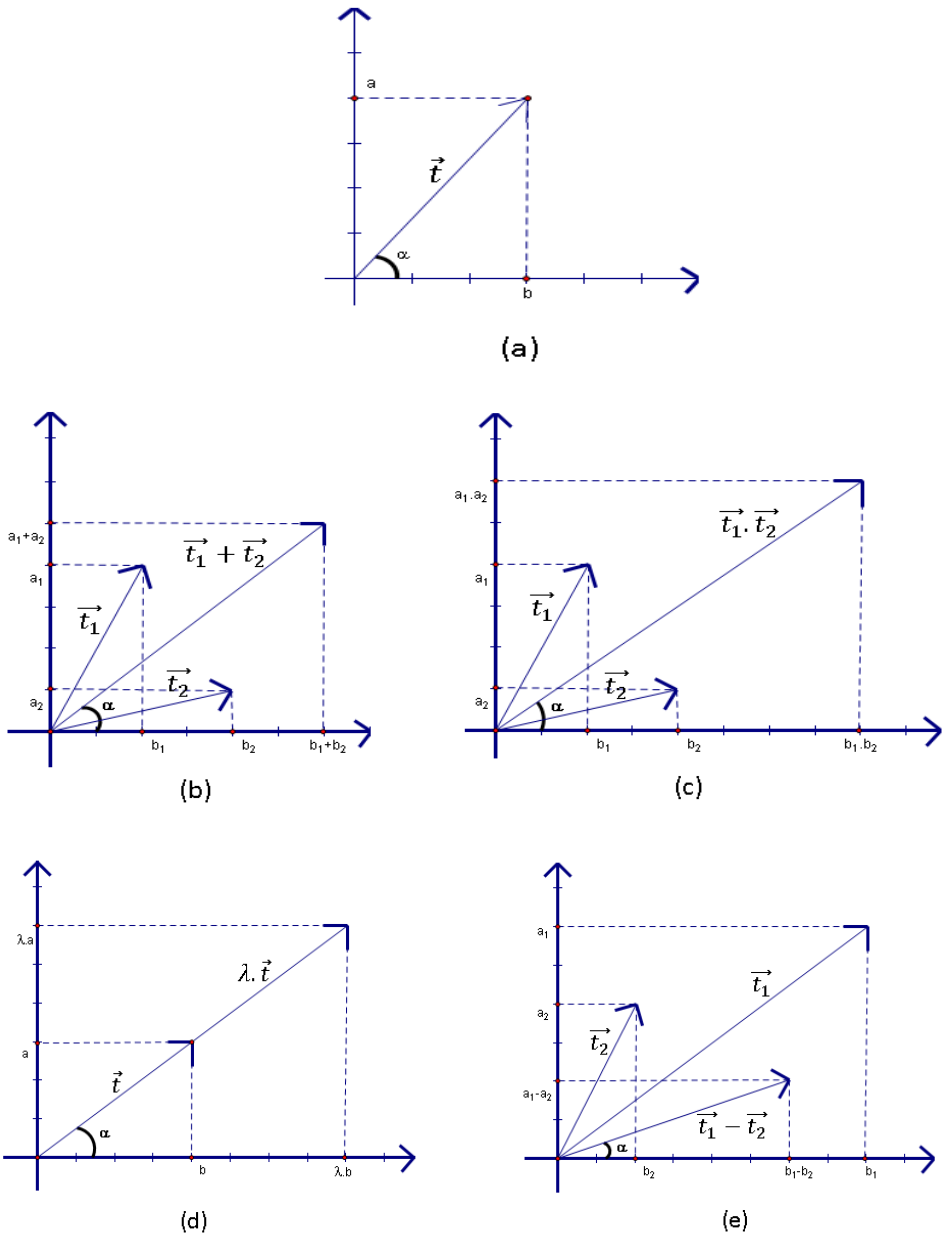
$f = \frac{a}{b} \in \Theta$ için $\vec{t} = \{b, a\}$ vektörü (Şekil 3.1.a) deki gibi gösterilir. Bu durumda f nin modülü $\vec{t}$ vektörünün Ox-ekseni ile pozitif yönde yaptığı açının tanjantına eşittir. Yani, $|f| = \tan\alpha$ dır.

$\frac{0}{0} \neq f_1 = \frac{a_1}{b_1}, \frac{0}{0} \neq f_2 = \frac{a_2}{b_2} \in \Theta$ için f_1 ve f_2 ye karşılık gelen vektörler sırasıyla $\vec{t_1}, \vec{t_2}$ ve $f_1 \oplus f_2$ ye karşılık gelen vektörün Ox-ekseni ile pozitif yönde yaptığı açıda α olsun (Şekil 3.1.b). O zaman $|f_1 \oplus f_2| = \tan\alpha$ dır.

$\frac{0}{0} \neq f_1 = \frac{a_1}{b_1}, \frac{0}{0} \neq f_2 = \frac{a_2}{b_2} \in \Theta$ için f_1 ve f_2 ye karşılık gelen vektörler sırasıyla $\vec{t_1}, \vec{t_2}$ ve $f_1.f_2$ ye karşılık gelen vektörün Ox-ekseni ile pozitif yönde yaptığı açıda α olsun (Şekil 3.1.c). O zaman $|f_1.f_2| = \tan\alpha$ dır.

$\frac{0}{0} \neq f = \frac{a}{b} \in \Theta$ için $\odot$ işlemine, $\vec{t}$ vektörünün $\lambda \neq 0$ sayısı ile çarpımı karşılık gelir (Şekil 3.1.d). α , f nin Ox-ekseni ile pozitif yönde yaptığı açı olmak üzere, $|\lambda \odot f| = \tan\alpha$ dır.

$\frac{0}{0} \neq f_1 = \frac{a_1}{b_1}, \frac{0}{0} \neq f_2 = \frac{a_2}{b_2} \in \Theta$ için f_1 ve f_2 ye karşılık gelen vektörler sırasıyla $\vec{t}_1, \vec{t}_2$ ve $f_1 \ominus f_2$ ye karşılık gelen vektörün Ox-ekseni ile pozitif yönde yaptığı açıda α olsun (Şekil 3.1.e). O zaman $|f_1 \ominus f_2| = \tan\alpha$ dır.



Şekil 3.1 Tanımlanan ikili işlemlerin vektörel gösterimi.

a) f nin düzlemde vektörel gösterimi, b) $f_1 \oplus f_2$ nin düzlemde vektörel gösterimi, c) $f_1 . f_2$ nin düzlemde vektörel gösterimi, d) $\lambda \odot f$ nin düzlemde vektörel gösterimi, e) $f_1 \ominus f_2$ nin düzlemde vektörel gösterimi

3.3 Kesirler Cebiri için Temel Bazı Eşitsizlikler

Bu bölümde ispatlanacak olan eşitsizlikler diğer bölümlerde kullanılmak üzere temel oluşturacaktır.

Önerme 3.3.1 Her $f_1,\ f_2 \in \Theta$ için $|f_1| \geq |f_2|$ ise $|f_1|+|f_2| \geq |f_1| \geq |f_1 \oplus f_2| \geq |f_2|$ dir.

İspat $f_1 = \frac{a_1}{b_1}$ ve $f_2 = \frac{a_2}{b_2}$ olmak üzere $|f_1| = k_1$ ve $|f_2| = k_2$ olsun. Buradan

$|f_1| = \left|\frac{a_1}{b_1}\right| = k_1 \Rightarrow a_1 = b_1 k_1$ ve $|f_2| = \left|\frac{a_2}{b_2}\right| = k_2 \Rightarrow a_2 = b_2 k_2$ yazılabilir. $|f_1| \geq |f_2|$

olduğundan $k_1 \geq k_2$ dir.

$$|f_1 \oplus f_2| = \left|\frac{a_1}{b_1} \oplus \frac{a_2}{b_2}\right| = \left|\frac{a_1 + a_2}{b_1 + b_2}\right| = \left|\frac{b_1 k_1 + b_2 k_2}{b_1 + b_2}\right| \geq \left|\frac{b_1 k_2 + b_2 k_2}{b_1 + b_2}\right| = \left|\frac{(b_1 + b_2) k_2}{b_1 + b_2}\right| = k_2 = |f_2|$$

Dolayısıyla

$$|f_1 \oplus f_2| \geq |f_2| \tag{3.3.1}$$

dir. Aynı zamanda

$$|f_1 \oplus f_2| = \left|\frac{a_1}{b_1} \oplus \frac{a_2}{b_2}\right| = \left|\frac{a_1 + a_2}{b_1 + b_2}\right| = \left|\frac{b_1 k_1 + b_2 k_2}{b_1 + b_2}\right| \leq \left|\frac{b_1 k_1 + b_2 k_1}{b_1 + b_2}\right| = \left|\frac{(b_1 + b_2) k_1}{b_1 + b_2}\right| = k_1 = |f_1|$$

olduğundan

$$|f_1 \oplus f_2| \leq |f_1| \tag{3.3.2}$$

dir. Böylece (3.3.1) ve (3.3.2) kullanılarak istenilen eşitsizlik elde edilir.

Önerme 3.3.2

$|f_1| \geq |f_2|$ ve $\lambda, \mu \in \mathbb{R} \setminus \{0\}$ olmak üzere $|f_1| \geq |(\lambda \odot f_1) \oplus (\mu \odot f_2)| \geq |f_2|$ eşitsizliği sağlanır.

İspat Önerme 3.3.1 den

$$|(\lambda \odot f_1) \oplus (\mu \odot f_2)| \geq |\mu \odot f_2| = |f_2| \tag{3.3.3}$$

$$|f_1| = |\lambda \odot f_1| \geq |(\lambda \odot f_1) \oplus (\mu \odot f_2)| \tag{3.3.4}$$

vardır. Böylece (3.3.3) ve (3.3.4) kullanılarak istenilen eşitsizlik ispatlanmış olur.

Önerme 3.3.3 $\lambda_1, \lambda_2, \mu_1, \mu_2 \in \mathbb{R}^+$ ve $f_1, f_2 \in \Theta_{\mathbb{R}^+}$ için $|f_1| \geq |f_2|$ ve $\lambda_1\mu_2 \geq \lambda_2\mu_1$ olsun. Bu durumda $|(\lambda_1 \odot f_1) \oplus (\mu_1 \odot f_2)| \geq |(\lambda_2 \odot f_1) \oplus (\mu_2 \odot f_2)|$ eşitsizliği sağlanır.

İspat $|f_1| = \left|\frac{a_1}{b_1}\right| = k_1$ ve $|f_2| = \left|\frac{a_2}{b_2}\right| = k_2$ olmak üzere $a_1 = k_1 b_1$ ve $a_2 = k_2 b_2$ şeklinde yazılabilir.

$$(\lambda_1 \odot f_1) \oplus (\mu_1 \odot f_2) - (\lambda_2 \odot f_1) \oplus (\mu_2 \odot f_2) = \frac{\lambda_1 k_1 b_1 + \mu_1 k_2 b_2}{\lambda_1 b_1 + \mu_1 b_2} - \frac{\lambda_2 k_1 b_1 + \mu_2 k_2 b_2}{\lambda_2 b_1 + \mu_2 b_2}$$

$$= \frac{(\lambda_1 k_1 b_1 + \mu_1 k_2 b_2)(\lambda_2 b_1 + \mu_2 b_2) - (\lambda_2 k_1 b_1 + \mu_2 k_2 b_2)(\lambda_1 b_1 + \mu_1 b_2)}{(\lambda_1 b_1 + \mu_1 b_2)(\lambda_2 b_1 + \mu_2 b_2)}$$

$$= \frac{(\lambda_1\mu_2 - \lambda_2\mu_1) b_1 b_2 (k_1 - k_2)}{(\lambda_1 b_1 + \mu_1 b_2)(\lambda_2 b_1 + \mu_2 b_2)}$$

elde edilir. Kabulden $\lambda_1\mu_2 \geq \lambda_2\mu_1$, $\lambda_1\mu_2 - \lambda_2\mu_1 \geq 0$ ve $k_1 \geq k_2, k_1 - k_2 \geq 0$ olduğundan

$$|(\lambda_1 \odot f_1) \oplus (\mu_1 \odot f_2) - (\lambda_2 \odot f_1) \oplus (\mu_2 \odot f_2)| \geq 0$$

elde edilir.

Önerme 3.3.4 Her $f, g, \varphi \in \Theta$ için $|f| > |g|$ ve $|f \oplus g| \leq |\varphi|$ olsun. Bu durumda $|f| < |\varphi|$ ve ya $|f| > |\varphi|$ ise $|g| < |\varphi|$ eşitsizlikleri sağlanır.

İspat $f_1 = f$ ve $f_2 = g$ olarak alınırsa, Önerme 3.3.1 kullanılarak $|f| \geq |f \oplus g| \geq |g|$ elde edilir. $|f \oplus g| \leq |\varphi|$ hipotezinden iki durum dikkate alınır:
İlk olarak, $|f| \geq |\varphi| \geq |f \oplus g| \geq |g|$ dir ve buradan $|g| \leq |\varphi|$ iken $|f| \geq |\varphi|$ dir.
İkinci olarak, $|\varphi| \geq |f| \geq |f \oplus g| \geq |g|$ dir ve buradan $|f| \leq |\varphi|$ dir.

4 İNVERSİYON ÖZELLİKLERİ VE FARKLI STRATEJİLERE GÖRE TOPLAMLAR

4.1 İnversiyon Özellikleri

Bu bölümde Θ kümesinin $\mathbb{R}^+$ üzerinde tanımlı, modülüne göre sıralanmış elemanların $\oplus$ işlemi altında sıralamayı koruyup-korumadığı incelenecektir. Burada her $F, f_1, f_2 \in \Theta_{\mathbb{R}^+}$ ve $|F| \geq |f_1| \geq |f_2|$ olmak üzere

$$F_1 = F \oplus f_1,\ F_2 = F \oplus f_2$$
$$F^* = max\{F_1,\ F_2\}$$
$$f^* = min\{f_1,\ f_2\}$$

olarak alınacaktır.

Teorem 4.1.1 Eğer $|f_1 \ominus f_2| \notin (|f_2|, |F_2|)$ ise $|F_1| \geq |F_2|$ dir.

İspat $F = \frac{A}{B},\ f_1 = \frac{a+\Delta a}{b+\Delta b},\ f_2 = \frac{a}{b}$ olmak üzere,

$$F_1 = F \oplus f_1 = \frac{A}{B} \oplus \frac{a+\Delta a}{b+\Delta b} = \frac{A+a+\Delta a}{B+b+\Delta b}$$
$$F_2 = F \oplus f_2 = \frac{A}{B} \oplus \frac{a}{b} = \frac{A+a}{B+b}$$
$$f_1 \ominus f_2 = \frac{\Delta a}{\Delta b}$$

dir.

$F_1 - F_2$ ve $f_1 - f_2$ sırasıyla ΔF ve Δf ile tanımlanırsa

$$\Delta F = F_1 - F_2 = \frac{A+a+\Delta a}{B+b+\Delta b} - \frac{A+a}{B+b} \tag{4.1.1}$$

$$\Delta f = f_1 - f_2 = \frac{a+\Delta a}{b+\Delta b} - \frac{a}{b} \tag{4.1.2}$$

olarak elde edilir.

Bu durumda

$$\Delta F = \rho.\Delta f, \qquad \rho > 0 \tag{4.1.3}$$

dir.

Teoremin ispatı için

$$\frac{\Delta a}{\Delta b} \notin \left[\frac{a}{b}, \frac{A+a}{B+b}\right] \tag{4.1.4}$$

olduğunu göstermek yeterlidir. (4.1.1) ve (4.1.2) denklemlerini yeniden düzenlersek,

$$\begin{aligned}\Delta F = F_1 - F_2 &= \frac{A+a+\Delta a}{B+b+\Delta b} - \frac{A+a}{B+b} \\ &= \frac{AB+Ab+aB+ab+B\Delta a+b\Delta a-AB-Ab-A\Delta b-aB-ab-a\Delta b}{(B+b+\Delta b)(B+b)} \\ &= \frac{B\Delta a+b\Delta a-A\Delta b-a\Delta b}{(B+b+\Delta b)(B+b)} = \frac{\Delta a-\frac{(A+a)}{B+b}\Delta b}{(B+b+\Delta b)} = \frac{\Delta a-F_2\Delta b}{(B+b+\Delta b)}\end{aligned}$$

ve

$$\begin{aligned}\Delta f = f_1 - f_2 &= \frac{a+\Delta a}{b+\Delta b} - \frac{a}{b} = \frac{ab-b\Delta a-ab-a\Delta b}{b(b+\Delta b)} \\ &= \frac{\Delta a-\frac{a}{b}\Delta b}{b+\Delta b} = \frac{\Delta a-f_2\Delta b}{b+\Delta b}\end{aligned}$$

olarak bulunur.

$$\Delta F = \frac{\Delta a-F_2\Delta b}{\Delta a-f_2\Delta b}\,\frac{b+\Delta b}{B+b+\Delta b}\,\frac{\Delta a-f_2\Delta b}{b+\Delta b}$$

şeklinde yazalırsa

$$\Delta F = \frac{\Delta a - F_2 \Delta b}{\Delta a - f_2 \Delta b} \frac{b + \Delta b}{B + b + \Delta b} \Delta f \tag{4.1.5}$$

olarak elde edilir. Eğer

$$\frac{\Delta a - F_2 \Delta b}{\Delta a - f_2 \Delta b} \frac{b + \Delta b}{B + b + \Delta b} > 0 \tag{4.1.6}$$

olduğu gösterilirse (4.1.5) ile (4.1.3) eşitsizlikleri aynı olur. Kesir doğrusal fonksiyonunun paynın elemanları $\mathbb{R}^+$ dan alındığı için $b + \Delta b > 0$ dır. Bu durumda (4.1.6) eşitsizliğinin sağladığını göstermek için

$$\frac{\Delta a - F_2 \Delta b}{\Delta a - f_2 \Delta b} > 0 \tag{4.1.7}$$

olduğunu ispatlamak gerekir. Aşağıdaki koşulların birisinin sağlaması durumunda (4.1.7) eşitsizliği vardır.

(i) $\dfrac{\Delta a}{\Delta b} > \max\{F_2,\ f_2\} = F_2 = \dfrac{A + a}{B + b}$

(ii) $\dfrac{\Delta a}{\Delta b} < \min\{F_2,\ f_2\} = f_2 = \dfrac{a}{b}$

Ancak (i) ve (ii) koşulları (4.1.4) in sağlandığı anlamına gelir. ■

Sonuç Teorem 4.1.1 Eğer $|f_1 \ominus f_2| \notin (|f_2|, |F_2|)$ ise $|f_1 \ominus f_2| \notin (|f_1|, |F_1|)$ sağlanır.

İspat Teorem 4.1.1 den, $|f_1 \ominus f_2| \notin (|f_2|, |F_2|)$ ise $|F_1| > |F_2|$ dir. $|f_1 \ominus f_2| \notin (|f_2|, |F_2|)$ ise $f_1 \ominus f_2 = f_\Delta$ olarak tanımlanmak üzere söz konusu iki durum aşağıdaki gibidir:

(i) $f_\Delta \le |f_2|$ ise, $|f_1| \ge |f_2|$ olduğundan $f_\Delta \le |f_1|$ dir.

(ii) $f_\Delta \geq |F_2|$ ise, $F_1 = F_2 \oplus f_\Delta$ olduğundan $f_\Delta \geq |F_1|$ dir.

Böylece $f_\Delta \notin (|f_1|, |F_1|)$ olduğu elde edilir. ■

Sonuç 4.1.2 $|f_1 \ominus f_2| \notin (|f_1|, |F_1|)$ ve ya $|f_1 \ominus f_2| \notin (|f_2|, |F_2|)$ ise $|f_1 \ominus f_2| \in (|f^*|, |F^*|)$ dır.

İspat Teorem 4.1.1 ve sonuç 4.1.2 den açıktır. ■

Tanım 4.1.1 $M_{f_1, f_2} = (-\infty, |f^*|) \cup (|F^*|, +\infty)$ 'a karşılı gelen aralığa *Monoton alanı* ve bu aralığın dışında kalan, $U_{f_1, f_2} = (|f^*|, |F^*|)$ ile tanımlanan aralığa *İnversiyon alanı* adı verilir.

$F_1 = F \oplus f_1$ ve $F_2 = F \oplus f_2$ olmak üzere $|f_1| \geq |f_2|$ ise $|F_1| \geq |F_2|$ olduğu bu tanımdan görülür. İnversiyon alanında ise tam tersi durum söz konusudur, yani $|f_1| \geq |f_2|$ ise $|F_1| \leq |F_2|$ dir.

Tanım 4.1.2 $f_1 = \frac{a_1}{b_1}$, $f_2 = \frac{a_2}{b_2}$ olmak üzere $|f_1 \ominus f_2| = |f_1|$ ise $|f_1| = |f_2| = |f^*|$ dır ve $b_1 \leq b_2$ ise $|F_1| > |F_2|$ dir. Bu durumda f^* noktasına *çok değerlilik noktası* adı verilir.

Tanım 4.1.3 $|f_1 \ominus f_2| = |F_1|$ ise $|F_1| = |F_2| = |F^*|$ dır. Bu durumda F^* noktasına *değişmez (invaryant) nokta* adı verilir.

