MİNİMAL AĞIRLIKLI DOMİNANT ALT KÜME PROBLEMİNİN ÇÖZÜMÜ İÇİN BİR YAKLAŞIM

Urfat G. NURİYEV
Ege Üniversitesi Fen Fakültesi, Matematik Bölümü 35100 Bornova, İzmir
Burak ORDİN
Ege Üniversitesi Fen Fakültesi, Matematik Bölümü 35100 Bornova, İzmir

ÖZET

Bu çalışmada; bir işe birden fazla işlem noktasının karşı getirilebilmesi bakımından Atama Problemi'nin genelleştirilmesi olarak düşünülebilecek Minimal Ağırlıklı Dominant Alt Küme isimli bir kombinatoryal optimizasyon problemi incelenmiştir. Bu problem, Global Optimizasyon problemlerinin Kesen Açılar Yöntemi (Cutting Angle Method) ile çözümünde karşılaşılan bir Yardımcı probleme denk bir problem olarak ele alınıp, problemin matematiksel modeli tanımlanmış, ekonomik yorumu verilmiş ve özellikleri incelenmiştir. Daha sonra problemi çözmek için yeni bir algoritma önerilip, C++ dilinde programı tasarlanmış ve hesaplama denemelerinin sonuçları verilmiştir.

Anahtar Kelimeler. Minimal Ağırlıklı Dominant Altküme Problemi, Global Optimizasyon, Atama Problemi, Kesen Açılar Yöntemi.

1. GİRİŞ

Son yıllarda Global Optimizasyon problemlerinin büyük bir sınıfını çözmek için Kesen Açılar Yöntemi (KAY) (Cutting Angle Method, (CAM)) olarak adlandırılan yeni bir yöntem üzerinde çalışılmaktadır [1, 11]. Bu yöntem konveks minimizasyonundaki Kesen Yüzey Metodu'nun bir genelleştirilmesi olarak geliştirilmektedir ve iteratiftir. Yani her bir adımında gene bir global optimizasyon problemi olan bir yardımcı problem (altproblem) in çözülmesi gerekir. Bu altproblem $S = \{x \mid \sum_{i=1}^{n} x_i = 1, x_i \geq 0, i = 1,...,n\}, x = (x_1,...,x_n)$ kümesi üzerinde tanımlı bir dereceli pozitif artan homojen fonksiyonların minimizasyonunda ortaya çıkar.

Bir Global Optimizasyon probleminin KAY ile çözümünde altproblemin çözümü için kullanılan metodun iyi olması genel çözümü etkiler [2].

"Minimal Ağırlıklı Dominant Alt Küme Problemi" (MADAK) olarak adlandırdığımız problem boole değişkenli kombinatoryal bir optimizasyon problemi olup, yukarıda adı geçen Yardımcı Probleme denk bir problem gibi incelenmeye başlanmıştır [3-8].

[7] çalışmasında bu problemin NP-Zor sınıftan olduğu gösterilmiştir ve [3, 5, 6] çalışmalarında MADAK problemini çözmek için farklı yöntemler geliştirilmiştir.

Bu çalışmada, Global Optimizasyon problemlerinin Kesen Açılar Yöntemi ile çözümünde karşılaşılan yardımcı problemin MADAK problemine dönüşümü gösterilmiş, MADAK probleminin matematiksel modeli ve ekonomik yorumu verilmiştir. Daha sonrada problemin özelliklerinden yararlanarak çözüm için yeni bir algoritma önerilmiştir. Algoritma C++ dilinde tasarlanmış ve farklı boyutlu problemlerle hesaplama denemeleri yapılmıştır.

2. YARDIMCI PROBLEMİN FORMÜLASYONU

Varsayalım ki, (m*n) boyutlu (l_i^k) matrisi verilsin ve (l_i^k) matrisinin ilk n satırı diagonal matris olsun. (k = 1,…,m - satırları, i = 1,…n - sütunları göstersin.) Yani k ≤ n ve k ≠ i iken m ≥ n ve $l_i^k = 0$, diğer durumlarda $l_i^k > 0$ dır.

Burada, h(x) fonksiyonu aşağıdaki gibi tanımlanır :

$$h(x) = \max_k \min_{i \in I(l^k)} l_i^k x_i \,, \; I(l^k) = \{i: \; l_i^k > 0\}.$$

Buna göre problemin modeli aşağıdaki gibi verilir:

Altproblem

$$h = \min_{x} h(x) . \quad (1)$$

$$x \in S = \{x \Big| \sum_{i=1}^{n} x_i = 1, x_i \geq 0, i = 1,...,n\} \quad (2)$$

3. YARDIMCI PROBLEMİN BOOLEAN BİR PROBLEME DÖNÜŞÜMÜ

İzleyen dönüşüm altında

$p=m-n$, $u_i^j = \frac{1}{l_i^i} - \frac{1}{l_i^{j+n}}$, i=1,2,…n; j = 1,2,…,p.