4.2 Farklı Stratejilere Göre Toplamlar

Bu bölümde, üç tane farklı stratejiye göre toplamdan bahsedilerek bunların kıyaslanması yapılacak ve Lokal ve Global kriterlerin birbirlerine eşit olma koşulu aranacaktır.

τ_n dizilişi,

$$\tau_n = \left\{ f_i \,\middle|\, i \in \omega_n,\ \omega_n = 1,2,...,n,\ |f_i| \geq |f_{i+1}|,\ i = 1,...,n-1 \right\}$$

şeklinde tanımlansın. $f_i \in \tau_n$ olmak üzere

1. $F_k^L = \bigoplus_{i=1}^{k} f_i$

olarak tanımlanan F_k^L, *Lokal Kritere* göre toplamayı gösterir. Yani burada F_k^L, modülüne göre sıralanmış durumdaki ilk *k* elemanın koordinatsal toplamı anlamına gelir. $\omega_k^L = \{1,2,...,k\}$ ise sıralanmış, ilk *k* indis kümesini göstersin.

2. $F_1^G = f_1$,

$$F_{k+1}^G = \left\{ F_k^G \oplus f_j \,\middle|\, \left|F_k^G \oplus f_j\right| = \max_{i \in \omega_n \setminus \omega_k^G} \left\{ \left|F_k^G \oplus f_i\right| \right\} \right\},$$

$$\omega_1^G = \{1\},$$

$$\omega_{k+1}^G = \left\{ w_k^G \cup j \,\middle|\, F_{k+1}^G = F_k^G \oplus f_j \right\}$$

olmak üzere F_k^G, *Global Kritere* göre toplamayı gösterir. F_{k+1}^G ise bir önceki adımda belirlenmis, olan F_k^G yi otomatik olarak F_{k+1}^G içerisine koyar ve kalan elemanlar içerisinden $(k+1)$. eleman olarak koordinatsal toplamı maksimum yapan f_j elemanını seçer.

3. F_k^* ise $k < n$ için tüm durumlar incelenerek koordinatsal toplamı maksimum yapan k tane f_i kesirini seçer, w_k^* da F_k^* da seçilen f_i lerin indisleri kümesi olarak alınır.

$$F_k^* = \left| \bigoplus_{i \in \omega_k^*} f_i \right| = \max_{\omega_k \subset \omega_n} \left\{ \left| \bigoplus_{i \in \omega_k} f_i \right| \right\}$$

olarak tanımlıdır.

Bu problem $k=1$ ve $k=2$ için kolayca çözülür. Gerçekten k=1 için $\omega_1^* = \{1\}$ dir, yani lokal ve global toplamlar üst üste düşer. ω_2^G ise, $\omega_2^G = \{(1,m) \mid |f_1 \oplus f_m|\} = max\{|f_1 \oplus f_i|\}$ şeklinde tanımlıdır.

Teorem 4.2.1 $\omega_2^G = \omega_2^*$ dır.

İspat Teoremin tersini kabul edelim. Yani
$\omega_2^G = \{(1,m) \mid |f_1 \oplus f_m|\} = max\{|f_1 \oplus f_i|\} = \{1,m\}$ ve $\omega_2^* = \{p,s\} = \{max|f_i \oplus f_j|\}$
olsun. Bu durumda

$$|f_p \oplus f_s| > |f_1 \oplus f_m| \qquad (4.2.1)$$

olsun, kesinlik için ise

$$|f_1 \oplus f_s| \le |f_1 \oplus f_p| \qquad (4.2.2)$$

olduğunu kabul edelim. (4.2.1) ve ω_2^G nin tanımından

$$|f_1 \oplus f_p| < |f_1 \oplus f_m| \qquad (4.2.3)$$

dir. $|f_1| + |f_2| \ge |f_1| \ge |f_1 \oplus f_2| \ge |f_1|$ eşitsizliği (4.2.2) de uygulanırsa

$$\left|(f_1 \oplus f_s) \oplus (f_1 \oplus f_p)\right| \ge |f_1 \oplus f_s| \qquad (4.2.4)$$

elde edilir. Θ kümesinin elemanları $\oplus$ ile değişme ve birleşme özelliğine sahiptir. Dolayısıyla

$$\left|(f_1 \oplus f_1) \oplus (f_s \oplus f_p)\right| \le |f_1 \oplus f_m| \qquad (4.2.5)$$

dir. Bunun yanında $|f_1 \oplus f_1| = |f_1|$ olduğundan

$$|f_1 \oplus f_1| > |f_1 \oplus f_m| \tag{4.2.6}$$

dir. Önerme 3.3.4, (4.2.1) de uygulanırsa yani

$$|f_p \oplus f_s| > |f_1 \oplus f_1| \text{ ve } |(f_p \oplus f_s) \oplus (f_1 \oplus f_1)| \leq |f_1 \oplus f_m| \tag{4.2.7}$$

olduğundan ya

$$|f_p \oplus f_s| \leq |f_1 \oplus f_m| \tag{4.2.8}$$

yada

$$|f_1 \oplus f_s| > |f_1 \oplus f_m| \text{ ise } |f_1 \oplus f_1| < |f_1 \oplus f_m| \tag{4.2.9}$$

dir. (4.2.9), (4.2.6) ile, (4.2.8) ise kabulumuz olan (4.2.1) ile çelişir ve ispat biter. ■

Lokal ve Global kriterlere göre toplamlar her zaman maksimal değeri vermeyebilir. Bunu gösteren bir örnek verelim.

$\tau_n = \left\{ f_1 = \frac{50}{10},\ f_2 = \frac{70}{20},\ f_3 = \frac{50}{15},\ f_4 = \frac{11}{5} \right\}$ olmak üzere $|f_1| > |f_2| > |f_3| > |f_4|$ dir.

a) $F_1^L = \left\{ \frac{50}{10} \right\} \Rightarrow \omega_1^L = \{1\}$, $F_1^G = \left\{ \frac{50}{10} \right\} \Rightarrow \omega_1^G = \{1\}$ ve

$F_1^* = \left\{ \frac{50}{10} \right\} = f_1 \Rightarrow \omega_1^* = \{1\}$

b) $F_2^L = f_1 \oplus f_2$ olduğundan $\omega_2^L = \{1,2\}$

$$\begin{cases} \frac{50}{10} \oplus \frac{70}{20} = \frac{120}{30} \\ \frac{50}{10} \oplus \frac{50}{15} = \frac{100}{25} \\ \frac{50}{10} \oplus \frac{11}{5} = \frac{61}{15} \end{cases}$$

Bu üç toplamın içerisinden üçüncü en büyük olduğundan $F_2^G = \frac{50}{10} \oplus \frac{11}{5} = \frac{61}{15} = F_1^G \oplus f_4$ ve buradan $\omega_2^G = \omega_1^G \cup \{4\} = \{1, 4\}$ olarak bulunur.

$$\begin{cases} \frac{50}{10} \oplus \frac{70}{20} = \frac{120}{30}, & \frac{70}{20} \oplus \frac{50}{15} = \frac{120}{35} \\ \frac{50}{10} \oplus \frac{50}{15} = \frac{100}{25}, & \frac{70}{20} \oplus \frac{11}{5} = \frac{81}{25} \\ \frac{50}{10} \oplus \frac{11}{5} = \frac{61}{15}, & \frac{50}{15} \oplus \frac{11}{5} = \frac{61}{20} \end{cases}$$

Bu toplamlar içerisinden beşincisi en büyük olduğundan $F_2^* = \frac{50}{10} \oplus \frac{11}{5} = f_1 \oplus f_4$ ve böylece $\omega_2^* = \{1, 4\}$ olarak bulunur.

c) $F_3^L = f_1 \oplus f_2 \oplus f_3 = \frac{50}{10} \oplus \frac{70}{20} \oplus \frac{50}{15} = \frac{170}{45}, \quad \omega_3^L = \{1, 2, 3\}$

$$\begin{cases} \frac{61}{15} \oplus \frac{70}{20} = \frac{131}{35} \\ \frac{61}{15} \oplus \frac{50}{15} = \frac{111}{30} \end{cases}$$

Bu iki toplamın içinde birincisi en büyük olduğu için $F_3^G = \frac{61}{15} \oplus \frac{70}{20} = \frac{131}{35} = F_2^G \oplus f_2$ ve buradan $\omega_3^G = \omega_2^G \cup \{2\} = \{1, 2, 4\}$ olarak bulunur.

$$\begin{cases} \frac{50}{10} \oplus \frac{70}{20} \oplus \frac{50}{15} = \frac{170}{45} \\ \frac{50}{10} \oplus \frac{70}{20} \oplus \frac{11}{5} = \frac{131}{35} \\ \frac{70}{20} \oplus \frac{50}{15} \oplus \frac{11}{5} = \frac{131}{40} \end{cases}$$

Bu üç toplamın içerisinde birincisi en büyük olduğu için $F_3^* = \frac{50}{10} \oplus \frac{70}{20} \oplus \frac{50}{15} = \frac{170}{45} = f_1 \oplus f_2 \oplus f_3$ ve buradan $\omega_3^* = \{1, 2, 3\}$ olarak bulunur.

Lokal ve Global Kriterlere göre toplamanın hangi koşul altında eşit oldugunu gösteren bir Teorem verelim.

Teorem 4.2.2 Her $i, j \in \omega_n$ için $\left| f_i \ominus f_j \right| \notin U_{f_i f_j}$ ise her k için, $\omega_k^G = \omega_k^L = \omega_k^*$ dır.

İspat $\omega_k^G = \omega_k^L$ eşitliğinin ispatı teoremin koşulları göz önüne alındığında $\omega_k^G = \omega_k^L$ kümelerinin kurulması algoritmasından açıktır. $\omega_k^G = \omega_k^*$ eşitliğini tümevarım üzerinden gösterirsek;

$k = 1$ için $\omega_1^G = \omega_1^*$ olduğu açıktır.

$k = m-1$ için doğru olduğunu kabul edelim. Yani $\omega_{m-1}^G = \omega_{m-1}^*$ olsun. $\omega_m^G = \omega_m^*$ eşitliğinin sağlanmadığını kabul edelim. O zaman

$$\left| \bigoplus_{i \in \tilde{\omega}_m} f_i \right| > \left| \bigoplus_{i \in \omega_m^G} f_i \right| \tag{4.2.10}$$

olacak şekilde en az öyle bir $\tilde{\omega}_m$ vardır. Bu durumda $\tilde{\omega}_m \cap \omega_m^G = \varnothing$ ve $\tilde{\omega}_m \cap \omega_m^G \neq \varnothing$ olacak şekilde iki durum söz konusudur.

İlk olarak $\tilde{\omega}_m \cap \omega_m^G = \varnothing$ olduğunu kabul edelim. Bu durumda $\exists\, p \in \tilde{\omega}_m$ ve $\forall s \in \tilde{\omega}_m$ için

$$\left| \left(\bigoplus_{i \in \omega_{m-1}^G} f_i \right) \oplus f_s \right| \leq \left| \left(\bigoplus_{i \in \omega_{m-1}^G} f \right) \oplus f_p \right| \tag{4.2.11}$$

dir. ω_{k+1}^G tanımından,

$$\left| \left(\bigoplus_{i \in \omega_{m-1}^G} f_i \right) \oplus f_p \right| \leq \left| \left(\bigoplus_{i \in \omega_{m-1}^G} f_i \right) \oplus f_m \right| = \left| \left(\bigoplus_{i \in \omega_m^G} f_i \right) \right| = F_m^G \tag{4.2.12}$$

dir. (4.2.13) ve Önerme 3.3.1 kullanılarak

$$\left|\bigoplus_{t\in\bar{\omega}_m}\left[\left(\bigoplus_{i\in\omega_{m-1}^G} f_i\right)\oplus f_t\right]\right| \le \left|\left(\bigoplus_{i\in\omega_{m-1}^G} f_i\right)\oplus f_p\right| \tag{4.2.13}$$

$$\left|\left[\bigoplus_{t\in\bar{\omega}_m}\bigoplus_{i\in\omega_{m-1}^G} f_i\right]\oplus\left[\bigoplus_{t\in\bar{\omega}_m} f_t\right]\right| \le \left|\left(\bigoplus_{i\in\omega_{m-1}^G} f_i\right)\oplus f_p\right| \tag{4.2.14}$$

$$\left|\left[m\otimes\left(\bigoplus_{i\in\omega_{m-1}^G} f_i\right)\right]\oplus\left[\bigoplus_{t\in\bar{\omega}_m} f_t\right]\right| \le \left|\left(\bigoplus_{t\in\omega_m^G} f_i\right)\right| = F_m^G \tag{4.2.15}$$

$$\left|m\otimes\left(\bigoplus_{i\in\omega_{m-1}^G} f_i\right)\right| = \left|\left(\bigoplus_{i\in\omega_{m-1}^G} f_i\right)\right| = F_{m-1}^G \tag{4.2.16}$$

(4.2.15) ve (4.2.16) den

$$F_m^G \le F_{m-1}^G \tag{4.2.17}$$

eşitsizliği elde edilir. (4.2.17) ve Önerme 4.1 4 ten

$$\left|\bigoplus_{t\in\bar{\omega}_m} f_t\right| \le \left|\bigoplus_{i\in\omega_m^G} f_i\right| \tag{4.2.18}$$

eşitsizliği elde edilir. Ancak (4.2.18) kabulumuz olan (4.2.11) ile çelişir. Böylece bu durum için ispat tamamlanmış olur.

İkinci olarak ise $\tilde{\omega}_m \cap \omega_m^G \neq \varnothing$ olduğunu kabul edelim. Sadelik adına sadece $m=3$ olma durumunu göz önnde bulunduralım. Bu tür ispatı $m<n$ olan bütün m ler için yazabiliriz. $\tilde{\omega}_m = \{i_1, i_2, i_3\}$, $\omega_3^G = \{1,2,3\}$, $\omega_2^G = \{1,2\}$ olsun. Genelliği bozmadan $i_1 = 1$ yani $\tilde{\omega}_m \cap \omega_2^G = 1$, $i_2 = p$ ve

$$\left|\left(\bigoplus_{i\in\omega_2^G} f_i\right)\oplus f_{i_3}\right| \le \left|\left(\bigoplus_{i\in\omega_2^G} f_i\right)\oplus f_{i_2}\right| \tag{4.2.19}$$

dir. ω_{k+1}^G nin tanımından

$$\left|\left(\underset{i\in\omega_2^G}{\oplus} f_i\right)\oplus f_{i_2}\right| \le \left|\left(\underset{i\in\omega_2^G}{\oplus} f_i\right)\oplus f_3\right| = \left|\underset{i\in\omega_3^G}{\oplus} f_i\right| = F_3^G \tag{4.2.20}$$

$\oplus$ nın değişme ve birleşme özelliklerini kullanarak (4.2.16) den

$$\left|\left[(f_1\oplus f_2)\oplus f_2\right]\oplus\left[f_1\oplus\left(f_{i_2}\oplus f_{i_3}\right)\right]\right| = \left|\left[(f_1\oplus f_2)\oplus f_{i_2}\right]\oplus\left[(f_1\oplus f_2)\oplus f_{i_3}\right]\right| \tag{4.2.21}$$

elde edilir. (4.2.21) de Önerme 3.3.1 uygulanırsa

$$\left|\left[(f_1\oplus f_2)\oplus f_{i_2}\right]\oplus\left[(f_1\oplus f_2)\oplus f_{i_3}\right]\right| \le \left|(f_1\oplus f_2)\oplus f_{i_2}\right| \tag{4.2.22}$$

Teoremdeki $\left|f_2\ominus f_3\right| \notin U_{f_2 f_3}$ koşulundan

$$\left|f_1\oplus f_{i_2}\oplus f_{i_3}\right| \ge \left|f_1\oplus f_{i_2}\oplus f_3\right| \tag{4.2.23}$$

elde edilir ve (4.2.22) ile (4.2.23) göz önünde bulundurularak

$$\left|f_1\oplus f_{i_2}\oplus f_{i_3}\right| \le \left|f_1\oplus f_{i_2}\oplus f_3\right| \tag{4.2.24}$$

bulunur. Bu ise (4.2.23) ile çelişir. ■

4.3 Kesirler Cebiri Üzerine Bazı Önermeler

Global kritere göre tanımlanan toplama olan F_k^G ile Lokal kritere göre tanımlanan toplama olan F_k^L arasındaki bağlantıyı bilmek önemlidir. $\left|F_2^G\right| > \left|F_2^L\right|$ iken $\left|F_3^G\right| < \left|F_3^L\right|$ olduğunu hesaplamıştık. Ayrıca bu bölüm boyunca $f_i = \frac{a_i}{b_i}$ olarak alınacaktır.

Önerme 4.3.1 Her $k<n$ için $B_k^G = \sum_{i\in\omega_k^G} b_i$ ve $B_k^L = \sum_{i\in\omega_k^L} b_i$ şeklinde tanımlı ise $B_k^G \le B_k^L$ dir.

İspat İspatı tümevarım üzerinden yaparsak, $k=1$ için $\omega_1^G = 1 = \omega_1^L$ olmak üzere

$$B_1^G = \sum_{i\in\omega_1^G} b_i = \sum_{i\in\omega_1^L} b_i = B_1^L$$

olduğundan önermenin doğruluğu açıktır.

$k=m$ için $B_m^G = \sum_{i\in\omega_m^G} b_i = b_{i_1} + b_{i_2} + ... + b_{i_m}$ ve $B_m^L = \sum_{i\in\omega_m^L} b_i = b_1 + b_2 + ... + b_m$

olduğunu kabul edelim.

$k=m+1$ için önermeyi ispat edelim. Lokal ve Global stratejiye göre toplamanın tanımından $B_{m+1}^L = B_m^L + b_{m+1}$ ve $B_{m+1}^G = B_m + b_{i_{m+1}}$ dir.

(i) Eğer $i_{m+1} = m+1$ ise $B_{m+1}^G \le B_{m+1}^L$ dir. Çünkü hipotezimizde eşitsizliği $k=m$ için kabul etmiştik. Bu durumda her iki tarafa $m+1$ eklenirse eşitsizlik bozulmaz.

(ii) $i_{m+1} < m+1$ ise $i_{m+1} = t$ olacak şekilde $t \in \omega_m^G$ vardır. O zaman $i_t \in \omega_m^L$ dir. $i_t < m+1$ ise $i_t = s$ olacak şekilde $s \in \omega_m^G$ vardır. Bu durumda $i_s \in \omega_m^L$ dir. $i_s > m+1$ ise $\left| f_{i_t} \ominus f_t \right| \in U_{f_{i_t} f_t}$, $\left| f_{i_s} \ominus f_s \right| \in U_{f_{i_s} f_s}$ olduğu açıktır. Bu durumda $b_{i_s} < b_s$ dir, sonuç olarak $B_{m+1}^G \le B_{m+1}^L$ elde edilir.

(iii) $i_{m+1} > m+1$ ise $\left| f_{i_{m+1}} \ominus f_{m+1} \right| \in U_{f_{i_{m+1}} f_{m+1}}$ dir. Bu durumda $b_{m+1} > b_{i_{m+1}}$ olur. Sonuç olarak $B_{m+1}^G \le B_{m+1}^L$ elde edilir. ■

Önerme 4.3.2 Her $k \le n$ için $\left|F_k^G\right| \ge \left|f_k\right|$ ve $B_k^G > b_k$ dır.

İspat Teoremin ikinci kısmı açıktır. $\left|F_k^G\right| \ge \left|f_k\right|$ nın ispatını tümevarım yöntemi ile yapalım.

$k=1$ için Global stratejiye göre toplamının tanımından $F_1^G = f_1$ dir. Dolayısıyla $\left|F_1^G\right| = \left|f_1\right|$ olur.

$k=m$ için $\left|F_m^G\right| \geq \left|f_m\right|$ olur. $k=m+1$ için $\left|F_{m+1}^G\right| \geq \left|f_{m+1}\right|$ eşitsizliğinin doğruluğunu gösterelim. $F_{m+1}^G = F_m^G \oplus f_s$ olacak şekilde bir f_s vardır.

i) Eğer $s=m+1$ ise Önerme 4.3.1 den $\left|F_{m+1}\right| \geq \left|f_{m+1}\right|$ olur.

ii) Eğer $s \neq m+1$ ise $m+1 \in w_m^*$ dır. Tersini kabul edelim, yani $\left|F_m^G \oplus f_s\right| < \left|f_{m+1}\right|$ olsun. Tümevarıma göre

$$\left|F_m^G\right| \geq \left|f_m\right| \geq \left|f_{m+1}\right| \tag{4.3.1}$$

ve

$$\left|F_m^G \oplus f_{m+1}\right| \geq \left|f_{m+1}\right| \tag{4.3.2}$$

dir.

(4.3.1) ve (4.3.2) den $\left|F_m^G \oplus f_{m+1}\right| > \left|f_m^G \oplus f_s\right|$ elde edilir. Bu ise kabulumuz ile çelişir. ■

Önerme 4.3.3 Her $k<n$ için $\left|F_k^G\right| \geq \left|F_{k+1}^G\right|$ ve $B_k^G \leq B_{k+1}^G$ dir. Yani $\left\{F_k^G\right\}_{k=1}^n$ monoton azalan, $\left\{B_k^G\right\}_{k=1}^n$ monoton artandır.

İspat Teoremin ikinci kısmı açıktır. Birinci kısmını tümevarımla ispatlayalım. $k=1$ için $F_1^G = f_1$, $F_2^G = f_1 \oplus f_s$ olacak şekilde f_s elemanı vardır. Önerme 3.3.1 den $\left|f_1\right| \geq \left|f_1 \oplus f_s\right|$ dir. Buradan $F_1^G \geq F_2$ elde edilir.

$k=m$ için $\left|F_m^G\right| \geq \left|F_{m+1}^G\right|$ olsun. $k=m+1$ için eşitsizliğinin doğru olduğunu gösterelim. Yani, $\left|F_m^G\right| \geq \left|f_s\right|$ olduğunu gösterirsek Önermeden 3.3.1 den

$\left|F_m^G\right| \geq \left|F_m^G \oplus f_s\right|$ dir. Buradan $\left|F_m^G\right| \geq \left|F_{m+1}^G\right|$ elde edilir. $\left|F_m^G\right| \geq |f_s|$ olduğunu ispatlarken,

i) $s \geq m$ olsun. Önerme 4.3.2 den $\left|F_m^G\right| \geq |f_m| \geq |f_s|$ bulunur.

ii) $s < m$ için $\left|F_m^G\right| < |f_s|$ tir. $\bar{F}_m$ ise

$$\bar{F}_m = F_{m-1}^G \oplus f_s \tag{4.3.3}$$

olarak tanımlansın. Kabulumuzden

$$F_{m-1}^G \geq F_m^G \tag{4.3.4}$$

dir. (4.3.3), (4.3.4) eşitsizlikleri ve Önerme 4.3.1 kullanılarak,

$$\left|F_{m-1}^G \oplus f_s\right| \geq \left|F_m^G\right| \tag{4.3.5}$$

elde edilir. Yani $\bar{F}_m \geq F_m^G$ dir. Bu ise F_m^G nin maksimum olması ile çelişir. Dolayısıyla $\left|F_m^G\right| \geq |f_s|$ eşitsizliği bulunur. ■

Önerme 4.3.4 Her $\bar{\omega}_k, \tilde{\omega}_k \in \omega_n$ için $\max_{i \in \bar{\omega}_k}\{i\} < \min_{i \in \tilde{\omega}_k}\{i\}$ ise $\left|\bigoplus_{i \in \bar{\omega}_k} f_i\right| \geq \left|\bigoplus_{i \in \tilde{\omega}_k} f_i\right|$ dir.

İspat $\max_{i \in \bar{\omega}_k}\{i\} = i_1$ ve $\min_{i \in \tilde{\omega}_k}\{i\} = i_2$ olsun. Önerme 4.3.1 den

$$\left|\bigoplus_{i \in \bar{\omega}_k} f_i\right| \geq f_{i_1} \tag{4.3.6}$$

elde edilir.

Önerme 4.3.1 den

$$\left| \bigoplus_{i \in \tilde{\omega}_k} f_i \right| \geq f_{i_2} \tag{4.3.7}$$

bulunur. $\max_{i \in \tilde{\omega}_k} \{i\} < \min_{i \in \tilde{\omega}_k} \{i\}$ olduğundan $\left| f_{i_1} \right| \geq \left| f_{i_2} \right|$ olduğu görülür. (4.3.6) ve (4.3.7) den istenilen eşitsizlik elde edilir. ■

5 SIRT ÇANTASI PROBLEMİ VE BU PROBLEMİN ÇÖZÜMLERİNİN BULUNMASI İÇİN SEZGİSEL STRATEJİLERİN KURULMASI

5.1 Sırt Çantası Problemi

Sırt çantası problemleri tamsayılı programlama problemleri içinde en basit türlerinden biridir. Bunun nedeniyse problemin tanımının anlaşılabilir olması ve gerçek hayatta karşılaşılan birçok durumun bu probleme bağlı olarak kolayca ifade edilebilmesidir.

Son yıllarda sırt çantası problemleri yoğun olarak çalışılmaktadır. Bunun nedeni teorisyenler açısından problemin basit yapısı olup uygulama alanındakiler içinse bir çok endüstriyel duruma uygulanabilmesidir. Örneğin sermaye bütçeleme, proje seçimi, kargo yükleme, kesme gibi problemler KP' nin en klasik uygulamalarıdır ve bu problem çerçevesinde çözülebilirler.

Sırt çantası problemi, bir maksimizasyon problemi için en basit prototip olması nedeniyle yüz yıllardır çalışılmaktadır. 1897 yılında, çeşitli kısıtların nasıl tek bir sırt çantası kısıtında toplanabildiğini gösterilmiştir. Bu, genel bir tamsayılı problemin sırt çantası problemine indirgemesinin ilk örneklerinden biridir ve dolayısıyla sırt çantası problemini çözmenin en az bir tamsayılı problemi çözmek kadar zor olduğunu ispatlar. Martello ve Toth (1979) 0/1 KP için kesin algoritmalar ve bunların ortalama performanslarını araştırmışlar sonrasında ise bu çalışmayı diğer KP türleri ve yaklaşık algoritmalar için genişletmişlerdir.

Sırt Çantası Problemi Nedir?

- Bir tur için sırt çantasını hazırlayan ve hangi eşyaları yanına alması gerektiğini belirlemek zorunda olan bir dağcı düşünün. Tur için kullanışlı olabilecek bir sürü nesne ve kapasitesi sınırlı olan bir de çanta var. Bu nesnelerin her biri 1'den n'ye kadar numaralandırılmış ve her birinin sağladığı yarar $p_i > 0$ sayısıyla, ağırlıkları ise $w_i > 0$ sayısıyla ölçülüyor. Bu kişi çantasını maksimum faydayla ve çanta kapasitesi olan 'c' yi aşmayacak şekilde doldurmak istiyor. Hangi eşyaları almalı?

- Bir yatırımcının elinde “c” kadar belirli miktarda parası olsun ve bu parayı da kazanç sağlayabileceği işlere yatırmak istesin. Kararları için öncelikle olası yatırımların uzun bir listesini hazırlıyor ki burada w_i, o işe yatırması gereken para; p_i ise o işten beklenen net kazanç olarak belirtiliyor (Burada risk durumu hesaba katılmıyor.). Nereye yatırım yapmalı?

Yukarıda anlatılan her iki problem de KP şeklinde modellenen problemlerdir. Problemin formal tanımı ise aşağıdaki gibidir:

Yanımıza almak istediğimiz *n* adet nesne olsun ve bu nesneleri *i* ile indeksleyelim, $I = \{1, 2, ..., n\}$,

p_i : i. nesnenin sağladığı fayda,

w_i : i. nesnenin ağırlığı,

c: Sırt çantasının toplam kapasitesi

olsun. (Genellikle tüm bu değerler pozitif tamsayı olarak alınır.) Burada amaç, nesneler kümesinden, toplam faydanın maksimum olduğu ve toplam ağırlığın c kapasitesini geçmediği bir alt küme seçmektir.

Kısıtlar:

$$\sum_{i \in I} w_i x_i \leq c \qquad (5.1.1)$$

$$x_i \in \{0,1\}, \quad i \in I \qquad (5.1.2)$$

Amaç fonksiyonu:

$$KP = \sum_{i \in I} p_i x_i \qquad (5.1.3)$$

Karar değişkenlerinin tanımı bakımından sırt çantası problemini genel olarak üç türde sınıflandırabiliriz:

1. 0/1 Sırt Çantası Problemleri

2. Negatif olmayan tamsayılı sırt çantası problemleri (Kısaca tamsayılı diyeceğiz.)

3. Sınırlı tamsayılı sırt çantası problemleri

Yukarıdaki formülasyon birinci tür KP problemidir. İkinci türdeki sırt çantası problemleri içinse (5.1.2) kısıtı yerine

$$x_i \geq 0 \text{ ve } tamsayı,\ i \in I \qquad (5.1.2)'$$

kısıtı kullanılır. (Bu problem *sınırsız tamsayılı KP* olarak da bilinir.)

Verilen problemde çantaya yerleştirilecek nesnelerin sayıları sınırlandırılmış olabilir. Bu durumda problem sınırlı tamsayılı KP' dir ve bu modelde (5.1.2) kısıtı,

$$0 \leq x_i \leq b_i \text{ ve } tamsayı,\ i \in I \qquad (5.1.2)''$$

kısıtıyla yer değiştirir ve dolayısıyla problem çözümünde her hangi bir x_i nesnesi en fazla b_i değeriyle yer alır.

Problem içerdiği kısıt sayısı bakımından ise

1. Bir boyutlu sırt çantası problemleri

2. Çok boyutlu sırt çantası problemleri

olarak sınıflandırılır.

Eğer Sırt çantası Problemin' de sadece bir tane kısıt varsa bu tür problemler "Bir boyutlu sırt çantası problemi" olarak adlandırılır; yukarıdaki formülasyon ve tanım bu türe örnektir.

Varsayalım ki her bir maddenin hem ağırlığı hem de çantada kaplayacağı hacim değeri verilmiş olsun. Dağcının çantası taşıyacağı ağırlık bakımından sınırlı olduğu gibi hacim bakımından da sınırlı olsun. Dolayısıyla problem şimdi toplam hacim ve toplam ağırlık bakımından iki kısıt altındadır. Bu türden bir modelin genel formu, m tane kısıt içerecek şekilde aşağıdaki gibi tanımlanabilir:

$$\sum_{i \in I} w_{ji} x_i \leq c_j, \qquad j = 1, \ldots, m$$

$$x_i \in \{0,1\}, \qquad i \in I$$

k.a.

$$enb\ Z^* = \sum_{i \in I} p_i x_i$$

Yukarıda bahsedilen Sırt Çantası Problemi çeşitlerinin yanı sıra çalışılan bazı diğer sırt çantası problemleri aşağıdaki gibidir:

- Minimizasyon sırt çantası problemi
- Katlı sırt çantası problemi
- Çok seçmeli sırt çantası problemi
- Kuadratik sırt çantası problemi
- Çok amaçlı sırt çantası problemi

Özel olarak, minimizasyon Sırt Çantası Problemi için sonlu sayıdaki nesneler kümesinin öyle bir alt kümesi seçilmelidir ki nesnelerin toplam değeri

minimum ve toplam ağırlık en az *d* kadar olsun. Bu durumda yeni model 0/1 tamsayılı için aşağıdaki gibi olur:

$$\sum_{i \in I} w_i y_i \geq d$$

$$y_i \in \{0,1\}, \quad i \in I$$

k.a.

$$enk\ Z^* = \sum_{i \in I} p_i y_i$$

İlerleyen bölümlerdeki Sırt Çantası Probleminin türleri için ele alınan algoritmalarda aşağıdaki sıralamalar göz önünde bulundurulacaktır:

Birim faydalarına göre artmayan sıralama:

$$\frac{p_1}{w_1} \geq \frac{p_2}{w_2} \geq ... \geq \frac{p_n}{w_n} \qquad (5.1.3)$$

Birim faydalarına göre azalmayan sıralama:

$$\frac{p_1}{w_1} \leq \frac{p_2}{w_2} \leq ... \leq \frac{p_n}{w_n} \qquad (5.1.4)$$

5.2 Sırt Çantası Probleminin Çözümü için Sezgisel Stratejiler

Sırt Çantası probleminde nesneler kesinlikle kesilip biçilemez veya parçalanamaz. Bir nesne çantaya ya koyulur ya da koyulamaz. Bu durumda ikili (binary, boole) bir karar durumu söz konusudur ve x_i değişkenleri 0/1 türünden tanımlanır. Bu haliyle problem 0/1 Sırt Çantası (Knapsack) Problemi olarak bilinir (Başkaya, 2005). $A_n = \{a_1, a_2, ..., a_n\}$ gibi *n* sayıda eşya verilmek üzere bu eşyaların indis kümesi $\omega_n = \{1, 2, ..., n\}$ ile gösterilsin. $i \in \omega_n$ olmak üzere w_i, a_i eşyasının ağırlığını ve p_i de a_i eşyasının değerini göstersin. Bu problemde,

kapasitesi c olan çanta, toplam değeri maksimal olacak şekilde yüklemek isteniyor. $i \in \omega_n$ olmak üzere x_i değişkenleri aşağıdaki gibi tanımlansın:

$$x_i = \begin{cases} 1, & eğer\ i.eşya\ çantaya\ yüklenirse \\ 0, & aksi\ halde \end{cases}$$

Bu gösterimler altında 0/1 değişkenli tek boyutlu çanta probleminin matematiksel modeli aşağıdaki gibi olur :

$$\sum_{i \in \omega_n} p_i x_i \to max \tag{5.2.1}$$

$$\sum_{i \in \omega_n} w_i x_i \le c \tag{5.2.2}$$

$$x_i = \{0,1\}, \quad i \in \omega_n \tag{5.2.3}$$

Burada, $c > 0,\ p_i > 0,\ w_i > 0 \quad w_i \le c,\ i \in \omega_n, \sum_{i \in \omega_n} w_i > c$ olduğu kabul edilir.

Sırt çantası probleminin çözümü için sezgisel algoritmalar yazılmıştır. Bu bölümde yazılan algoritmalar incelenmiş, bu yolla herhangi bir baz çözüm alınarak optimalliği kontrol edilerek optimal olması durumunda amaç fonksiyonunun değeri daha iyi olan başka bir uygun çözüm bulunmuştur. Alınan çözümün optimalliği için bazı yeterli koşullar getirilmiştir.

Sonraki bölümlerde anlatımı kolaylaştırmak için, bazı tanım ve gösterimler aşağıda verilmiştir.

Tanım 5.2.1 $v_i = \dfrac{p_i}{w_i}$ olmak üzere, bu gösterimi a_i elemanının *lokal verimliliği* olarak adlandıralım.

Tanım 5.2.2 A_n kümesinin k sayıdaki elemanlar kümesini $\langle k \rangle$ - takım olarak adlandıralım ve A_k ile gösterelim. $A_k = \{a_{i_1}, a_{i_2}, ..., a_{i_k}\}$ ve $A_k \subset A_n$ olmak üzere uygun indisler kümesini de ω_k ile gösterelim. $\omega_k = \{i_1, i_2, ..., i_k\}$ olmak üzere $\omega_k \subset \omega_n$ dir.

Tanım 5.2.3 $v_{A_k} = \dfrac{\sum_{i\in\omega_k} p_i}{\sum_{i\in\omega_k} w_i}$ olmak üzere, v_{A_k} ' yı $\langle k\rangle -$ *takımın verimliliği* veya *global verimliliği* olarak adlandıralım.

Tanım 5.2.4 (5.2.3) koşulunu sağlayan $x=(x_1,x_2,...,x_n)$ boolean vektörüne *çözüm*, bu çözüm (5.2.2) koşulunu sağlıyorsa *uygun çözüm*; (5.2.1) ifadesi, uygun çözüm için maksimal değer alıyorsa *optimal çözüm* olarak adlandırılır.

G Stratejisi Yukarıdaki sırt çantası probleminin çözümünde, global verimliliğin maksimum olması için öyle bir A_r^G $\langle r\rangle$ - takımı seçilmelidir yani, $v_{A_r^G} = max\left\{v_{A_r} \mid A_r \subset A_n,\ c_r \le c,\ c_r + w_{i_{r+1}} > c\right\}$ dir.

G stratejinin doğruluğunu gösteren bir önerme verelim.

Önerme 5.2.1 G stratejisine uygun olarak seçilmiş A_r^G $\langle r\rangle$ - takımı için $\sum_{i\in\omega_r^G} w_i = c_r^G \le c$ koşulu sağlansın. Bu durumda A_r^G , yukarıdaki sırt çantası probleminin son kısıtının bu koşul ile değiştirildiğinde oluşan yeni problemin çözümü olur.

$v_{A_r^G}$ nin tanımına göre

$$v_{A_r^G} = \frac{\sum_{i\in\omega_r^G} p_i}{\sum_{i\in\omega_r^G} w_i} = max\left\{\frac{\sum_{i\in\omega_j} p_i}{\sum_{\substack{i\in\omega_j \\ \omega_j\subset\omega_n \\ c_j\le c_r^G}} w_i}\right\}$$

dir. Bu yüzden, $P_r^G = \sum_{i\in\omega_r^G} p_i = v_{A_r^G}.\sum_{i\in\omega_r^G} w_i = v_{A_r^G}.c_r^G$ maksimal değer olacaktır.

Sonuç 5.2.1 Yukarıdaki önermenin koşulları altında $c_r^G = c$ olma durumunda A_r^G sırt çantası probleminin kesin çözümüdür.

$v_{A_r^G} = max\left\{v_{A_r} \mid A_r \subset A_n,\ c_r \le c,\ c_r + w_{i_{r+1}} > c\right\}$ sezgisel kriteri globaldir, yani $\langle k \rangle$ - takım seçilirken, daha önce seçilmiş $\langle k-1 \rangle$ takım dikkate alınmaz. Bu nedenle A_r^G'yi bulmak için $N^G = C_n^r = \dfrac{n!}{r!(n-r)!}$ sayıda takımı incelemek gerekir. Bu durumun, algoritmanın çalışma zamanını uzattığı açıktır. Bu nedenle de aşağıdaki strateji önerilir.

GA stratejisi A_r^{GA}'yi hesaplama sürecinde, A_k^{GA} $\langle k \rangle$ – takımının seçilmiş olduğunu varsayalım. A_{k+1}^{GA} de, $\langle k+1 \rangle$ – takımının seçimi için global verimliliği maksimal artıracak şekilde bir eleman seçilmelidir, yani

$$v_{A_{k+1}^{GA}} = \max_{i \in \omega_n \setminus \omega_k^{GA}} \left\{ \frac{\sum_{j \in \omega_k^{GA}} p_j + p_i}{\sum_{j \in \omega_k^{GA}} w_j + w_i} \right\}$$

dır. A_r^{GA} yaklaşık değerinin GA stratejisine göre hesaplanması global-ardışık stratejidir (Nikitin and Nuriev, 1982) (uygun değişkenleri üst indiste GA olarak gösterelim). Böylece GA stratejiye uygun olarak birinci verimliliği aşağıdaki gibi belirlenen eleman seçilir.

$$v_{A_1^{GA}} = v_{i_1} = \max_{i \in \omega_n} \left\{ \frac{p_i}{w_i} \right\} = \frac{p_{i_1}}{w_{i_1}} \text{ yani } A_1^{GA} = \left\{a_{i_1}\right\},\ \omega_1^{GA} = \left\{i_1\right\},$$

$$v_{A_2^{GA}} = \max_{i \in \omega_n \setminus \omega_1^{GA}} \left\{ \frac{p_{i_1} + p_i}{w_{i_1} + w_i} \right\} = \frac{p_{i_1} + p_{i_2}}{w_{i_1} + w_{i_2}}, \text{ yani } A_2^{GA} = \left\{a_{i_1}, a_{i_2}\right\},\ \omega_2^{GA} = \left\{i_1, i_2\right\}$$

A_r^{GA}'nı bulmak için maksimum $N^{GA} = \dfrac{n(n+1)}{2}$ durum incelenir.

Yukarıdaki çanta probleminin yaklaşık çözümünü bulmak için GA strateji ile birlikte lokal strateji (LA) kullanılır.

LA Stratejisi Eğer a_{i_1} ve a_{i_2} elemanlardan a_{i_1} elemanı daha yüksek lokal verimliliğe sahip ise A_{k+1}^{LA} de a_{i_2} elemanının yerine a_{i_1} elemanı kullanılması global verimliliği iyileştirir. Yani, eğer $v_{i_1} \geq v_{i_2}$ ise $v_{\omega_k^{LA} \cup i_1} \geq v_{\omega_k^{LA} \cup i_2}$ dir.

Önerme 5.2.2 A_q^{LA}, LA gtratejisine uygun olarak seçilmiş $\langle q \rangle$ - takım ve

$$\sum_{i \in \omega_q^{LA}} w_i = c_q^{LA} \leq c \tag{5.2.7}$$

olsun. O zaman $A_q{}^{LA}$, (5.2.1), (5.2.7), (5.2.3) probleminin çözümüdür.

İspat Önermenin tersini, yani

$$\sum_{i \in \omega_t} w_i = c_t \leq c_q^{LA} \tag{5.2.8}$$

ve

$$P_t > P_q^{LA} \tag{5.2.9}$$

olacak şekilde öyle $\langle t \rangle$ - takımının var olduğunu kabul edelim. $\omega_q^{LA} \cap \omega_t = \omega_{qt}$, $\bar{\omega}_q = \omega_q^{LA} \setminus \omega_{qt}$, $\bar{\omega}_t = \bar{\omega}_t \setminus \omega_{qt}$ ve $P_{qt} = \sum_{i \in \omega_{qt}} p_i$ tanımlamalarına göre

$$\bar{P}_q = \sum_{i \in \bar{\omega}_q} p_i = \sum_{i \in \bar{\omega}_q} v_i w_i \geq v_q \sum_{i \in \bar{\omega}_q} w_i \geq v_q \sum_{i \in \bar{\omega}_t} w_i \geq \sum_{i \in \bar{\omega}_t} v_i w_i = \bar{P}_t \Rightarrow \bar{P}_q \geq \bar{P}_t$$
$$\Rightarrow P_q^{LA} = \bar{P}_q + P_{qt} \geq \bar{P}_t + P_{qt} = P_t,$$

olduğu açıktır.