Altproblemin (l_i^k) matrisi bir (u_i^j) matrisine dönüşür.

Aşağıdaki fonksiyonu tanımlayalım.

$$Sg(x) = \begin{cases} 1, & \text{eğer } x \geq 0 \\ 0, & \text{eğer } x < 0 \end{cases}$$

ve $x_i^j, i = 1,2,...,n; j = 1,2,...,p$ değişkenlerini düşünelim:

$$x_i^j = \begin{cases} 1, & \text{eğer } l_i^{j+n} \to l_i^j \text{ dönüşümü gerçeklendiğinde} \\ 0, & \text{aksi halde} \end{cases}.$$

O zaman (1)-(2) Altproblemi izleyen Boolean 0-1 programlama problemine dönüştürülür.

$$\sum_{i=1}^{n}\sum_{j=1}^{p} u_i^j x_i^j \to \min_{x_i^j} \quad (3)$$

$$\sum_{i=1}^{n} x_i^j \leq 1, \text{ j = 1, 2,…, p,} \quad (4)$$

$$\sum_{j=1}^{p} x_i^j \leq 1, \text{ i = 1, 2,…, n,} \quad (5)$$

$$\sum_{i=1}^{n}\sum_{j=1}^{p} x_i^j \geq 1, \quad (6)$$

$$\sum_{i=1}^{n} y_i^j \geq 1, \text{ j = 1, 2,…, p,} \quad (7)$$

$$x_i^j = 0 \vee 1, \text{ i = 1, 2,…, n; j = 1, 2,…, p.} \quad (8)$$

$$y_i^j = Sg\left(\max_{k=\overline{1,p}} \{u_i^k x_i^k\} - u_i^j\right), \text{ i = 1, 2,…, n ; j = 1, 2,…, p,} \quad (9)$$

[4] makalesinde izleyen teorem ispatlanmıştır.

Teorem 1 (1)-(2) Altproblemi ve (3)-(9) problemi denktir.

4. MİNİMAL AĞIRLIKLI DOMİNANT ALTKÜME PROBLEMİ

(3)-(9) problemini Minimal Ağırlıklı Dominant Altküme problemi olarak adlandıralım. Bu problemi aşağıdaki yorumlayabiliriz.

(u_i^j), $u_i^j \geq 0$ (j = 1, 2, ..., p; i = 1, 2, ..., n) olmak üzere, p*n boyutlu bir matristir.

Burada amaç matrisin elemanlarından öyle bir altküme seçilmelidir ki,

i) Seçilmiş elemanların toplamı enküçük olsun,

ii) Her bir satır için ya bir seçilmiş eleman vardır yada bu satırın öyle bir elemanı vardır ki, bu elemanın yerleştiği sütunda bulunan ve ondan büyük olan seçilmiş bir eleman vardır.

Bu problemin ekonomik yorumu aşağıdaki gibi verilebilir:

p sayıda (j = 1, 2, ..., p) işlemden oluşan bir iş n sayıda (i = 1, 2, ..., n) işlemcide icra görebilir. Varsayalım ki (u_i^j) matrisi bu işlemlerin verilen işlemcilerde icra görmesi için gerekli zamanı (veya maliyeti) göstersin,

Eğer i. sütun için aşağıdaki koşul sağlanıyorsa

$$u_i^{j_1} \leq u_i^{j_2} \leq \ldots \leq u_i^{j_p} \qquad (10)$$

$u_i^{j_1}$ - j_1 işleminin i. işlemcideki icra zamanını,

$u_i^{j_2}$ - j_1 ve j_2 işlemlerinin i. işlemcideki icra zamanını,

$u_i^{j_p}$ - bütün işlemlerin ($j_1, j_2, \ldots, j_p$) birlikte i. işlemcideki icra zamanını gösterir.

İşlemler işlemciler arasında öyle paylaştırılmalıdır ki, bütün iş en kısa zamanda / en az maliyetle icra olunsun. Açıktır ki, bu problem Atama Problemi'nin bir genellemesidir [10].
Atama problemi karmaşıklığı $O(r^3)$ ($r = \max\{p, n\}$ olmak üzere) olan Macar metoduyla çözülebilmesine rağmen ilerleyen bölümlerde gösterildiği gibi Madak problemi NP-Zor sınıftandır [7].