Böylece, $P_q^{LA} \geq P_t$ dır. Bu ise, (5.2.9) ile çelişkidir. ■

Tanım 5.2.5 A_q^{LA} (A_q^{GA})'yı, LA (GA) startejisine göre *baz (temel) çözüm* adını verelim.

Tanım 5.2.6 $\Delta c_q^{LA} = c - c_q^{LA}$ ($\Delta c_p^{GA} = c - c_p^{GA}$) LA (GA) startejisine göre $A_q{}^{LA}\left(A_p{}^{GA}\right)$ çözümünün *uyuşmazlığı* olarak adlandıralım.

(5.2.1)-(5.2.3) problemi ile birlikte (5.2.1), (5.2.2), (5.2.10) problemini de ele alalım. Bu problem sırt çantası problemindeki tamsayılı olma koşulunun çıkarılması ile elde edilen yeni problem olur.

$$0 \le x_i \le 1,\ i \in \omega_n \qquad (5.2.10)$$

Doğrusal programlama teorisine göre aşağıdaki Önerme doğrudur.

Önerme 5.2.3 $\tilde{a}_{q+1}$, a_{q+1} eşyasının bir kısmını ifade etmek üzere $\tilde{w}_{q+1} = \left| w_{q+1} - (w_{q+1} - \Delta c_q^{LA}) \right| = \left| \Delta c_q^{LA} \right|$, $\tilde{p}_{q+1} = \tilde{w}_{q+1}.v_{q+1}$ dir. $\tilde{A}_{q+1} = A_q^{LA} \cup \tilde{a}_{q+1}$ (5.2.1), (5.2.2), (5.2.10) probleminin optimal çözümüdür.

Gerçekten de, önerme 5.2.2 den A_q^{LA} (5.2.1), (5.2.2), (5.2.7) problemin çözümüdür ve $v_{q+1} = \max_{i \in \omega_n / \omega_q^{LA}} \{v_i\}$ dir. O zaman $\tilde{P}_{q+1} = P_q^{LA} + v_{q+1}\tilde{w}_{q+1}$ maksimal olur.

(5.2.1)-(5.2.3) probleminin keyfi uygun çözümünün amaç fonksiyonun değeri (5.2.1), (5.2.2), (5.2.10) probleminin optimal çözümünün amaç fonksiyonunun değerinden büyük olmayacağı açıktır yani, $P_q^{LA} \le \tilde{P}_{q+1}$, $P_q^{GA} \le \tilde{P}_{q+1}$ dir.

$\tilde{P}_{q+1}$, (5.2.1)-(5.2.3) probleminin çözümü için üst sınır olur. $\tilde{P}_{q+1}$ i *teorik optimum* (TO) olarak adlandıralım.

5.3 Sırt Çantası Probleminin Çözümünü İyileştirmek için Değişim Koşulları

Δc_q^{LA} nın sıfırdan farklı olması, Önerme 5.2.2 den A_q^{LA} 'nın optimalliğini garanti etmez. $\Delta c_q^{LA} \neq 0$ ise *(q+1).* eşyanın seçilmesi doğru değildir. Çünkü maksimal verimliliği olan a_{q+1}, (5.2.3) koşulunu bozacaktır. Bu yüzden de, (5.2.3) koşulunu daha uygun olan ve amaç fonksiyonun değeri bulunandan az olmayacak şekilde bir çözüm seçilecektir. Uyuşmazlığın, (5.2.1)-(5.2.3) probleminin amaç fonksiyonunun değerini kötüleştirmeden üç yolla azaltıldığını kolaylıkla görebiliriz.

1. Çantaya $A_n \setminus A_q^{LA}$'dan ağırlığı Δc_q^{LA}'dan çok olmayan elemanları eklemek (Açgözlü Algoritmalarda Baz çözümü iyileştirmek için bu yöntem kullanılır).

2. A_q^{LA}'dan ayrı ayrı elemanları $A_n \setminus A_q^{LA}$'dan ayrı ayrı elemanları ile değişim yapmak (verimliliklerinin küçük olmasına rağmen birinci elemanların değeri ikinci elemanların değerlerinden büyük olmalıdır).

3. A_q^{LA}'dan $\langle t \rangle$- takımı $A_n \setminus A_q^{LA}$ $\langle s \rangle$-takımıyla değişim yapmak (Nikitin and Nuriev, 1983) çalışmasında dinamik programlama kullanılarak bu yöntem uygulanmıştır).

Birinci ve ikinci yöntem lokal özelliğe sahiptir ve büyük hesaplama zamanı gerektirmez. Üçüncü yöntem ise, global özelliğe sahiptir, yani kombinatör tiplidir ve büyük hesaplama zamanı gerektirir. Bu nedenle biz birinci ve ikinci yöntemleri kullanmayı tercih edeceğiz.

a_i, a_j elemanların değişim koşullarını inceleyelim:

$$a_i \Leftrightarrow a_j,\ a_i \in A_q^{LA},\ a_j \in A_n \setminus A_q^{LA} \Rightarrow v_i > v_j$$

Pozitif değere sahip değişimleri ele alacağız, yani

$$\Delta P_{ji} > 0,\ \Delta P_{ji} = P_j - P_i > 0 \Rightarrow w_j > w_i \Rightarrow w_j = w_i + \Delta w_{ji},$$

$$P_j - P_i = w_j v_j - w_i v_i = (w_i + \Delta w_{ji}) v_j - w_i v_i = w_i v_j + \Delta w_{ji} v_j - w_i v_i$$

$$= w_i (v_j - v_i) + \Delta w_{ji} v_j > 0 \Rightarrow \Delta w_{ji} v_j > w_i (v_i - v_j)$$

$$\frac{\Delta w_{ji}}{w_i} > \frac{v_i - v_j}{v_j} = \frac{v_i}{v_j} - 1 \qquad (5.3.1)$$

(5.3.1) koşulunu *değişim koşulu* olarak adlandıralım .

Önerme 5.3.1 Eğer $a_i \in A_q^{LA}$ ve $a_j \in A_n \setminus A_q^{LA}$ (5.3.1) koşulunu sağlıyorsa ve $\Delta w_{ji} < \Delta c_q^{LA}$, o zaman $a_i \Leftrightarrow a_j$ değişimi (5.2.1)-(5.2.3) probleminin amaç fonksiyonunu iyileştirir.

(5.3.1) değişim koşuluna benzer eşitsizlikler (Veliev and Mamedov, 1981) çalışmasında değişkenleri sayısını azaltmak için önerilmiştir.

İspat. Kesirler Cebirinden (Gürsoy et al., 2011) açıktır. ■

Önerme 5.3.2 $A_t \subset A_q^{LA}, A_s \subset A_n \setminus A_q^{LA}$ olsun, o zaman $v_{A_t} \geq v_{A_s}$ dir.

Önerme 5.3.1 ve Önerme 5.3.2 ten aşağıdaki teorem elde edilir.

Teorem 5.3.1 Eğer $A_t \subset A_q^{LA}, A_s \subset A_n \setminus A_q^{LA}$ aşağıdaki koşulunu sağlıyor ise,

$$\frac{c_{A_s} - c_{A_t}}{c_{A_t}} > \frac{v_{A_t}}{v_{A_s}} - 1 \qquad (5.3.1)'$$

ve $c_{A_s} - c_{A_t} \leq \Delta c_q^{LA}$ o zaman $A_t \Leftrightarrow A_s$ değişimi (5.2.1)-(5.2.3) problemin amaç fonksiyonunu iyileştirir.

Teorem 5.3.2 (A_q^{LA} 'nin optimallılık koşulu) $w^* = \min\limits_{k \in \omega_n \setminus \omega_q^{LA}} \{w_k\}$, $w^{**} = \max\limits_{k \in \omega_q^{LA}} \{w_k\}$ olmak üzere eğer,

$$\Delta c_q^{LA} < w^* \qquad (5.3.2)$$

ve

$$\frac{\Delta c_q^{LA}}{w^{**}} < \frac{v_q}{v_{q+1}} - 1 \qquad (5.3.3)$$

ise A_q^{LA}, (5.2.1)-(5.2.3) probleminin optimal çözümüdür.

Her bir $\langle t \rangle, \langle s \rangle$ -takımları için

$$\frac{\Delta c_q^{LA}}{w^{**}} < \max_{\substack{i \in \omega_q^{LA} \\ j \in \omega_n \setminus \omega_q^{LA}}} \left\{ \frac{\Delta w_{ji}}{w_i} \right\} \qquad (5.3.4)$$

ve

$$\frac{v_q}{v_{q+1}} = \min_{\substack{i \in \omega_q^{LA} \\ j \in \omega_n \setminus \omega_q^{LA}}} \left\{ \frac{v_i}{v_j} \right\} \qquad (5.3.5)$$

eşitlikleri doğrudur. $\left(A_t \subset A_q^{LA}, A_s \subset A_n \setminus A_q^{LA},\ A_t \Leftrightarrow A_s\right)$ (5.3.2)'den birinci değişim yöntemi uygulanamaz. (5.3.3), (5.3.4), (5.3.5) ve Teorem 5.3.1 den hiç bir değişimin pozitif değere sahip olmadığı çıkar. Yani, (5.2.1)-(5.2.3) probleminin amaç fonksiyonunun iyileştirilmesinin imkanı yoktur.

5.4 Sırt Çantası Probleminin Çözümünün Bulunması için Önerilen Algoritmalar

Sırt Çantası probleminin çözümünü bulmak için LA ve GA stratejilerine dayanan algoritmalar önerilmiş ve onların farklı kombinasyonlarından oluşan yeni algoritmalar geliştirilmiştir.

Aşağıda, (5.1.1)-(5.1.3) probleminin çözümünü bulmak için LA stratejisine dayanan K_1 algoritması tasarlanmıştır.

K_1 Algoritması

1. $k = \max\left\{ l \,\middle|\, \sum_{i=1}^{l} w_i \le c \right\}$

2. $p = \sum_{i=1}^{k} p_i\ ,\ w = \sum_{i=1}^{k} w_i$

3. $i = k + 2$

4. while $i \le n$ do

5. If $w_i \le c - w$ then $p = p + p_i\ ,\ w = w + w_i$

6. $i = i + 1$

7. od

8. $K_1 = p$

Burada 2. adımdan sonra problemin LA stratejisine göre *temel çözümü* bulunur, 4-7 adımlarında, bulunmuş çözüm çantaya koyulmamış nesneler dikkate alınarak, LA stratejisi ile iyileştirilir.

Aşağıda, (5.2.1)-(5.2.3) probleminin çözümü bulmak için GA stratejisine dayanan K_2 algoritması tasarlanmıştır:

K_2 Algoritması

1. $p=0,\ w=0,\ i=1,\ j=2$

2. while 1 do

3. If $p_i \geq p_j$ then $k=i,\ i=j$

4. else $k=j$

5. If $w_k \leq c-w$ then $p=p+p_k,\ w=w+w_k,\ j=j+1$

6. else *exit do*

7. od

8. $j=j+1$

9. While $j \leq n$ do

10. If $p_i \geq p_j$ then $k=i,\ i=j$

11. else $k=j$

12. If $w_k \leq c-w$ then $p=p+p_k,\ w=w+w_k$

13. $j=j+1$

14. od

15. If $w_i \leq c-w$ then $K_2=p+p_i$

16. else $K_2=p$

Burada 5. adımdan sonra problemin GA stratejisine göre *temel çözümü* bulunur, 9-14 adımlarında bulunmuş çözüm çantaya koyulmamış nesneler dikkate alınarak, GA stratejisi ile iyileştirilir.

Böylece her iki algoritmada da, önce bir temel çözüm bulunur ve sonra bu çözüm iyileştirilmeye çalışılır. Bu durum göz önüne alınarak aşağıda oluşturulan K_3 algoritmasında, temel çözümü bulmak için LA strateji, iyileştirmek için ise GA strateji kullanılmıştır.

K_3 **Algoritması:**

1. $k = \max\left\{ l \left| \sum_{i=1}^{l} w_i \le c \right. \right\}$

2. $p = \sum_{i=1}^{k} p_i \,,\; w = \sum_{i=1}^{k} w_i$

3. $i = k+1,\; j = i$

4. $j = j+1$

5. While $j \le n$ do

6. If $p_i \ge p_j$ then $k = i,\; i = j$

7. else $k = j$

8. If $w_k \le c - w$ then $p = p + p_k \,,\; w = w + w_k$

9. $j = j+1$

10. od

11. If $w_i \le c - w$ then $K_3 = p + p_i$

12. else $K_3 = p$

Aşağıda oluşturulan K_4 algoritmasında ise, K_3 algoritmasını aksine, temel çözümü bulmak için GA stratejisi, iyileştirme için ise LA stratejisi kullanılmıştır.

K_4 **Algoritması:**

1. $p=0,\ w=0,\ i=1,\ j=2$

2. while 1 do

3. If $p_i \geq p_j$ then $k=i,\ i=j$

4. else $k=j$

5. If $w_k \leq c-w$ then $p=p+p_k,\ w=w+w_k,\ j=j+1$

6. else *exit do*

7. od

8. while $i \leq n$ do

9. If $w_i \leq c-w$ then $p=p+p_i,\ w=w+w_i$

10. $i=i+1$

11. od

12. $K_4=p$

6 SIFIR-BİR SIRT ÇANTASI PROBLEMLERİ İÇİN GARANTİ DEĞERLİ GREEDY ALGORİTMALARI

Bu bölümde klasik 0-1 sırt çantası probleminin minimizasyon ve maksimizasyon versiyonları ele alınmıştır. Bu problemler için greedy algoritmaları ve algoritmaların garanti değerleri hesaplanmaları incelenmiştir. Bilinen garanti değerli algoritmalara değinerek 0/1 maksimizasyon ve minimizasyon problemleri için yeni algoritmalar önerilmiş, hesaplama denemeleri yapılarak bu algoritmaların önceki algoritmalarla kıyaslanması yapılmıştır. Yapılmış hesaplama denemelerinin sonuçları önerilmiş algoritmaların verimliliğinin yüksek olduğunu göstermektedir

6.1 Sürekli Maksimizasyon Sırt Çantası Problemi

Herhangi bir tamsayılı programlama probleminin yapısını daha iyi anlayabilmek ve çözüm değerinin ilk tahmin değerini elde edebilmek için standart bir yaklaşım, tamlık şartının bırakılmasıdır. 0-1 KP' nin doğrusal programlama gevşetmesi; yani *sürekli sırt çantası problemi, LKP,* aşağıdaki gibi tamlık şartının kaldırılmasıyla elde edilir.

$$LKP = max\left\{ \sum_{i \in N} p_i x_i \;\middle|\; \sum_{i \in N} w_i x_i \le c,\ 0 \le x_i \le 1,\ \ i \in N \right\}$$

LKP çözümü basit bir yolla hesaplanabilir; burada nesneler birim faydalarına göre artmayan sırada çantaya dahil edilir. Dolayısıyla her adımda, birim ağırlık başına düşen kar en yüksek eleman seçilir. Problem için önerilen algoritma aşağıda verilmiştir ve algoritmada (5.1.3) sıralamasının mevcut olduğu kabul edilmektedir.

$A(LKP)$ Algoritması

1. $\sum_{j=1}^{k} w_j \le c < \sum_{j=1}^{k+1} w_j$ durumunu sağlayan k indisi bulunur.

2. $x_j = 1, \quad j = \overline{1,k}$

$$x_{k+1} = \frac{1}{w_{k+1}} \left(c - \sum_{j=1}^{k} w_j \right)$$

$$x_j = 0, \quad j = \overline{k+2,n}$$

$$LKP = \sum_{j=1}^{k} p_j + \frac{p_{k+1}}{w_{k+1}} \left(c - \sum_{j=1}^{k} w_j \right)$$

3. SON.

$A(LKP)$ algoritmasının bulduğu çözüm vektörü, $X = (x_1, x_2, ..., x_n)$, aynı zamanda optimal çözüm vektörüdür ve optimal çözüm değeri aşağıdaki gibidir:

$$LKP = \sum_{j=1}^{k} p_j + \frac{p_{k+1}}{w_{k+1}} \left(c - \sum_{j=1}^{k} w_j \right)$$

6.2 Sıfır - Bir Maksimizasyon Sırt Çantası Problemi (KP) ve Garanti Değerli A_1 Algoritması

$$KP = max \left\{ \sum_{i \in N} p_i x_i \;\middle|\; \sum_{i \in N} w_i x_i \le c,\ x_i \in \{0,1\},\ \ i \in N \right\} \qquad (6.2.1)$$

Greedy algoritması ile sürekli problemi optimal olarak çözebildiğimizi göstermiştik. *KP* değerini bulmak için ilk olarak, bir önceki algoritmayı kullanarak elde edilen kesirli değişkenin tamsayıya yuvarlanması akla gelebilir.

KP için ilk yaklaşım şemaları Ibarra ve Kim (1975), Sahni (1975) tarafından kurulmuştur. Bütün bu algoritmalar dinamik programlamaya dayanır ve giriş verilerinin "ölçeklendirilmesi" tekniğinde birbirlerinden ayrılırlar.

0/1 maksimizasyon sırt çantası problemi için $\Delta \geq 1/2$ garanti değerli Greedy algoritmaları bulunabilir. Bu algoritmalardan biri aşağıdaki gibi verilmiştir (Papadimitriou and Steiglitz, 1982). Burada yine değişkenlerin (5.1.3)'deki gibi sıralı olduğu kabul edilir.

A_1 Algoritması

1. $\sum_{i=1}^{k} w_i \leq c < \sum_{i=1}^{k+1} w_i$ durumunu sağlayan *k* indisini bulunur.

2. $A_1 = max\left\{\sum_{i=1}^{k} p_i, p_{k+1}\right\}$

3. SON.

Teorem 6.2.1 Sıfır-bir maksimizasyon sırt çantası için A_1 algoritmasının garanti değeri $\Delta_1 = 1/2$, yani

$$\frac{1}{2} KP \leq A_1 \leq KP$$

İspat $S = \sum_{i=1}^{k} p_i$ olsun. Bu durumda $A_1 = max\{S, p_{k+1}\}$ dir.

$$SS = S + p_{k+1}. \tag{6.2.2}$$

Açıktır ki;

$$A_1 < KP \leq LKP \leq SS \tag{6.2.3}$$

Bu durumda (6.2.2) ve (6.2.3)' den

$$\Rightarrow A_1 \geq \frac{1}{2} SS \geq \frac{1}{2} KP$$

$$\Rightarrow KP \geq A_1 \geq \frac{1}{2} KP \quad \Rightarrow \frac{A_1}{KP} \geq \frac{1}{2}$$

6.3 Sürekli Minimizasyon Sırt Çantası Problemi

Sıfır-bir minimizasyon sırt çantası probleminin doğrusal programlama gevşetmesi; yani sürekli minimizasyon sırt çantası problemi, LKM, tüm değişkenlerin sadece 0/1 değerlerini alma şartı yerine, bu aralıktaki reel sayı değerlerinin de seçilebilmesiyle elde edilir:

$$LKM = min\left\{ \sum_{j\in N} p_j x_j \;\middle|\; \sum_{j\in N} m_j x_j \geq c,\ 0 \leq x_j \leq 1,\ j \in N \right\}$$

LKM nin çözümü *LKP* çözümüne benzer şekilde basit olarak hesaplanabilir: Nesneler birim katkılarına göre azalmayan sırada çantaya alınır; böylece her adımda her bir ağırlık biriminden en düşük fayda sağlanır. Bu problem için önerilen greedy algoritması aşağıda verilmiştir; burada (5.1.4) sıralaması kabul edilmektedir.

$A(LKM)$ Algoritması

1. $\sum_{j=1}^{k} w_j < c \leq \sum_{j=1}^{k+1} w_j$ durumunu sağlayan k indisini bulunur.

2. $x_j = 1, \qquad j = 1, 2, ..., k$

$$x_{k+1} = \frac{1}{w_{k+1}} \left(c - \sum_{j=1}^{k} w_j \right)$$

$$x_j = 0, \qquad j = k+2, k+3, ..., n$$

$$LKM = \sum_{j=1}^{k} p_j + \frac{p_{k+1}}{w_{k+1}} \left(c - \sum_{j=1}^{k} w_j \right)$$

3. SON.

$A(LKM)$ algoritmasının bulduğu çözüm vektörü, $X=(x_1,x_2,...,x_n)$, aynı zamanda optimal çözüm vektörüdür ve optimal çözüm değeri aşağıdaki gibidir:

$$LKM=\sum_{j=1}^{k}p_j+\frac{p_{k+1}}{w_{k+1}}\left(c-\sum_{j=1}^{k}w_j\right)$$

İspatı maksimizasyon versiyonuna benzer şekilde, sıralamanın değiştiği de göz önünde bulundurularak yapılabilir.

6.4 Sıfır-Bir Minimizasyon Sırt Çantası Problemi (KM) ve Garanti Değerli A_2 Algoritması

$$KM=min\left\{\sum_{j\in N}p_jx_j\;\middle|\;\sum_{j\in N}w_jx_j\geq c,\ x_j\in\{0,1\},\ j\in N\right\}$$

KP probleminde sırt çantasına girmeyen eşyaların kazançlarının, $d=\sum_{i=1}^{n}w_i-c$ nin durumuna bağlı olacak şekilde en küçüklenmesi problemi olarak ifade edebilir. Bu tanımlama ile yukarıda tanımlanan sırt çantası problemi minimizasyonu (KM) aşağıdaki gibi belirtilmiş olur (Kellerer and Pferschy, 2004).

$$KM=min\left\{\sum_{i=1}^{n}p_iy_i\;\middle|\;\sum_{i=1}^{n}w_iy_i\geq d,\ y_i\in\{0,1\},\ i=1,\ldots,n\right\}$$

KM' deki y_i değişkenleri hangi eşyaların sırt çantasına girmeyeceğini belirtir. Bunlar maksimizasyon KP' deki çantanın içine girmeyen eşyalardır. Sonuç olarak; KP' deki x_i sonuçlarının tersi ile KM' deki $y_i:=1-x_i$ $(i=1,...,n)$' leri tanımlamış oluruz.

A_2 Algoritması

1. $A_2 = \sum_{i=1}^{n} p_i$ olsun.

2. $k = min\left\{ l \left| \sum_{i=1}^{l} w_i \geq d \right. \right\}$ bulunur.

3. $L = \sum_{i=1}^{k} p_i$

4. $A_2 = \min\{A_2, L\}, \quad J = J \setminus k$

5. $\sum_{i \in J} w_i \geq d$ ise 2 'ye geç

6. Son.

Teorem 6.4.1 Sıfır-bir minimizasyon sırt çantası problemi için $KM \leq A_2 \leq 2KM$ dir.

İspat Teoremin ispatı (Gens and Levner, 1980a, 1980b) çalışmalarında şu şekilde verilmiştir. $X^* = \{ x_1^*, ..., x_n^* \}$ problemin optimal çözüm değeri; S = $\{ i \mid x_i^* = 1 \}$ ve R algoritmanın sonucunda elde edilen küme olsun.

$\sum_{i \notin R} w_j < c$ olduğu için S kümesinde R kümesinden en az bir eleman vardır. Böyle elemanlardan birincisinin indisi q olsun. Algoritmamızda q elemanının R kümesine ilk defa girdiği adıma bakalım ve bu adımdaki küme $R' \subseteq R$ olsun. Bu durumda;

$$L = \sum_{i \in N_1} p_i + p_q = \sum_{i \in N_2} p_i + \sum_{i \in N_3} p_i + p_q,$$

olur. Burada,

$$N_1 = \{ i \mid (i < q) \wedge (i \notin R') \},$$
$$N_2 = \{ i \mid (i < q) \wedge (i \in S) \},$$
$$N_3 = \{ i \mid (i < q) \wedge (i \in S) \wedge (i \notin R') \}$$

Ele aldığımız adımda $A_2 \le \sum_{i \in N_2} p_i + \sum_{i \in N_3} p_i + p_q$ olur ve bu değer bir sonraki adımda artmaz. Optimal çözüm değeri:

$$KM = \sum_{i \in N_2} p_i + \sum_{i \in N_4} p_i + p_q .$$

Burada $N_4 = \{ i \,|\, (i > q) \wedge (i \in S) \}$ kümesini belirtir.

İki durum söz konusudur:

(i) $p_q \ge \sum_{i \in N_2} p_i + \sum_{i \in N_3} p_i$

Bu durumda $p_q \le KM$ (yani q indisli eleman optimal çözüme giriyor) ve

$$KM \le A_2 \le \sum_{I \in N_2} p_i + \sum_{I \in N_3} p_i + p_q \le 2p_q \le 2KM$$

Buradan, $KM \le A_2 \le 2KM$ olduğu görülür.

Dikkat edilirse, eğer optimal çözümde yalnız bir değişkenin değeri 1'e eşitse; yani $x_q^* = 1$ ise o zaman **(i)** durumu geçerlidir. Başka bir deyişle;

$$\sum_{i \in N_2} p_i + \sum_{i \in N_3} p_i \le \left(\sum_{i \in N_2} w_i + \sum_{i \in N_3} w_i \right) \frac{p_q}{w_q} < c \cdot \frac{p_q}{w_q} \le p_q$$

(ii) $p_q < \sum_{i \in N_2} p_i + \sum_{i \in N_3} p_i$

Tüm elemanlar (5.1.4)'e göre sıralanmıştır; ayrıca

$$\sum_{i \in N_2} w_i + \sum_{i \in N_3} w_i < c ; \qquad (6.4.1)$$

$$\sum_{i \in N_2} w_i + \sum_{i \in N_4} w_i + w_q \ge c ; \qquad (6.4.2)$$

Burada (6.4.1) ve (6.4.2)'den $\sum_{i\in N_3} w_i < \sum_{i\in N_4} w_i + w_q$ olduğu görülür. (5.1.4) sıralamasından dolayı

$$\frac{\sum_{i\in N_3} p_i}{\sum_{i\in N_3} w_i} \le \frac{p_q}{w_q} \le \frac{\sum_{i\in N_4} p_i}{\sum_{i\in N_4} w_i} \quad \text{ve} \sum_{i\in N_3} p_i = \sum_{i\in N_3} w_i \cdot \frac{\sum_{i\in N_3} p_i}{\sum_{i\in N_3} w_i} < \left(\sum_{i\in N_4} w_i + w_q\right) \cdot \frac{\sum_{i\in N_3} p_i}{\sum_{i\in N_3} w_i} \le \sum_{i\in N_4} p_i + p_q$$

eşitsizlikleri de mevcuttur. Bu durumda,

$$\frac{A_2}{2} \le \frac{1}{2}\left(\sum_{i\in N_2} p_i + \sum_{i\in N_3} p_i + p_q\right) < \left(\sum_{i\in N_2} p_i + \sum_{i\in N_3} p_i\right) < \sum_{i\in N_2} p_i + \sum_{i\in N_4} p_i + p_q = KM$$

Sonuç olarak görülür ki;

$$KM \le A_2 \le 2KM$$

dir. ■

Bu problem için garanti değeri 2 olan bir başka algoritma da (Güntzer and Jungnickel, 2000) çalışmasında önerilmiştir.

6.5 Sıfır - Bir Minimizasyon Sırt Çantası Problemi için Garanti Değerli A_3 Algoritması

Güntzer and Jungnickel (2000), KP ve KM yi çözmek için bir minimasyon Algoritması (A_3) önermiştir. Greedy algoritmasına benzer olarak çözüm, eşyaların ağırlık başına kazanç oranlarına göre seçilmesiyle oluşur, yani (5.1.4) koşulunun sağlandığı varsayılır. Algoritma geçici bir çözümle başlar, burada tüm karar değişkenleri 0 olarak ayarlanır ve tüm nesneler işlem görmemiş olarak işaretlenir. Daha sonra tüm nesneler işlem görene kadar yerleştirme ve tamamlama aşamaları tekrarlanır.

Yerleştirme aşaması en ağır işlenmemiş nesnelerin yerleşimi uygun bir çözüm üretene kadar tüm işlenmemiş nesneleri verilen sırada birer birer

yerleştirerek başlar. Takip eden tamamlama aşamasında güncel çözüme katkısı uygun bir çözüme ulaştıracak tüm nesneler düşünülür. Bu şekilde oluşturulmuş her yeni çözüm o zamana kadar elde edilmiş en iyi çözümle karşılaştırılır ve daha iyiyse saklanır. Yerleştirme veya tamamlama aşamalarında kullanılan bir nesne işlenmiş olarak işaretlenir.

Son olarak, o ana kadarki en iyi çözüm, toplam ağırlığı ve tamamlayıcı nesnesiyle saklanır. Final çözümü belirlemek için tamamlayıcı nesne çantaya yerleştirilir ve diğer nesneler final ağırlığı geçmeyecek şekilde verilen sıraya göre birer birer yerleştirilir.

A_3 Algoritması

1: Başla

2: $w = 0, p = 0$ [Geçerli Çözüm]

3: $\bar{w} = \sum_{i=1}^{n} w_i$; $\bar{p} = \sum_{i=1}^{n} p_i$ [Şimdiye kadar en iyi çözüm]

4: $k = 1$ [Tamamlama Eşyası]

5: $S = \{1, 2, \ldots, n\}$ [Eşyaleri işlenmemiş olarak ayarla]

6: *While* $S \neq 0$ *and* $p < \bar{p}$ *do*

7: *While* $max\{w_i : i \in S\} < d - w$ [Yerleştirme Evresi]

8: $J = minS, w = w + w_j, p = p + p_j, \mathrm{S} := \mathrm{S} \setminus \{\mathrm{j}\}$

9: *od*

10: *While* $max\{w_i : i \in S\} \geq d - w$ *do* [Tamamlama Evresi]

11: Find j with $w_j = \max\{w_j : j \in S\}$; $S = S \setminus \{j\}$

12: *If* $p + p_j < \bar{p}$ *then* $\bar{w} = w + w_j$; $\bar{p} := p + p_j; k = j$ fi

13: *od*

14: *od*

Teorem 6.5.1 A_3 algoritması KM için 1-yaklaşık; KP için $1/2$ yaklaşıktır.

İspat KM için ispatlayalım. j, $y_1, y_2, \ldots, y_n$ ilk uygun çözümünü tamamlayan nesne olsun ve p^* optimal fayda olsun. Nesnelerin sıralaması nedeniyle

$$\sum_{i=1}^{j-1} p_i y_i < p^*$$

olur.

Eğer $p_j \le p^*$ ise $\sum_{i=1}^{j-1} p_i y_i + p_j < 2p^*$ olduğundan $\varepsilon = 1$ açıktır.

Aksi halde, j nesnesi optimal çözümü değiştirmeksizin KM örneğinden çıkarılabilir. Bu nesne, tamamlama için kullanıldığından o anki çözüme yerleştirilemez. Dolayısıyla, A_3 ün final çözüm değeri, d kapasitesi ve $1,\ldots,j-1,j+1,\ldots,n$ nesnelerinden oluşan problem örneğine uygulanan A_3 algoritmasının çözüm değerini geçmez.

İspat tümevarım yöntemiyle bu şekilde tamamlanır.

KP için ispatlayalım. $j = min\left\{ i = 1,\ldots,n : \sum_{k=1}^{i} m_k \ge d \right\}$, kritik nesne, $x_1, x_2, \ldots, x_n$ A_3 algoritmasını KP çözümü ve p^* optimal fayda olsun. Sıralamadan dolayı

$$\sum_{j=1}^{n} p_j > p^*$$

dir. Açıktır ki, A_3 algoritması KM için j nesnesiyle tamamlanmış bir çözüm bulur ki bu da KP için en az

$$\sum_{i=1}^{n} p_i x_i \ge \sum_{i=j+1}^{n} p_i$$

dir. Faydalı bir çözüme denk gelir. $m_i \le c$ şartından dolayı

$$d \le d + c - m_j = \sum_{\substack{i=1 \\ i \ne j}}^{n} m_i$$

dır. Dolayısıyla, A_3 tarafından bulunan en az bir değer uygun çözüm vardır (j nesnesiyle tamamlanan çözümden önce yada sonra) Bu çözümde j nesnesi KM çantasına yerleştirilmiştir ve dolayısıyla buna ilişkin KP çantası en az j nesnesini içerir.

$$\sum_{i=1}^{n} p_i x_i \ge p_j$$

Son olarak

$$\sum_{i=1}^{n} p_i x_i \ge max\left\{ \sum_{i=j+1}^{n} p_i p_j \right\} \ge \frac{1}{2}\left(\sum_{i=j+1}^{n} p_i + p_j \right) > \frac{1}{2} p^*$$

olur ve ispat tamamlanır.

KM için bu oran sıkıdır.

$d = k$, $m_1 = p_1 = k-2$, $m_2 = k-1$, $p_2 = k$, $m_3 = k-1$, $p_3 = k$, $m_4 = 1$, $p_4 = 2$ problem serisi için optimal fayda açıktır ki $p^* = k + 2$ ve A_3 nin çözümü $p = 2k - 2$ dir. $\frac{(p - p^*)}{p^*} \xrightarrow{m \to \infty} 1$ dir. $\left(\frac{2k-2-k-2}{k+2} \to 1 \right)$

KP için de $\frac{1}{2}$ oranı sıkıdır. Yukarıdaki problem serisi ile çalışırsak, $c = \sum_{i=1}^{n} m_i - d = 2k - 3$ kapasitesi için optimal fayda $p^* = 2k - 2$ ‘dır. A_3 , $p = k + 2$ sonucunu verir ve bu durumda

$$\lim_{m\to\infty}\frac{p^*-p}{p^*}=\frac{2k-2-k-2}{2k-2}\to\frac{1}{2}$$

olur. ■

Altını çizelim ki, KP ve KM problemlerinin çözümleri arasındaki ilişki ilk defa Nuriyev (1986) yapmıştır. Burada KM problemi KP'nin tümleyen problemi gibi tanımlanmıştır.

İki versiyondan biri için olan kesin algoritmalar diğeri için de kesin bir çözüm verirken bu durum ε - yaklaşık algoritmalar için uygulanamaz.

A_1 ve A_2 algoritmalarının bulduğu çözümler ile KP ve KM problemlerinin optimal çözümleri arasında aşağıdaki bağıntılar bulunmaktadır:

$$A_1 \le KP \le 2A_1$$

$$\frac{1}{2}A_2 \le KM \le A_2 .$$

$P=\sum_{i=1}^{n} p_i$ olmak üzere $KP=\alpha_2 P$ ve $KM=\alpha_1 P$ olacak şekilde α_1 ve α_2 olduğunu kabul edelim. O zaman aşağıdaki teoremler doğrudur (Nuriyev, 1986).

Teorem 6.5.2 A_2 algoritmasının bulduğu A_2 değeri aşağıdaki koşulları sağlamaktadır:

$$A_2 \le \begin{cases} 2KM, & eğer\ \alpha_1 < \frac{1}{3} \\ \frac{\alpha_1+1}{2\alpha_1}KM, & eğer\ \alpha_1 \ge \frac{1}{3} \end{cases}$$

Teorem 6.5.3 A_2 algoritmasının bulduğu $P-A_2$ değeri aşağıdaki koşulu sağlamaktadır:

$$P - A_2 \geq \begin{cases} \frac{1}{2} KP, & eğer\ \alpha_2 < \frac{2}{3} \\ \frac{2\alpha_2 - 1}{\alpha_2} KP, & eğer\ \alpha_2 \geq \frac{2}{3} \end{cases}$$

Tümleyen problem kavramı, tamsayılı programlama problemleri için de geliştirilmiş ve önerilmiş algoritmalar için uygun teoremler ispatlanmıştır (Guler et al., 2012).

6.6 Sıfır-Bir Minimizasyon Sırt Çantası Problemi için Kesirler Cebirine Dayanan A_4 Algoritması

Nikitin and Nuriyev (1982), Kesir–doğrusal Boole minimizasyon programlama problemlerini çözmek için kesirler üzerinde yeni işlemler tanımlanmış ve Gürsoy et al., (2011), bu işlemlerin kesirler kümesinde bir cebir oluşturduğu gösterilmiştir. Bu cebirde tanımlanmış toplama işlemine göre inverslik koşulu belirlenmiştir. Aşağıda bu inverslik koşulu göz önüne alınarak (5.1.4) sıralamasının olduğu da varsayılarak KM için bir greedy algoritma önerilmiştir. Bu algoritmada eşyaların seçimi zamanı inverslik koşulunu sağlayan elemanlara öncelik verilir:

A_4 ***Algoritması***

1. $pp = 0,\ ww = 0,\ i = 1,\ j = 2$

2. A: If $p_i < p_j$ then $k = i,\ i = j$

3. else $k = j$

4. $pp = pp + p_k,\ \ ww = ww + w_k$

5. $j = j + 1$

6. If $ww < d$ then go to A

7. else $pr = pp,\ wr = ww$

8. $j = j - 1$

9. B: $pp = pp + p_k,\ \ ww = ww + w_k$

10. C: $j = j + 1$

11. If $j > n$ then go to S

12. else If $p_i < p_j$ then $k = i,\ \ i = j$

13. else $k = j$

14. $pp = pp + p_k,\ \ ww = ww + w_k$

15. If $ww < d$ then go to C

16. If $pp < pr$ then $wr = ww,\ \ pr = pp$

17. go to B

18. S: $A_4 = pr$

19. Son

6.7 Sıfır-Bir Minimizasyon Sırt Çantası Problemi için Garanti Değerli A_5 Algoritması

Burada da (5.1.4) koşulunun sağlandığı varsayılarak A_2 Algoritması geliştirilerek A_5 algoritması oluşturulmuştur.

A_5 Algoritması:

1. $k = \min\left\{ l \mid \sum_{i=1}^{n} w_i \geq d \right\}$ indisini bul.

2. $p = \sum_{i=1}^{k} p_i, \quad w = \sum_{i=1}^{k} w_i, \quad pr = p, \ wr = w, \ i = i+1, \ j = i$

3. A: $p = p - p_k, \ w = w - w_k$

4. B: $j = j+1$

5. If $j > n$ then go to S

6. If $p_i < p_j$ then

7. If $w_i \leq d - w \ \& w_j \leq d - w$ then $k = i, \ i = j,$ go to C

8. If $w_i < d - w \ \& w_j > d - w$ then $k = j,$ go to D

9. If $w_i > d - w \ \& w_j \leq d - w$ then $k = i, \ i = j,$ go to D

10. If $w_i \geq d - w \ \& w_j > d - w$ then $k = i, \ i = j,$ go to D

11. else

12. If $w_i \leq d - w$ then $k = i, \ i = j,$ go to D

13. If $w_i > d - w \ \& w_j < d - w$ then $k = i, \ i = j,$ go to D

14. If $w_i > d - w \ \& w_j > d - w$ then $k = j$ go to D

15. C: $p = p + p_k, \ w = w + w_k,$ go to B

16. D: $p = p + p_k, \ w = w + w_k$

17. If $p \leq p_r$ then $pr = p, \ wr = w$

18. go to A

19. S: $A_5 = pr$

20. Son

Teorem 6.7.1 A_5 algoritmasının garanti değeri $\Delta_5 \leq 2$ dir, yani $KM \leq A_5 < 2KM$ dir.

İspat A_5 algoritmasının bulduğu çözüm hiç bir zaman A_2 algoritmasının bulduğu çözümden kötü değildir. Çünkü $j > i \geq k$ için $p_i < p_j$, $w_i < w_j$, $w_i < d - w$ ve $w_j \geq d - w$ durumunda A_2 algoritması önce *i.* eşyayı sonra *j.* eşyayı seçer, ancak A_5 algoritması yalnız *j*. eşyayı seçer ve bu seçimi A_2 algoritması hiçbir zaman yapamaz. $p + p_i + p_j > p + p_j$ olduğu için A_5 algoritması daha iyi çözüme (yani $A_5 \leq A_2$) ulaşabilir.

Bu durum dışında her iki algoritma aynı eşyaları seçer. Dolayısıyla A_2 algoritmasının bulduğu çözüm için (4.2) koşulu sağlandığından A_5 algoritmasının bulduğu çözüm için de (4.2) koşulu sağlanacak, yani $\Delta_5 \leq 2$ olacaktır. ■

7 SIFIR - BİR SIRT ÇANTASI PROBLEMİ KÜTÜPHANESİ

Ayrık optimizasyon algoritmalarının verimliliği, farklı problem örnekleri üzerinde hesaplama denemeleri yapılarak ve elde edilen sonuçlar karşılaştırılarak değerlendirilir. Bu örnek problemler başka araştırmacılar tarafından kolayca ulaşılabilmeli ve örnek problemlerin dışında kalan yazılım, donanım gibi diğer etmenlerin çalışmaları etkilememesi için dikkat edilmelidir.

Bir çok çalışmada, örnek problemlerin katsayıları rastgele sayılardır fakat test problemlerinin sonraki denemelerde de aynı değerlerle elde edilmesi koşulu ayrık optimizasyon problemleri için özellikle önemlidir. Sadece bir katsayının bir birim değişmesi, optimal çözümü bulmak için gereken çalışma zamanını çok değiştirdiği veya bulunmuş yaklaşık çözümün optimal çözümden farkının çok artırdığı bilinmektedir.

0/1 değişkenli Çanta Problemlerine yönelik algoritmaların verimliliğini araştırmak için KNAPSACK Test Problemleri Kütüphanesi hazırlanmıştır (http://fen.ege.edu.tr/~urfat/knapsack/). 0/1 Sırt Çantası problemleri için farklı özelliklerde 120 adet test problemi oluşturulmuş ve maksimizasyon ve minimizasyon olmak üzere iki başlık altında http://fen.ege.edu.tr/~urfat/knapsack adresinde erişime açılmıştır. Test Problemlerinin optimal çözümleri WinQSB (Bakır ve Altunkaynak, 2003; Başkaya, 2005) programı ile elde edilmiştir. Kütüphanede, problemlerin parametreleri (Ek 1), onların optimal çözümleri ve önerilmiş algoritmaların bu problemler için bulduğu çözümler detaylı şekilde verilmiştir.