5. MADAK PROBLEMİ İÇİN BAZI TANIMLAR

Tanım 1. u_i^j elemanını ele alalım. Eğer öyle $j_1^{(u_i^j)}, j_2^{(u_i^j)}, \ldots, j_{t(u_i^j)}^{(u_i^j)}$ satırları varsa ki, $u_i^j \geq u_i^{j_1^{(u_i^j)}}, u_i^j \geq u_i^{j_2^{(u_i^j)}}, \ldots, u_i^j \geq u_i^{j_{t(u_i^j)}^{(u_i^j)}}$ olursa, o zaman u_i^j elemanını $u_i^{j_1^{(u_i^j)}}$, $u_i^{j_2^{(u_i^j)}}$, ..., $u_i^{j_{t(u_i^j)}^{(u_i^j)}}$ elemanlarının *dominantı* (veya *i.nci sütuna göre dominantı*), j.nci satırı ise *i.nci sütuna göre* $j_1^{(u_i^j)}, j_2^{(u_i^j)}, \ldots, j_{t(u_i^j)}^{(u_i^j)}$ *satırlarının dominantı* olarak adlandıralım. u_i^j elemanının dominantını ise $\overline{u}_i^j$ ile işaretleyelim.

Tanım 2. Dominantlık kavramı u_i^j *elemanının* $j, j_1^{(u_i^j)}, j_2^{(u_i^j)}, \ldots, j_{t(u_i^j)}^{(u_i^j)}$ *numaralı satırları örtmesi* anlamına gelir. (u_i^j) matrisinin tüm satırlarını örten dominant elemanların altkümesini *dominant altküme* olarak adlandıralım.

Tanım 3. Matrisin her bir elemanının değerini onun *ağırlığı* ve her bir altkümenin elemanlarının ağırlıkları toplamını *altkümenin ağırlığı* gibi tanımlarsak MADAK problemi, *ağırlığı enküçük olan dominant altkümenin bulunmasına* getirilir.

Tanım 4. (4)-(9) koşulunu (veya (ii) koşulunu) sağlayan herbir (x_i^j) matrisini *olurlu çözüm* olarak adlandıralım. Olurlu çözümü $\hat{X}$ ile ve ona uygun amaç fonksiyonunun değerini $\hat{U}$ ile işaretleyelim.

Tanım 5. Eğer olurlu çözüm (2) koşulunu (veya (i) koşulunu) da sağlarsa elde edilen çözüm *optimal çözüm* olacaktır. Optimal çözümü X^* ile, ona uygun amaç fonksiyonunun değerini de U^* ile işaretleyelim. Açıktır ki, her bir olurlu çözüme bir dominant altküme, optimal çözüme de minimal ağırlıklı dominant altküme uygun gelir.

Tanım 6. Açıktır ki, her bir eleman hem kendisinin, hem de kendi yerleştiği satırın dominantıdır. Ona göre de her bir u_i^j elemanının örttüğü satırların sayısı $(t(u_i^j)+1)$ olacak, burada $t\left(u_i^j\right)$, u_i^j elemanının i'nci sütuna göre dominantı olduğu satırların sayısıdır. $(t(u_i^j)+1)$ sayısına u_i^j elemanının *dominantlık derecesi*, $u_i^j /(t(u_i^j)+1)$ oranına ise u_i^j elemanının *dominant ağırlık oranı* adı verelim ve bu oranı $R\left(u_i^j\right)$ olarak gösterelim.

Tanım 7. Varsayalım ki, $\overline{u}_i^j$, u_i^j elemanının herhangi bir dominantıdır. Eğer $\overline{u}_i^j$ elemanının dominant ağırlık oranı u_i^j elemanının dominant ağırlık oranından büyük değilse, yani $R\left(\overline{u_i^j}\right) \le R\left(u_i^j\right)$ $\left(\dfrac{\overline{u}_i^j}{t(\overline{u}_i^j)+1} \le \dfrac{u_i^j}{t(u_i^j)+1}\right)$ olursa $\overline{u}_i^j$ *elemanını* u_i^j *elemanına göre uygun dominant* olarak adlandıralım.

Tanım 8. Her bir j satırı için aşağıdaki gibi u^j elemanını belirleyelim:
$u^j = \min\limits_{i=\overline{1,n}}\{u_i^j\}, \mathrm{j}=1,...,\mathrm{p}$ ve $\mathrm{i_j} = arg\min\limits_{i=\overline{1,n}}\{u_i^j\}, \mathrm{j}=1,...,\mathrm{p}$ işaretleyelim. Açıktırki, $u^j = u_{i_j}^j$ olacaktır. $u_{i_j}^j$ elemanını j.nci satır için *kritik eleman* olarak adlandıralım.