7.1 Test Problemlerinin Oluşturulması

Çok sayıda test probleminin yaratılmasında ve yayınlanmasında teknik zorluklarla karşılaşılabilmektedir. Bu nedenle de test problemlerini oluşturmak için bir yöntem hazırlanması önemlidir. Bu anlamda çanta probleminin katsayıları için test problemleri üretmek üzere verilen m ve n için aşağıdaki formül verilmiştir (Babayev et al., 1978):

$$a_{ij} = \left[\left| \alpha_{ij} \right| \cdot 10^{1+r} \right] - \left[10^{r} \left[\left| \alpha_{ij} \right| \cdot 10^{1+r} \right] \Big/ 10^{r} \right] \tag{7.1.1}$$

$$\alpha_{ij} = \sin\left(\pi \cdot t\left(m - i + \theta\right) \cdot \frac{j}{i} \right) \cdot \sin\left(\pi \cdot t\left(n - j + \theta\right) \cdot \frac{i}{j} \right) \tag{7.1.2}$$

$$c_j = a_{m+1,\,j} \tag{7.1.3}$$

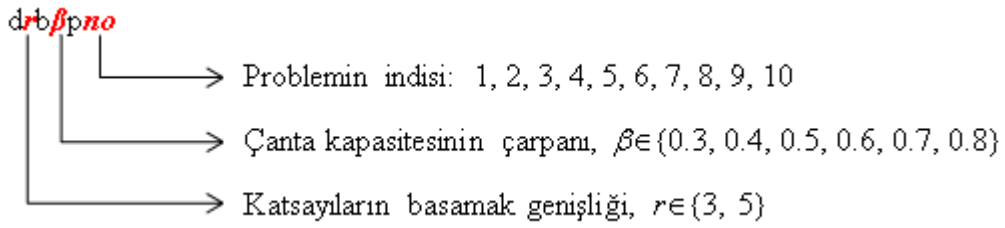
$$b_i = \left[\beta \cdot \sum_{j=1}^{n} a_{ij} \right] \tag{7.1.4}$$

$$0 < \beta < 1 \tag{7.1.5}$$

$$\left[z \right] : z\text{'nin tam kısmı} \tag{7.1.6}$$

7.1.2 deki, t doğal sayısı problemin numarasını gösterir. Eğer aynı boyuta sahip birden çok problem üretmek gerekirse, o zaman t, her problem için farklı değer alır ve aynı boyutlu farklı katsayılı problemler üretilir. $\theta \in Q$, sinüs fonksiyonunun argümanı $\frac{\pi}{2}$ nin katı olamayacak şekilde seçilir.

7.1.1 de, $0 < \left| \alpha_{ij} \right| < 1$ dir. Burada r basamak sayısı olmak üzere a_{ij} ve c_j , r basamaklı negatif olmayan tamsayılar olarak alınır. $\left| \alpha_{ij} \right|$ nin onluk tabandaki yazılışında, virgülden sonraki $\left(l+1\right)$. basamaktan başlayarak r tane sayı alır. Hesaplama denemeleri 7.1.1 formülünün beraber paylanmış rastgele sayılar ürettiğini gösterir. Burada, $\pi = 3.14$, $\theta = 1.33$, $l = 0$, $m = 1$ (problemlerin kısıt sayısı), $i = 1$, $j = 1, 2, ..., n$, $w_j = a_{1j}$, $p_j = a_{2j}$, $c = b_1$, $n = 100$ (problemlerdeki değişkenlerin sayısı), $r \in \{3, 5\}$ (problemlerdeki katsayılar için olabilecek en fazla basamak sayısı), $\beta \in \{0.3, 0.4, 0.5, 0.6, 0.7, 0.8\}$ (problemlerin sağ taraf parametrelerini (kapasite) belirleyen çarpan) ve $t = 1, \ldots, 120$ (problemlerin numarası) olarak kullanılır. Her bir r ve β için 10 Test Problemi oluşturulmuştur.



Şekil 7.1 Problem isimlendirme formatı

Ek 1'de 120 test problemine ait bilgiler listelenmiştir. Bu çizelgede (Ek 1) her bir probleme ait problem adı (d*r*b*β*p*no*) (Şekil 7.1), optimum değer (maksimizasyon için), eşyaların karlarının toplamı $\left(\mathrm{P}=\sum_{j=1}^{n} p_j\right)$, eşyaların ağırlıklarının toplamı $\left(\mathrm{W}=\sum_{j=1}^{n} w_j\right)$, çanta kapasitesi $\left(c=\beta\cdot\sum_{j=1}^{n} w_j\right)$ ve çanta kapasitesinin tümleyeni $\left(d=\sum_{j=1}^{n} w_j - c\right)$ bilgileri yer almaktadır.

Ayrıca her bir probleme ait tüm detaylar (katsayılar, optimal çözüm, algoritma çözümleri vb.) problemin sayfasında listelenmektedir (Çizelge 7.1, Çizelge 7.2, Çizelge 7.6 ve Çizelge 7.10).

Çizelge 7.1 *m*=1 ve *n*=10 için 1 kısıtlı 10 değişkenli örnek problem tamsayılı doğrusal programlama modeli (WinQSB girdisi olarak)

LP m1n10 10 1

Variable -->	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10		R.H.S.
Maximize (p_i)	442	735	988	765	522	22	316	425	998	662		
Weight (w_i)	733	767	25	424	664	27	763	881	83	324	<=	2345
LowerBound	0	0	0	0	0	0	0	0	0	0		
UpperBound	1	1	1	1	1	1	1	1	1	1		
VariableType	Binary	Binary	Binary	Binary	Binary	Binary	Binary	Binary	Binary	Binary		

Çizelge 7.2 Örnek problemin optimum çözümü (WinQSB çıktısı)

	Decision Variable	Solution Value	Unit Cost or Profit c(j)	Total Contribution	Reduced Cost	Basis Status
1	X1	0	442	0	442	at bound
2	X2	1	735	735	0	basic
3	X3	1	988	988	0	basic
4	X4	1	765	765	0	basic
5	X5	1	522	522	0	basic
6	X6	1	22	22,00	0	basic
7	X7	0	316	0	316	at bound
8	X8	0	425	0	425	at bound
9	X9	1	998	998	0	basic
10	X10	1	662	662	0	basic

Objective Function (Max.) =	4.692,00

	Constraint	Left Hand Side	Direction	Right Hand Side	Slack or Surplus	Shadow Price
1	C1	2.314,00	<=	2.345,00	31	0

7.2 Maksimizasyon 0/1 Sırt Çantası Problemi Alt Kütüphanesi

Maksimizasyon 0/1 Sırt Çantası Problemi için, Literatürden bildiğimiz K_1 algoritmasına ek olarak, Kesirler Cebirine dayanan K_2 Algoritması önerilmiş, K_1 ve K_2 algoritmalarının kombinasyonu şeklinde oluşturulan K_3 ve K_4 algoritmaları geliştirilmiştir.

K_1, K_2, K_3 ve K_4 algoritmaları ile test problemleri üzerinde hesaplama denemeleri yapılarak elde edilen sonuçlar kıyaslanmıştır. Hesaplama denemelerinin sonuçları detaylı olarak http://fen.ege.edu.tr/~urfat/knapsack/max/ adresinde bulunmaktadır. K_2 algoritmasının, optimal çözüme daha çok durumda ulaştığı çizelgelerden görülmektedir. Ancak K_3 algoritmasının bulduğu en iyi çözümlerin sayısı diğer algoritmalara göre daha fazladır. K_3 algoritması, K_1

Algoritmasının bulduğu temel çözümün K_2 Algoritmasıyla iyileştirilmesi sonucunda elde edilmiştir. Bu durum göz önüne alındığında, K_2 algoritması K_1 algoritmasına göre daha verimlidir.

Hesaplama denemelerinin sonuçları, basamak sayıları en çok 3 olan (1 - 999) parametreler için Çizelge 7.3'de, basamak sayıları en çok 5 (1 - 99999) için ise Çizelge 7.4'de verilmiştir. Çizelge 7.3 ve Çizelge 7.4'daki optimal sonuçlar gri renkle işaretlenmiştir. Bu sonuçlar Ek 2'de listelenmiş, listelenen bu sonuçlar özetlenerek Çizelge 7.5'de algoritmaların kıyaslanması verilmiştir.

Çizelge 7.3 Üç basamaklı parametreler (r=3) için hesaplama denemelerinin sonuçları

	β=0.3										β=0.4									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
K1	1	1	1	1	1	0	0	1	1	0	1	1	0	1	1	0	1	1	0	0
K2	1	1	1	1	1	0	0	1	1	0	1	1	0	1	1	0	1	1	0	1
K3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0
K4	1	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	1	0	0
	β=0.5										β=0.6									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
K1	0	1	1	0	1	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1
K2	1	0	1	0	1	1	1	1	1	0	1	1	1	1	1	1	0	1	1	1
K3	0	1	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
K4	1	1	1	0	0	0	0	1	0	0	1	0	1	0	0	0	1	0	0	1
	β=0.7										β=0.8									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
K1	1	1	1	0	1	0	1	1	1	1	1	1	1	1	0	1	1	0	1	1
K2	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1
K3	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
K4	0	1	0	0	1	0	0	0	0	0	0	1	1	1	0	0	0	1	1	0

* Gri renkli hücreler optimal sonuçlara ulaşan algoritma ve problemi, hücrelerdeki 1 değeri ise ilgili algoritmanın diğer algoritmalar arasında en iyi sonuca ulaştığını göstermektedir.

Çizelge 7.4 Beş basamaklı parametreler (r=5) için hesaplama denemelerinin sonuçları

	β=0.3										β=0.4									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
K1	1	1	1	0	1	1	1	0	1	1	1	0	0	1	1	1	1	1	1	1
K2	1	1	1	1	1	1	1	0	1	1	1	0	0	1	1	1	1	1	1	1
K3	1	1	0	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
K4	1	1	0	1	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0
	β=0.5										β=0.6									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
K1	1	1	0	1	1	1	0	0	1	0	1	0	0	1	1	1	1	1	1	1
K2	1	1	1	1	1	1	0	0	1	1	1	1	0	1	1	1	1	1	1	1
K3	1	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
K4	0	1	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0	1	1	0
	β=0.7										β=0.8									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
K1	0	0	1	1	1	1	0	0	1	1	1	1	1	0	1	1	1	1	1	1
K2	1	1	1	1	1	1	0	0	1	1	1	1	1	0	1	1	1	1	1	1
K3	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
K4	1	1	0	1	1	0	0	0	0	1	0	0	0	0	1	1	1	0	1	1

* Gri renkli hücreler optimal sonuçlara ulaşan algoritma ve problemi, hücrelerdeki 1 değeri ise ilgili algoritmanın diğer algoritmalar arasında en iyi sonuca ulaştığını göstermektedir.

Çizelge 7.5 K_1, K_2, K_3 ve K_4 algoritmalarının kıyaslanması

Algoritma	**İyi çözüm sayıları**	**%***	**Optimal çözüm sayıları**	**%***	**Ortalama yaklaşım oranı**
K1	90	75%	36	30%	0,9990
K2	99	83%	42	35%	0,9992
K3	105	88%	39	33%	0,9992
K4	42	35%	25	21%	0,9959

* İlgili sonuçların 120'ye oranı

Çizelge 7.6 K_1, K_2, K_3 ve K_4 algoritmalarının örnek problem (Çizelge 7.1) üzerindeki çözümleri

K1->Max	Ob_1	Ob_2	Ob_3	Ob_4	Ob_5	Ob_6	Ob_7	Ob_8	Ob_9	Ob_10
p_i:	988	998	662	765	735	22	522	442	425	316
w_i:	25	83	324	424	767	27	664	733	881	763
e_i:	39,5	12	2	1,8	1	0,8	0,8	0,6	0,5	0,4
First Order:	3	9	10	4	2	6	5	1	8	7
x_i:	1	1	1	1	1	1	1	0	0	0

Constraint: 2314 <= 2345 (c)

Objective Value: 4692

K2->Max	Ob_1	Ob_2	Ob_3	Ob_4	Ob_5	Ob_6	Ob_7	Ob_8	Ob_9	Ob_10
p_i:	988	998	662	765	735	22	522	442	425	316
w_i:	25	83	324	424	767	27	664	733	881	763
e_i:	39,5	12	2	1,8	1	0,8	0,8	0,6	0,5	0,4
First Order:	3	9	10	4	2	6	5	1	8	7
x_i:	1	1	1	1	1	1	1	0	0	0

Constraint: 2314 <= 2345 (c)

Objective Value: 4692

K3->Max	Ob_1	Ob_2	Ob_3	Ob_4	Ob_5	Ob_6	Ob_7	Ob_8	Ob_9	Ob_10
p_i:	988	998	662	765	735	22	522	442	425	316
w_i:	25	83	324	424	767	27	664	733	881	763
e_i:	39,5	12	2	1,8	1	0,8	0,8	0,6	0,5	0,4
First Order:	3	9	10	4	2	6	5	1	8	7
x_i:	1	1	1	1	1	1	1	0	0	0

Constraint: 2314 <= 2345 (c)

Objective Value: 4692

K4->Max	Ob_1	Ob_2	Ob_3	Ob_4	Ob_5	Ob_6	Ob_7	Ob_8	Ob_9	Ob_10
p_i:	988	998	662	765	735	22	522	442	425	316
w_i:	25	83	324	424	767	27	664	733	881	763
e_i:	39,5	12	2	1,8	1	0,8	0,8	0,6	0,5	0,4
First Order:	3	9	10	4	2	6	5	1	8	7
x_i:	1	1	1	1	1	1	1	0	0	0

Constraint: 2314 <= 2345 (c)

Objective Value: 4692

7.3 Minimizasyon 0/1 Çanta Problemi Alt Kütüphanesi

Minimizasyon 0/1 Çanta Problemi için garanti değerli algoritma tasarlamak Maksimizasyon 0/1 Çanta Problemine göre daha zordur. Minimizasyon 0/1 Çanta Problemi için tasarlanmış A_2 ve A_3 Algortimaları ele alınarak geliştirilmiş ve Kesirler Cebirine dayanan A_4 ve A_5 Algoritmaları önerilmiştir. Bu dört algoritma ile test problemleri üzerinde hesaplama denemeleri yapılarak kıyaslamalar yapılmıştır. Hesaplama denemelerinin sonuçlarına detaylı olarak http://fen.ege.edu.tr/~urfat/knapsack/min/ adresinden erişilebilir. Çizelge 7.7 ve Çizelge 7.8'de A_5 algoritmasının A_2 algoritmasından her zaman iyi çözüm bulduğu gözlemlenmiştir. Ayrıca A_5 algoritmasının garanti değerinin $\Delta_5 \leq 2$ olduğu gösterilmiştir. Çizelgelerden A_2 ve A_4 algoritmalarını bir birilerini tamamladığı da gözlemlenmişdir. Böylece bu iki algoritmanın kombinasyonu ile her zaman daha iyi çözüm edilebilir.

Çizelge 7.7 Üç basamaklı sayılar (r=3) için hesaplama denemeleri

	β=0.3										β=0.4									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
A2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
A4	0	0	0	0	0	1	0	1	0	0	1	0	1	1	0	1	1	1	1	0
A3	1	0	0	1	0	1	1	1	0	1	1	1	1	1	0	1	1	1	1	0
A5	1	1	1	1	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
	β=0.5										β=0.6									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
A2	1	1	1	1	1	1	0	1	0	0	1	1	1	1	1	1	1	1	1	1
A4	0	1	1	0	1	0	1	1	1	1	0	0	1	1	0	1	0	0	0	0
A3	1	1	1	1	1	0	0	1	0	0	1	0	1	1	0	1	0	0	1	1
A5	1	1	1	1	1	1	0	1	0	0	1	0	1	1	0	1	1	0	1	1
	β=0.7										β=0.8									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
A2	1	1	1	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
A4	0	1	1	1	1	1	0	1	0	0	1	1	1	1	1	0	0	0	0	0
A3	0	1	1	0	0	1	0	1	0	1	0	1	1	1	1	1	0	1	1	0
A5	0	1	1	0	0	1	0	1	0	1	0	1	1	1	1	1	0	1	1	0

* Gri renkli hücreler optimal sonuçlara ulaşan algoritma ve problemi, hücrelerdeki 1 değeri ise ilgili algoritmanın diğer algoritmalar arasında en iyi sonuca ulaştığını göstermektedir.

Hesaplama denemelerinin sonuçları, en çok 3 basamaklı sayılar (1 - 999) için Çizelge 7.7'da, en çok 5 basamaklı sayılar (1 - 99999) için ise Çizelge 7.8'de verilmiştir. Çizelge 7.7 ve Çizelge 7.8'deki optimal sonuçlar gri renkle işaretlenmiştir. Bu sonuçlar Ek 3'de listelenmiş, listelenen bu sonuçlar özetlenerek Çizelge 7.9'da algoritmaların kıyaslanması verilmiştir.

Çizelge 7.8 Beş basamaklı sayılar (r=5) için hesaplama denemeleri

	β=0.3										β=0.4									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
A2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
A4	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1	0
A3	0	1	1	0	1	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1
A5	0	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
	β=0.5										β=0.6									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
A2	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
A4	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0	1	0	1	0
A3	0	1	0	1	0	0	1	0	1	1	0	0	1	0	1	0	1	1	1	1
A5	1	1	0	1	1	0	1	0	1	1	0	0	1	0	1	0	1	1	1	1
	β=0.7										β=0.8									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
A2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
A4	1	1	1	1	0	0	1	0	1	0	0	0	0	1	0	1	0	1	1	1
A3	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	1	1	1	1	1
A5	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1

* Gri renkli hücreler optimal sonuçlara ulaşan algoritma ve problemi, hücrelerdeki 1 değeri ise ilgili algoritmanın diğer algoritmalar arasında en iyi sonuca ulaştığını göstermektedir.

Çizelge 7.9 A_2, A_3, A_4 ve A_5 algoritmalarının kıyaslanması

Algoritma	**İyi çözüm sayısı**	**%***	**Optimal çözüm sayısı**	**%***	**Ortalama yaklaşım oranı**	**En kötü yaklaşım oranları**				
A2	113	94%	50	42%	0,005	0,074	0,038	0,032	0,032	0,026
A4	51	43%	17	14%	0,017	0,093	0,074	0,067	0,065	0,062
A3	83	69%	35	29%	0,008	0,074	0,069	0,062	0,055	0,038
A5	93	78%	38	32%	0,007	0,074	0,062	0,038	0,032	0,034

* Tüm problemler içindeki oranları

Çizelge 7.10 A_2, A_3, A_4 ve A_5 algoritmalarının örnek problem (Çizelge 7.1) üzerindeki çözümleri

A2->Min	Ob_1	Ob_2	Ob_3	Ob_4	Ob_5	Ob_6	Ob_7	Ob_8	Ob_9	Ob_10
p_i:	316	425	442	522	22	735	765	662	998	988
w_i:	763	881	733	664	27	767	424	324	83	25
e_i:	0,41	0,48	0,6	0,79	0,81	0,96	1,8	2,04	12,02	39,52
First Order:	7	8	1	5	6	2	4	10	9	3
y_i:	1	1	1	0	0	0	0	0	0	0

Constraint: 2377 >= 2346 (d)

Objective Value: 1183

MinGreedy(A3)->Min	Ob_1	Ob_2	Ob_3	Ob_4	Ob_5	Ob_6	Ob_7	Ob_8	Ob_9	Ob_10
p_i:	316	425	442	522	22	735	765	662	998	988
w_i:	763	881	733	664	27	767	424	324	83	25
e_i:	0,41	0,48	0,6	0,79	0,81	0,96	1,8	2,04	12,02	39,52
First Order:	7	8	1	5	6	2	4	10	9	3
y_i:	1	1	1	0	0	0	0	0	0	0

Constraint: 2377 >= 2346 (d)

Objective Value: 1183

A4->Min	Ob_1	Ob_2	Ob_3	Ob_4	Ob_5	Ob_6	Ob_7	Ob_8	Ob_9	Ob_10
p_i:	316	425	442	522	22	735	765	662	998	988
w_i:	763	881	733	664	27	767	424	324	83	25
e_i:	0,41	0,48	0,6	0,79	0,81	0,96	1,8	2,04	12,02	39,52
First Order:	7	8	1	5	6	2	4	10	9	3
y_i:	1	1	1	0	0	0	0	0	0	0

Constraint: 2377 >= 2346 (d)

Objective Value: 1183

A5->Min	Ob_1	Ob_2	Ob_3	Ob_4	Ob_5	Ob_6	Ob_7	Ob_8	Ob_9	Ob_10
p_i:	316	425	442	522	22	735	765	662	998	988
w_i:	763	881	733	664	27	767	424	324	83	25
e_i:	0,41	0,48	0,6	0,79	0,81	0,96	1,8	2,04	12,02	39,52
First Order:	7	8	1	5	6	2	4	10	9	3
y_i:	1	1	1	0	0	0	0	0	0	0

Constraint: 2377 >= 2346 (d)

Objective Value: 1183

8. SONUÇ

Bu tezde Kesir Doğrusal Boole Programlama Problemleri ele alınmış ve bu problemlerin çözümü için kesirler üzerinde "pay-pay, payda-payda" prensibi ile yapılan işlemler (koordinatsal işlemler) cebirsel ve geometrik olarak incelenerek temel bazı eşitsizlikler ispatlanmıştır. Tanımlanan işlemlerin İnversiyon özellikleri incelenerek, bu özellikler ile ilgili teorem ve sonuçlar verilmiştir. Bu cebirsel işlemler temel alınarak farklı stratejilere göre toplamlar geliştirilerek bazı önermeler ispatlanmıştır

Sıfır - bir maksimizasyon ve sıfır - bir minimizasyon sırt çantası problemleri ele alınarak, literatürdeki bu problemlerin çözümü için oluşturulmuş algoritmalar incelenmiştir. Çözümü iyileştirmek için değişim koşulları aranmıştır. Var olan algoritmalar geliştirilen stratejilerle birleştirilerek 0/1 maksimizasyon sırt çantası problemi için K_1, K_2, K_3 ve K_4 algoritmaları, 0/1 minimizasyon sırt çantası problemi için ise A_4 ve A_5 algoritmaları oluşturulmuştur. Önerilen algoritmaların garanti değerleri ele alınmıştır.

Son olarak ise geliştirilen bütün algoritmaların verimliliğini kıyaslayabilmek için, farklı özelliklere sahip 120 adet test problemi oluşturulmuştur. Bu örnek problemlerin başka araştırmacılar tarafından kolayca ulaşılabilmesi için KNAPSACK Test Problemleri Kütüphanesi hazırlanmıştır (http://fen.ege.edu.tr/~urfat/knapsack/). Bu kütüphanede, problemlerin parametreleri, onların optimal çözümleri ve önerilmiş algoritmaların bu problemler için bulduğu çözümler detaylı şekilde verilmiştir.

KAYNAKLAR DİZİNİ

Aarts, E. and Lenstra, J. K., 1997, Local Search in Combinatorial Optimization, John Wiley&Sons, England, 512p.

Arora, S. R., Puri, M. C. and Swarup, K., 1977, The Set Coveing Problem with Linear Fractional Functional, Indian Journal of Pure and Applied Mathematics, 8(5): 578-588 pp.

Ausiello, G., Cerscenzi, P., Kann, V., Marchetti-Spaccamela, A. and Protasi M., 1999, Complexity and Approximation: Combinatorial Optimization Problems and their Approximability Properties, Springer, Berlin, 524p.

Babayev, Dj. A., Mamedov, K. Sh., Mextiev, M. G., 1978, Methods for the suboptimal solution of the multidimensional knapsack problem, Computational Mathematics and Mathematical Physics, 18(6): 1443-1453 pp.

Bajalinov, Erik B., 2001, Linear-Fractional Programming: Theory, Methods, Kluwer Academic, Hungary, 423p.

Bakır, M. A. ve Altunkaynak, B. T., 2003, Tamsayılı Programlama, Nobel Yayınları, Ankara, 620s.

Balas E. and Zemel E., 1980, An Algortihm for Large Zero-one Knapsack Problems, Oper. Res., 28(5): 1130-1154 pp.

Başkaya, Z., 2005, Tamsayılı Programlama Algoritmaları ve Bilgisayar Uygulamalı Problem Çözümleri, Ekin Kitabevi, Ankara, 236s.

Chandra, S. and Chandramohan, M., A, 1980, Note on Integer Linear Fractional Program, Naval Research Logistics Quaterly, (27): 171-174 pp.

Cormen, T., Leiserson, C. E., Rivest, R. L. and Stein, C., 2009, Inroduction to Algorithms, MIT Press, 3rd Edition, Cambridge, 1292p.

Chernov Y. P. and Lange E. G., 1978, "Problems of Nonlinear Programming with Fractional Economic Criteria Methods and Applications", Kirgiz Academy of Science. Ilim, Frunze (in Russian).

Cura, T., 2008, Modern Sezgisel Teknikler ve Uygulamaları, Papatya Yayıncılık, İstanbul, 173s.

KAYNAKLAR DİZİNİ (devam)

Çetin, E., Mardikyan S. ve Lorcu, F., 2007, Sırt Çantalı Gezgin Satıcı Problemi, YA/EM 27. Ulusal Kongresi: 1112-1115 pp.

Fisher, M. L., 1980, Worst-case Analysis of Heuristic Algorithms, Management Science, 26(1): 1-17 pp.

Garey M. R. and Johnson D. S., 1979, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, San Francisco, 338p.

Gens, G. and Levner, E., 1980, Complexity of approximation algorithms for combinatorial problems: a survey, ACM SIGACT News, 12(3): 52-65 pp.

Gens, G. V. and Levner, E. V., 1980**,** Efficient Approximate Algortihms for Combinatorial Problems, The USSR Academy of Sciences, CEMI Press, Moscow, 66p.

Guler, A., Nuriyev, U., Berberler, M. E. ve Nuriyeva, F., 2012**,** Algorithms with Guarantee Value for Knapsack Problems**,** Optimization, 61(4) pp. 477-488.

Gupta, R. and Malhotra, R., 1995, Multi-Criteria Integer Linear Fractioal Programming Problem, Optimization, (35), 373-389 pp.

Güntzer, M. M. and Jungnickel, D., 2000, Approximate minimization algorithms for the 0/1 knapsack and subset-sum problem, Operations Research Letters, 26: 55-66 pp.

Gürsoy N. K., Firat A. and Nuriyev U., 2011, On the Algebra of Fractions, Ege Uni. Journal of Faculty of Sci., 35(2), pp 73-84.

Hochbaum, D. S., 1997, Approximation Algorithms for NP-Hard Problems, PWS Publishing, Boston, 596p.

Horowitz, E. and Sahni, S., 1978, Fundamentals of Computer Algorithms, Computer Science Press, New York, 626p.

Hungerford, T. W., 1974, Algebra, Holf, Rinehart and Wiston, Inc., New York, Chicago, 502p.

KAYNAKLAR DİZİNİ (devam)

Ibarra, O. H and Kim, C. E., 1975, Fast approximation algorithms for the knapsack and sum of subset problems, Journal of ACM, 22(4):463-468 pp.

Karlin, S., 1992, Mathematical Methods and Theory in Games, Programming and Economics, Dover Publications.

Karaboğa, D., 2004, Yapay Zeka Optimizasyon Algoritmaları, Atlas Yayın Dağıtım, İstanbul, 199s.

Kellerer, H., Pferschy, U., Pisinger, D., 2004, Knapsack Problems, Springer, 546 p.

Kulinkovich A. E., Nikitin A. I. and Nuriev U. G., 1982, "Efficient Algorithm For Optimizing The Allocation Of Computing Power Between Departments", Cybernetics 17: 772-778 pp (translation from Kibernetika, 1981).

Martello, S. and Toth, P., 1990, Knapsack Problems, John Wiley&Sons, England, 296p.

Nabiyev, V.V., 2011, Algoritmalar: Teoriden Uygulamalara, 3. Baskı, Seçkin Yayıncılık, Ankara, 830s.

Nabiyev, V.V., 2012, Yapay Zeka: İnsan-Bilgisayar Etkileşimi, 4. Baskı, Seçkin Yayıncılık, Ankara, 776s.

Nemhauser, G. L. and Woolsey, L. A., 1998, Integer and Combinatorial Optimization, Wiley, New York, 763p.

Nikitin, A.I. and Nuriev, U.G., 1982, A heuristic algorithm for solving a linear-fractional Boolean programming problem (Russian, English summary). Izvestiya Akad. Nauk Az. SSR, Ser. Fiz.-Tekn.-Math. Nauk, 3(5): 112 - 117 pp.

Nikitin A.I., Nuriev U.G., 1983, On a method of the solution of the Knapsack Problem, Kibernetika, (2): 108-110 pp.

Nuriyev, U. G., 1986, On the solution of the knapsack problem with guaranteed estimate, Problems of Computing Mathematics and Theoretical Cybernetics, 66-70.

KAYNAKLAR DİZİNİ (devam)

Papadimitriou, C. H. and Steiglitz, K., 1982, Combinatorial Optimization: Algorithms and Complexity, Prentice Hall, New Jersey, 496p.

Sergeyev, Y. D., 2003, Arithmetic of infinity, Edizioni Orizzonti Meridionali, 112p.

Sahni S., 1975, Approximate algorithms for the 0/1 knapsack problem, Journal of ACM, 22(1): 115-124 pp.

Vazirani, V. V., 2001, Approximation Algorithms, Springer, Berlin, 380p.

Veliev, G. P. and Mamedov, K. Sh., 1981, A method of solving the knapsack problem, Computational Mathematics and Mathematical Physics, 21(3): 605-611 pp.

Whitesitt, J. E., 1962**,** Boolean Algebra and its Applications, Addison-weslwy Pub. Company, Inc., London, England, 177p.

Woolsey, L. A., 1998, Integer programming, Wiley, New York, 264p.

ÖZGEÇMİŞ

Necla KIRCALI GÜRSOY, 26.06.1982 tarihinde Nazilli'de doğdu. İlkokulu Atça Atatürk İlkokulu'nda tamamladı. Ortaokul ve lise öğrenimini Atça Lisesinde birincilikle tamamladıktan sonra 1999 yılında Ege Üniversitesi Fen Fakültesi Matematik Bölümü'nü kazandı. 2004 yılında Matematik Bölümü Teorik Opsiyonu'ndan mezun olarak aynı yıl yüksek lisans öğrenimine başladı. Aydın Adnan Menderes Üniversitesi Fen Bilimleri Enstitüsü Matematik Anabilim Dalı Cebir ve Sayılar Teorisi Bilim Dalındaki Yüksek Lisans eğitimini "Türevli Asal Halkaların Değişmeliliği" isimli tez çalışmasıyla 2007 yılında başarılı bir şekilde tamamladı. 2008 yılında Ege Üniversitesi Fen Bilimleri Enstitüsü Fen ve Matematik Alanlar Eğitimi'nde Tezsiz Yüksek Lisansını tamamladı. 2010 yılında Ege Üniversitesi Fen Bilimleri Enstitüsü Matematik Anabilim Dalı Cebir ve Sayılar Teorisi Bilim Dalı'nda Doktora eğitimine başladı ve 2012 yılında doktora yeterliliğini tamamlayarak "Kesirler Cebiri ve Uygulamaları Üzerine" isimli tez çalışmasına başladı. 2005 – 2012 tarihleri arasında Ege Üniversitesi Ege Meslek Yüksek Okulun'da 31. Madde ile Öğretim Görevlisi olarak çalıştı. 2013 yılından beri Ege Üniversitesi Aliağa Meslek Yüksekokulun'da Öğretim Görevlisi olarak görev yapmaktadır.

EKLER

Ek 1 120 Test probleminin optimum değerleri ve parametreleri

Ek 2 K1, K2, K3 ve K4 algoritmalarının sonuçları ve oranları

Ek-3 A2, A3, A4 ve A5 algoritmalarının sonuçları ve oranları

Ek 1 120 Test probleminin optimum değerleri ve parametreleri

Problem Adı	Optimum Değer	Problem Parametreleri			
		ΣP	ΣW	c	d
d3b3p1	26693	42788	40255	12076	28179
d3b3p2	26005	40790	42559	12767	29792
d3b3p3	28087	42580	36518	10955	25563
d3b3p4	25606	39018	39965	11989	27976
d3b3p5	28279	44063	41439	12431	29008
d3b3p6	25339	37496	41435	12430	29005
d3b3p7	27858	44301	42752	12825	29927
d3b3p8	25029	38155	37996	11398	26598
d3b3p9	24992	38505	40676	12202	28474
d3b3p10	27414	39853	41250	12374	28876
d3b4p1	34166	43923	42710	17084	25626
d3b4p2	30343	37263	40813	16325	24488
d3b4p3	34222	44915	41950	16780	25170
d3b4p4	34568	40783	45069	18027	27042
d3b4p5	30693	40688	37392	14956	22436
d3b4p6	30269	37234	40820	16328	24492
d3b4p7	31970	40968	40556	16222	24334
d3b4p8	29232	38291	39098	15639	23459
d3b4p9	30709	39940	39099	15639	23460
d3b4p10	33008	42207	42791	17116	25675
d3b5p1	36264	43841	41665	20832	20833
d3b5p2	34760	41442	44664	22332	22332
d3b5p3	38006	44187	43399	21699	21700
d3b5p4	31145	36828	40081	20040	20041
d3b5p5	34809	41871	41671	20835	20836
d3b5p6	36947	45032	44050	22025	22025
d3b5p7	33208	39325	42283	21141	21142
d3b5p8	33465	38688	41916	20958	20958
d3b5p9	33717	40395	42396	21198	21198
d3b5p10	33471	39269	38723	19361	19362
d3b6p1	37297	40697	44296	26577	17719
d3b6p2	35177	40049	35664	21398	14266
d3b6p3	38857	42457	43497	26098	17399
d3b6p4	34256	37538	40966	24579	16387
d3b6p5	39834	44245	42929	25757	17172
d3b6p6	35590	37757	39804	23882	15922
d3b6p7	39868	42566	41830	25097	16733
d3b6p8	36275	39570	42133	25279	16854
d3b6p9	35799	39477	37580	22547	15033
d3b6p10	39048	42654	43408	26044	17364

d3b7p1	42128	43679	42503	29752	12751
d3b7p2	37838	39391	42568	29797	12771
d3b7p3	40311	42071	39379	27565	11814
d3b7p4	38273	41071	41241	28868	12373
d3b7p5	35985	38383	35751	25025	10726
d3b7p6	37600	39906	42734	29913	12821
d3b7p7	37458	39148	36567	25596	10971
d3b7p8	37413	38535	39530	27670	11860
d3b7p9	39474	41874	40395	28276	12119
d3b7p10	40551	42595	42860	30001	12859
d3b8p1	39170	39725	40070	32056	8014
d3b8p2	41819	42515	45665	36532	9133
d3b8p3	40645	41277	41961	33568	8393
d3b8p4	43038	43934	41856	33484	8372
d3b8p5	43176	44235	42499	33999	8500
d3b8p6	40314	41280	40626	32500	8126
d3b8p7	39010	39345	36548	29238	7310
d3b8p8	37920	38824	40279	32223	8056
d3b8p9	36985	37995	38753	31002	7751
d3b8p10	40383	41132	38763	31010	7753
d5b3p1	2880489	4247244	4425441	1327632	3097809
d5b3p2	2679945	4039103	4259358	1277807	2981551
d5b3p3	2658738	3958665	3892470	1167740	2724730
d5b3p4	3057056	4593441	3902077	1170623	2731454
d5b3p5	2846240	4193431	4596292	1378887	3217405
d5b3p6	2807483	4250283	4368534	1310560	3057974
d5b3p7	2485353	3644414	3820200	1146059	2674141
d5b3p8	2592549	3867931	3687591	1106277	2581314
d5b3p9	2910375	4236295	4400345	1320103	3080242
d5b3p10	2988143	4159314	4426451	1327935	3098516
d5b4p1	3194030	4232313	4033083	1613233	2419850
d5b4p2	3064005	3903307	3757352	1502940	2254412
d5b4p3	2870292	4136059	3755498	1502199	2253299
d5b4p4	3245426	4206443	4339454	1735781	2603673
d5b4p5	3032133	3892808	3819707	1527882	2291825
d5b4p6	2981278	3830005	4130607	1652242	2478365
d5b4p7	2877011	3755789	3621135	1448454	2172681
d5b4p8	3210884	4005580	3940846	1576338	2364508
d5b4p9	2889289	3750937	3783722	1513488	2270234
d5b4p10	3260577	4313150	4164784	1665913	2498871
d5b5p1	3511843	4034488	4035207	2017603	2017604
d5b5p2	3478192	4128525	4148024	2074012	2074012
d5b5p3	3640795	4133362	4372430	2186215	2186215
d5b5p4	3369365	3811165	4188036	2094018	2094018
d5b5p5	3548061	4262211	4677979	2338989	2338990

d5b5p6	3438798	3988672	3840446	1920223	1920223
d5b5p7	3281507	4276336	4057857	2028928	2028929
d5b5p8	3788714	4335521	4286349	2143174	2143175
d5b5p9	3416517	4070614	3686833	1843416	1843417
d5b5p10	3377832	3969085	3636161	1818080	1818081
d5b6p1	4329845	4903709	3768960	2261375	1507585
d5b6p2	3808796	4210459	4249006	2549403	1699603
d5b6p3	3985636	4277212	4148088	2488852	1659236
d5b6p4	3621918	3967173	3830180	2298107	1532073
d5b6p5	3828588	3964677	4338934	2603360	1735574
d5b6p6	3600117	3955008	3897013	2338207	1558806
d5b6p7	2388532	2615127	4186323	2511793	1674530
d5b6p8	3987016	4413536	4467600	2680559	1787041
d5b6p9	3765572	4245114	4288582	2573149	1715433
d5b6p10	3833482	4320355	4124302	2474581	1649721
d5b7p1	3573555	3722955	4141409	2898986	1242423
d5b7p2	3782659	4034204	3911262	2737883	1173379
d5b7p3	3637052	3875051	3825648	2677953	1147695
d5b7p4	3875122	3990684	3849623	2694736	1154887
d5b7p5	4247815	4469416	4413101	3089170	1323931
d5b7p6	3762073	3949388	4029198	2820438	1208760
d5b7p7	4052086	4198352	4345611	3041927	1303684
d5b7p8	3905197	4147909	4107786	2875450	1232336
d5b7p9	3522878	3674593	3790255	2653178	1137077
d5b7p10	4321285	4529218	4431770	3102238	1329532
d5b8p1	3807344	3890242	4369382	3495505	873877
d5b8p2	4146376	4241974	4326239	3460991	865248
d5b8p3	3843464	3901881	3966602	3173281	793321
d5b8p4	4121097	4152874	4118088	3294470	823618
d5b8p5	3818550	3888156	3954461	3163568	790893
d5b8p6	4132929	4207735	4142222	3313777	828445
d5b8p7	4154280	4225474	3944097	3155277	788820
d5b8p8	3995459	4057875	4103155	3282524	820631
d5b8p9	3763303	3775844	4091499	3273199	818300
d5b8p10	4163340	4214735	4060240	3248192	812048

Ek 2 K1, K2, K3 ve K4 algoritmalarının sonuçları ve oranları

Problem Adı	Optimum Değer	K1		K2		K3		K4	
		Amaç	%	Amaç	%	Amaç	%	Amaç	%
d3b3p1	26693	26693	1,0000	26693	1,0000	26693	1,0000	26693	1,0000
d3b3p2	26005	25984	0,9992	25984	0,9992	25984	0,9992	25504	0,9807
d3b3p3	28087	27989	0,9965	27989	0,9965	27989	0,9965	27836	0,9911
d3b3p4	25606	25606	1,0000	25606	1,0000	25606	1,0000	25567	0,9985
d3b3p5	28279	28250	0,9990	28250	0,9990	28250	0,9990	27922	0,9874
d3b3p6	25339	25311	0,9989	25311	0,9989	25312	0,9989	24657	0,9731
d3b3p7	27858	27824	0,9988	27824	0,9988	27858	1,0000	27279	0,9792
d3b3p8	25029	25029	1,0000	25029	1,0000	25029	1,0000	25029	1,0000
d3b3p9	24992	24988	0,9998	24988	0,9998	24988	0,9998	24988	0,9998
d3b3p10	27414	27366	0,9982	27366	0,9982	27381	0,9988	27155	0,9906
d3b4p1	34166	34156	0,9997	34156	0,9997	34156	0,9997	34096	0,9980
d3b4p2	30343	30315	0,9991	30315	0,9991	30315	0,9991	30315	0,9991
d3b4p3	34222	34203	0,9994	34203	0,9994	34214	0,9998	34129	0,9973
d3b4p4	34568	34504	0,9981	34504	0,9981	34504	0,9981	34162	0,9883
d3b4p5	30693	30613	0,9974	30613	0,9974	30613	0,9974	30401	0,9905
d3b4p6	30269	30223	0,9985	30223	0,9985	30255	0,9995	30010	0,9914
d3b4p7	31970	31970	1,0000	31970	1,0000	31948	0,9993	31503	0,9854
d3b4p8	29232	29220	0,9996	29220	0,9996	29220	0,9996	29220	0,9996
d3b4p9	30709	30706	0,9999	30706	0,9999	30709	1,0000	30687	0,9993
d3b4p10	33008	32987	0,9994	33007	1,0000	32987	0,9994	32787	0,9933
d3b5p1	36264	36166	0,9973	36264	1,0000	36172	0,9975	36264	1,0000
d3b5p2	34760	34755	0,9999	34750	0,9997	34755	0,9999	34755	0,9999
d3b5p3	38006	37972	0,9991	37972	0,9991	37969	0,9990	37972	0,9991
d3b5p4	31145	31103	0,9987	31103	0,9987	31127	0,9994	30863	0,9909
d3b5p5	34809	34708	0,9971	34708	0,9971	34708	0,9971	34261	0,9843
d3b5p6	36947	36917	0,9992	36943	0,9999	36907	0,9989	36748	0,9946
d3b5p7	33208	33184	0,9993	33184	0,9993	33184	0,9993	33107	0,9970
d3b5p8	33465	33465	1,0000	33465	1,0000	33465	1,0000	33465	1,0000
d3b5p9	33717	33627	0,9973	33627	0,9973	33627	0,9973	33623	0,9972
d3b5p10	33471	33416	0,9984	33416	0,9984	33471	1,0000	33250	0,9934
d3b6p1	37297	37297	1,0000	37297	1,0000	37297	1,0000	37297	1,0000
d3b6p2	35177	35160	0,9995	35160	0,9995	35160	0,9995	35012	0,9953
d3b6p3	38857	38815	0,9989	38815	0,9989	38815	0,9989	38815	0,9989
d3b6p4	34256	34256	1,0000	34256	1,0000	34256	1,0000	34231	0,9993
d3b6p5	39834	39834	1,0000	39834	1,0000	39834	1,0000	39696	0,9965
d3b6p6	35590	35550	0,9989	35550	0,9989	35550	0,9989	35503	0,9976
d3b6p7	39868	39855	0,9997	39846	0,9994	39855	0,9997	39855	0,9997
d3b6p8	36275	36275	1,0000	36275	1,0000	36275	1,0000	36146	0,9964
d3b6p9	35799	35780	0,9995	35780	0,9995	35780	0,9995	35741	0,9984
d3b6p10	39048	39036	0,9997	39036	0,9997	39036	0,9997	39036	0,9997
d3b7p1	42128	42102	0,9994	42102	0,9994	42102	0,9994	42072	0,9987

d3b7p2	37838	37838	1,0000	37838	1,0000	37838	1,0000	37838	1,0000
d3b7p3	40311	40279	0,9992	40279	0,9992	40279	0,9992	40273	0,9991
d3b7p4	38273	38126	0,9962	38255	0,9995	38178	0,9975	38055	0,9943
d3b7p5	35985	35968	0,9995	35968	0,9995	35968	0,9995	35968	0,9995
d3b7p6	37600	37569	0,9992	37569	0,9992	37595	0,9999	37485	0,9969
d3b7p7	37458	37458	1,0000	37458	1,0000	37458	1,0000	37277	0,9952
d3b7p8	37413	37407	0,9998	37407	0,9998	37407	0,9998	37351	0,9983
d3b7p9	39474	39434	0,9990	39434	0,9990	39434	0,9990	39395	0,9980
d3b7p10	40551	40545	0,9999	40545	0,9999	40545	0,9999	40513	0,9991
d3b8p1	39170	39170	1,0000	39170	1,0000	39170	1,0000	39109	0,9984
d3b8p2	41819	41819	1,0000	41819	1,0000	41819	1,0000	41819	1,0000
d3b8p3	40645	40645	1,0000	40645	1,0000	40645	1,0000	40645	1,0000
d3b8p4	43038	42954	0,9980	42954	0,9980	42954	0,9980	42954	0,9980
d3b8p5	43176	43125	0,9988	43125	0,9988	43134	0,9990	43009	0,9961
d3b8p6	40314	40237	0,9981	40237	0,9981	40237	0,9981	40219	0,9976
d3b8p7	39010	39010	1,0000	39010	1,0000	39010	1,0000	38991	0,9995
d3b8p8	37920	37906	0,9996	37920	1,0000	37906	0,9996	37920	1,0000
d3b8p9	36985	36980	0,9999	36980	0,9999	36980	0,9999	36980	0,9999
d3b8p10	40383	40376	0,9998	40376	0,9998	40376	0,9998	40284	0,9975
d5b3p1	2880489	2876267	0,9985	2876267	0,9985	2876267	0,9985	2876267	0,9985
d5b3p2	2679945	2679945	1,0000	2679945	1,0000	2679945	1,0000	2679945	1,0000
d5b3p3	2658738	2648003	0,9960	2648003	0,9960	2646992	0,9956	2593362	0,9754
d5b3p4	3057056	3046342	0,9965	3057056	1,0000	3046342	0,9965	3057056	1,0000
d5b3p5	2846240	2836807	0,9967	2836807	0,9967	2836807	0,9967	2792023	0,9810
d5b3p6	2807483	2798675	0,9969	2798675	0,9969	2798675	0,9969	2798675	0,9969
d5b3p7	2485353	2480661	0,9981	2480661	0,9981	2480661	0,9981	2458432	0,9892
d5b3p8	2592549	2589965	0,9990	2589965	0,9990	2590351	0,9992	2582564	0,9961
d5b3p9	2910375	2901588	0,9970	2901588	0,9970	2901588	0,9970	2875297	0,9879
d5b3p10	2988143	2986935	0,9996	2986935	0,9996	2986935	0,9996	2984276	0,9987
d5b4p1	3194030	3193637	0,9999	3193637	0,9999	3193637	0,9999	3193637	0,9999
d5b4p2	3064005	3057617	0,9979	3057617	0,9979	3061900	0,9993	3041558	0,9927
d5b4p3	2870292	2864426	0,9980	2864426	0,9980	2869724	0,9998	2814637	0,9806
d5b4p4	3245426	3239634	0,9982	3239634	0,9982	3239389	0,9981	3211407	0,9895
d5b4p5	3032133	3032133	1,0000	3032133	1,0000	3032133	1,0000	3032133	1,0000
d5b4p6	2981278	2978725	0,9991	2978725	0,9991	2978725	0,9991	2920898	0,9797
d5b4p7	2877011	2869470	0,9974	2869470	0,9974	2869470	0,9974	2856831	0,9930
d5b4p8	3210884	3200934	0,9969	3200934	0,9969	3200934	0,9969	3192081	0,9941
d5b4p9	2889289	2887422	0,9994	2887422	0,9994	2887422	0,9994	2882108	0,9975
d5b4p10	3260577	3257563	0,9991	3257563	0,9991	3257563	0,9991	3254752	0,9982
d5b5p1	3511843	3505864	0,9983	3505864	0,9983	3505864	0,9983	3496451	0,9956
d5b5p2	3478192	3477542	0,9998	3477542	0,9998	3477542	0,9998	3477542	0,9998
d5b5p3	3640795	3635186	0,9985	3638848	0,9995	3635186	0,9985	3635186	0,9985
d5b5p4	3369365	3369365	1,0000	3369365	1,0000	3369365	1,0000	3369365	1,0000
d5b5p5	3548061	3548061	1,0000	3548061	1,0000	3548061	1,0000	3543009	0,9986
d5b5p6	3438798	3438798	1,0000	3438798	1,0000	3438798	1,0000	3422032	0,9951

d5b5p7	3281507	3272300	0,9972	3272300	0,9972	3274612	0,9979	3226546	0,9833
d5b5p8	3788714	3775328	0,9965	3775328	0,9965	3777373	0,9970	3752893	0,9905
d5b5p9	3416517	3410064	0,9981	3410064	0,9981	3410064	0,9981	3401342	0,9956
d5b5p10	3377832	3350628	0,9919	3377832	1,0000	3360682	0,9949	3377832	1,0000
d5b6p1	4329845	4329845	1,0000	4329845	1,0000	4329845	1,0000	4315383	0,9967
d5b6p2	3808796	3806416	0,9994	3807832	0,9997	3807832	0,9997	3782410	0,9931
d5b6p3	3985636	3983058	0,9994	3983058	0,9994	3984585	0,9997	3973654	0,9970
d5b6p4	3621918	3621918	1,0000	3621918	1,0000	3621918	1,0000	3614483	0,9979
d5b6p5	3828588	3828588	1,0000	3828588	1,0000	3828588	1,0000	3828588	1,0000
d5b6p6	3600117	3600117	1,0000	3600117	1,0000	3600117	1,0000	3592770	0,9980
d5b6p7	2388532	2388532	1,0000	2388532	1,0000	2388532	1,0000	2388080	0,9998
d5b6p8	3987016	3987016	1,0000	3987016	1,0000	3987016	1,0000	3987016	1,0000
d5b6p9	3765572	3765572	1,0000	3765572	1,0000	3765572	1,0000	3765572	1,0000
d5b6p10	3833482	3833482	1,0000	3833482	1,0000	3833482	1,0000	3828525	0,9987
d5b7p1	3573555	3571388	0,9994	3573555	1,0000	3571388	0,9994	3573555	1,0000
d5b7p2	3782659	3778103	0,9988	3782659	1,0000	3772143	0,9972	3782659	1,0000
d5b7p3	3637052	3636438	0,9998	3636438	0,9998	3635813	0,9997	3619799	0,9953
d5b7p4	3875122	3874044	0,9997	3874044	0,9997	3874044	0,9997	3874044	0,9997
d5b7p5	4247815	4245471	0,9994	4245471	0,9994	4245471	0,9994	4245471	0,9994
d5b7p6	3762073	3756363	0,9985	3756363	0,9985	3756363	0,9985	3749432	0,9966
d5b7p7	4052086	4044522	0,9981	4045647	0,9984	4052086	1,0000	4029782	0,9945
d5b7p8	3905197	3900289	0,9987	3900289	0,9987	3901753	0,9991	3900289	0,9987
d5b7p9	3522878	3517885	0,9986	3517885	0,9986	3517885	0,9986	3514487	0,9976
d5b7p10	4321285	4321285	1,0000	4321285	1,0000	4321285	1,0000	4321285	1,0000
d5b8p1	3807344	3807344	1,0000	3807344	1,0000	3807344	1,0000	3796384	0,9971
d5b8p2	4146376	4143371	0,9993	4143371	0,9993	4143371	0,9993	4142693	0,9991
d5b8p3	3843464	3843464	1,0000	3843464	1,0000	3843464	1,0000	3837049	0,9983
d5b8p4	4121097	4119652	0,9996	4119652	0,9996	4119845	0,9997	4119652	0,9996
d5b8p5	3818550	3818550	1,0000	3818550	1,0000	3818550	1,0000	3818550	1,0000
d5b8p6	4132929	4132929	1,0000	4132929	1,0000	4132929	1,0000	4132929	1,0000
d5b8p7	4154280	4154280	1,0000	4154280	1,0000	4154280	1,0000	4154280	1,0000
d5b8p8	3995459	3995191	0,9999	3995191	0,9999	3995191	0,9999	3989106	0,9984
d5b8p9	3763303	3763303	1,0000	3763303	1,0000	3763303	1,0000	3763303	1,0000
d5b8p10	4163340	4163340	1,0000	4163340	1,0000	4163340	1,0000	4163340	1,0000
Ortalama Oran			0,9990		0,9992		0,9992		0,9959

Ek-3 A2, A3, A4 ve A5 algoritmalarının sonuçları ve oranları

Problem Adı	Optimum Değer	A2		A3		A4		A5	
		Amaç	%	Amaç	%	Amaç	%	Amaç	%
d3b3p1	16095	16095	1,000	16095	1,000	16197	1,006	16095	1,000
d3b3p2	14785	14816	1,002	14785	1,000	14816	1,002	14785	1,000
d3b3p3	14493	14556	1,004	14547	1,004	14556	1,004	14547	1,004
d3b3p4	13412	13451	1,003	13451	1,003	13472	1,004	13451	1,003
d3b3p5	15784	15864	1,005	15858	1,005	15864	1,005	15864	1,005
d3b3p6	12157	12157	1,000	12157	1,000	12157	1,000	12157	1,000
d3b3p7	16443	16447	1,000	16447	1,000	16522	1,005	16447	1,000
d3b3p8	13126	13126	1,000	13126	1,000	13126	1,000	13126	1,000
d3b3p9	13513	13517	1,000	13513	1,000	13700	1,014	13517	1,000
d3b3p10	12439	12439	1,000	12439	1,000	12763	1,026	12439	1,000
d3b4p1	9757	9762	1,001	9762	1,001	9762	1,001	9762	1,001
d3b4p2	6920	6963	1,006	6963	1,006	6993	1,011	6963	1,006
d3b4p3	10693	10739	1,004	10739	1,004	10739	1,004	10739	1,004
d3b4p4	6215	6236	1,003	6236	1,003	6236	1,003	6236	1,003
d3b4p5	9995	10008	1,001	9995	1,000	10008	1,001	9995	1,000
d3b4p6	6965	7015	1,007	7015	1,007	7015	1,007	7015	1,007
d3b4p7	8998	9039	1,005	9039	1,005	9039	1,005	9039	1,005
d3b4p8	9059	9060	1,000	9060	1,000	9060	1,000	9060	1,000
d3b4p9	9231	9256	1,003	9256	1,003	9256	1,003	9256	1,003
d3b4p10	9199	9293	1,010	9219	1,002	9293	1,010	9219	1,002
d3b5p1	7577	7577	1,000	7577	1,000	7737	1,021	7577	1,000
d3b5p2	6682	6724	1,006	6724	1,006	6724	1,006	6724	1,006
d3b5p3	6181	6191	1,002	6191	1,002	6191	1,002	6191	1,002
d3b5p4	5683	5683	1,000	5683	1,000	5831	1,026	5683	1,000
d3b5p5	7062	7144	1,012	7144	1,012	7144	1,012	7144	1,012
d3b5p6	8085	8116	1,004	8085	1,000	8227	1,018	8085	1,000
d3b5p7	6117	6120	1,000	6120	1,000	6117	1,000	6120	1,000
d3b5p8	5223	5223	1,000	5223	1,000	5223	1,000	5223	1,000
d3b5p9	6678	6710	1,005	6710	1,005	6709	1,005	6710	1,005
d3b5p10	5798	5943	1,025	5886	1,015	5839	1,007	5890	1,016
d3b6p1	3400	3400	1,000	3400	1,000	3456	1,016	3400	1,000
d3b6p2	4872	4941	1,014	4877	1,001	5016	1,030	4893	1,004
d3b6p3	3600	3607	1,002	3607	1,002	3607	1,002	3607	1,002
d3b6p4	3282	3308	1,008	3308	1,008	3308	1,008	3308	1,008
d3b6p5	4411	4473	1,014	4411	1,000	4496	1,019	4430	1,004
d3b6p6	2167	2186	1,009	2186	1,009	2186	1,009	2186	1,009
d3b6p7	2698	2760	1,023	2746	1,018	2784	1,032	2746	1,018
d3b6p8	3295	3314	1,006	3295	1,000	3424	1,039	3314	1,006
d3b6p9	3678	3704	1,007	3704	1,007	3813	1,037	3704	1,007
d3b6p10	3606	3618	1,003	3618	1,003	3725	1,033	3618	1,003
d3b7p1	1551	1607	1,036	1584	1,021	1615	1,041	1586	1,023

d3b7p2	1553	1561	1,005	1561	1,005	1561	1,005	1561	1,005
d3b7p3	1760	1806	1,026	1806	1,026	1806	1,026	1806	1,026
d3b7p4	2798	2806	1,003	2806	1,003	2798	1,000	2806	1,003
d3b7p5	2398	2430	1,013	2424	1,011	2419	1,009	2430	1,013
d3b7p6	2306	2306	1,000	2306	1,000	2306	1,000	2306	1,000
d3b7p7	1690	1783	1,055	1690	1,000	1783	1,055	1747	1,034
d3b7p8	1122	1131	1,008	1131	1,008	1131	1,008	1131	1,008
d3b7p9	2400	2457	1,024	2400	1,000	2496	1,040	2457	1,024
d3b7p10	2044	2076	1,016	2076	1,016	2085	1,020	2076	1,016
d3b8p1	555	573	1,032	573	1,032	568	1,023	573	1,032
d3b8p2	696	696	1,000	696	1,000	696	1,000	696	1,000
d3b8p3	632	632	1,000	632	1,000	632	1,000	632	1,000
d3b8p4	896	896	1,000	896	1,000	896	1,000	896	1,000
d3b8p5	1059	1059	1,000	1059	1,000	1059	1,000	1059	1,000
d3b8p6	966	966	1,000	966	1,000	1029	1,065	966	1,000
d3b8p7	335	358	1,069	335	1,000	366	1,093	345	1,030
d3b8p8	904	904	1,000	904	1,000	953	1,054	904	1,000
d3b8p9	1010	1010	1,000	1010	1,000	1015	1,005	1010	1,000
d3b8p10	749	765	1,021	756	1,009	765	1,021	765	1,021
d5b3p1	1366755	1369987	1,002	1366755	1,000	1387176	1,015	1368982	1,002
d5b3p2	1359158	1359354	1,000	1359354	1,000	1360918	1,001	1359354	1,000
d5b3p3	1299927	1299927	1,000	1299927	1,000	1299927	1,000	1299927	1,000
d5b3p4	1536385	1539495	1,002	1536385	1,000	1539495	1,002	1539495	1,002
d5b3p5	1347191	1349089	1,001	1349089	1,001	1357491	1,008	1349089	1,001
d5b3p6	1442800	1449545	1,005	1449545	1,005	1464413	1,015	1449545	1,005
d5b3p7	1159061	1169163	1,009	1167850	1,008	1169254	1,009	1167850	1,008
d5b3p8	1275382	1278167	1,002	1278167	1,002	1301538	1,021	1278167	1,002
d5b3p9	1325920	1325959	1,000	1325959	1,000	1351360	1,019	1325959	1,000
d5b3p10	1171171	1171171	1,000	1171171	1,000	1196748	1,022	1171171	1,000
d5b4p1	1038283	1038800	1,000	1038800	1,000	1048489	1,010	1038800	1,000
d5b4p2	839302	842229	1,003	842229	1,003	865520	1,031	842229	1,003
d5b4p3	1265767	1266471	1,001	1266471	1,001	1266471	1,001	1266471	1,001
d5b4p4	961017	972990	1,012	969693	1,009	975908	1,015	972903	1,012
d5b4p5	860675	860675	1,000	860675	1,000	868031	1,009	860675	1,000
d5b4p6	848727	849054	1,000	849054	1,000	849054	1,000	849054	1,000
d5b4p7	878778	878778	1,000	878778	1,000	889520	1,012	878778	1,000
d5b4p8	794696	798933	1,005	798933	1,005	798933	1,005	798933	1,005
d5b4p9	861648	862485	1,001	862485	1,001	862485	1,001	862485	1,001
d5b4p10	1052573	1052573	1,000	1052573	1,000	1058689	1,006	1052573	1,000
d5b5p1	522645	524478	1,004	522952	1,001	524478	1,004	522952	1,001
d5b5p2	650333	650333	1,000	650333	1,000	662475	1,019	650333	1,000
d5b5p3	492567	498176	1,011	496183	1,007	498176	1,011	498176	1,011
d5b5p4	441800	441800	1,000	441800	1,000	456051	1,032	441800	1,000
d5b5p5	714150	717910	1,005	716807	1,004	717910	1,005	716807	1,004
d5b5p6	549874	559773	1,018	550107	1,000	559773	1,018	557274	1,013

d5b5p7	994829	994829	1,000	994829	1,000	994829	1,000	994829	1,000
d5b5p8	546807	552177	1,010	550483	1,007	552177	1,010	552177	1,010
d5b5p9	654097	657471	1,005	657471	1,005	666235	1,019	657471	1,005
d5b5p10	591253	591253	1,000	591253	1,000	591253	1,000	591253	1,000
d5b6p1	573864	588972	1,026	573864	1,000	590967	1,030	588972	1,026
d5b6p2	401663	402620	1,002	402620	1,002	401852	1,000	402620	1,002
d5b6p3	291576	292054	1,002	292054	1,002	306447	1,051	292054	1,002
d5b6p4	345255	348590	1,010	345255	1,000	362131	1,049	348590	1,010
d5b6p5	136089	136089	1,000	136089	1,000	144540	1,062	136089	1,000
d5b6p6	354891	357404	1,007	354891	1,000	370102	1,043	357404	1,007
d5b6p7	226595	227047	1,002	227047	1,002	227047	1,002	227047	1,002
d5b6p8	426520	426520	1,000	426520	1,000	437909	1,027	426520	1,000
d5b6p9	479542	479542	1,000	479542	1,000	479542	1,000	479542	1,000
d5b6p10	486873	490752	1,008	490752	1,008	506135	1,040	490752	1,008
d5b7p1	149400	149400	1,000	149400	1,000	149400	1,000	149400	1,000
d5b7p2	251545	251545	1,000	251545	1,000	251545	1,000	251545	1,000
d5b7p3	237999	238722	1,003	238722	1,003	238722	1,003	238722	1,003
d5b7p4	115562	116640	1,009	116640	1,009	116640	1,009	116640	1,009
d5b7p5	221601	223945	1,011	223945	1,011	228935	1,033	223945	1,011
d5b7p6	187315	187845	1,003	187845	1,003	190385	1,016	187845	1,003
d5b7p7	146266	149181	1,020	149181	1,020	149181	1,020	149181	1,020
d5b7p8	242712	242712	1,000	242712	1,000	251623	1,037	242712	1,000
d5b7p9	151715	153241	1,010	153241	1,010	153241	1,010	153241	1,010
d5b7p10	207933	208151	1,001	208151	1,001	214553	1,032	208151	1,001
d5b8p1	82898	85659	1,033	85529	1,032	85659	1,033	85529	1,032
d5b8p2	95598	95598	1,000	95598	1,000	96883	1,013	95598	1,000
d5b8p3	58417	62050	1,062	58417	1,000	62050	1,062	62050	1,062
d5b8p4	31777	32971	1,038	32971	1,038	32971	1,038	32971	1,038
d5b8p5	69606	69606	1,000	69606	1,000	71057	1,021	69606	1,000
d5b8p6	74806	75340	1,007	75340	1,007	75340	1,007	75340	1,007
d5b8p7	71194	71194	1,000	71194	1,000	75991	1,067	71194	1,000
d5b8p8	62416	67032	1,074	67032	1,074	67032	1,074	67032	1,074
d5b8p9	12541	12674	1,011	12674	1,011	12674	1,011	12674	1,011
d5b8p10	51395	51395	1,000	51395	1,000	51395	1,000	51395	1,000
Ortalama Oran			1,008		1,005		1,017		1,007