Tanım 9. Her bir i sütunu için aşağıdaki gibi $\widetilde{u}_i$ elemanını belirleyelim:

$$\widetilde{u}_i = \begin{cases} 0, \ eğer\ i.\ nci\ sütunda\ kritik\ eleman\ yoksa \\ \max\limits_{j=\overline{1,p}}\left\{u_i^j \,\middle|\, u_i^j = u^j\right\}, \ aksi\ halde \end{cases}$$

Varsayalım ki, $\exists$i, $\widetilde{u}_i \neq 0$, o zaman $\widetilde{u}_i = \max\limits_{j=\overline{1,p}}\{u_i^j \,|\, u_i^j = u^j\} = \max\limits_{j=\overline{1,p}}\{u_{i_j}^j \,|\, i_j = i\} = u_i^{j_i}$, burada $j_i = arg\max\limits_{j=\overline{1,p}}\{u_{i_j}^j \,|\, i_j = i\}$. $u_i^{j_i}$ elemanını i.nci sütun için *üstün kritik eleman* olarak adlandıralım.

Hiç bir kritik elemanın olmadığı sütunu *bağımsız sütun* olarak adlandıralım.

Tanım 10. Her bir u_i^j kritik elemanı için aşağıdaki gibi bir veri yapısı oluşturalım:
$(u_i^j; j, j_1^{(u_i^j)}, j_2^{(u_i^j)}, ..., j_{t(u_i^j)}^{(u_i^j)})$. Burada j, u_i^j elemanının yerleştiği satırın numarası, $j_1^{(u_i^j)}, j_2^{(u_i^j)}, ..., j_{t(u_i^j)}^{(u_i^j)}$ ise j.nci satırın i.nci sütuna göre dominantı olduğu satırların numaralarıdır. $(j, j_1^{(u_i^j)}, j_2^{(u_i^j)}, ..., j_{t(u_i^j)}^{(u_i^j)})$ vektörüne u_i^j *elemanının dominantlık vektörü* adını verelim ve $V\left(u_i^j\right)$ olarak işaretleyelim. Açıktır ki, bu vektörün koordinatlarının sayısı $(t(u_i^j)+1)$ olacaktır.

Tanım 11. Eğer herhangi bir olurlu çözümde u_i^j yerine $\overline{u}_i^j$ uygun dominantı kullanılarak alınmış çözüm için amaç fonksiyonunun değeri daha iyi ise, $\overline{u}_i^j$ yi *iyileştiren uygun dominant* olarak adlandıralım.

Ayrıca aşağıdaki işaretlemeleri kabul edelim:

$$U_i = \max_{j=\overline{1,p}}\{u_i^j\}, \ \overline{U} = \min_{i=\overline{1,n}}\{u_i\}, \ \widetilde{U} = \max_{i=\overline{1,n}}\{\widetilde{u}_i\}$$

6. MADAK PROBLEMİNİN ÖZELLİKLERİ

MADAK Probleminin aşağıdaki özellikleri bilinmektedir [9]:

Özellik 1. Problemin üstün kritik elemanlar kümesi kritik elemanlar kümesine göre daha iyi bir olurlu çözüm verir, yani $\sum_{i=1}^{n} u_i^{j_i} \le \sum_{j=1}^{p} u_{i_j}^{j}$.

Özellik 2. Üstün kritik elemanların sayısı (q), sütunların sayısından (n) fazla değildir.

Özellik 3. Her bir olurlu çözümde üstün kritik elemanların en büyüğünden küçük olmayan bir eleman vardır, yani eğer $\hat{U} = \sum_{i=1}^{n}\sum_{j=1}^{p} \hat{u}_i^j \hat{x}_i^j$ ise, $\exists \hat{u}_i^j$ var ki, $\hat{u}_i^j \ge \bar{U}$.

Özellik 4. Her bir olurlu çözümde amaç fonksiyonunun değeri üstün kritik elemanların en büyüğünden küçük değildir, yani $\hat{U} \ge \bar{U}$.

Özellik 5. (u_i^j) matrisinin her sütununun en büyük elemanı $U_i \left(i = \overline{1,n}\right)$ MADAK probleminin bir olurlu çözümünü verir, yani her bir U_i (ii) koşulunu veya ((4)–(9) koşullarını) sağlıyor.

Özellik 6. MADAK probleminin optimal çözümü için amaç fonksiyonun değeri (u_i^j) matrisinin sütunlarının en büyük elemanlarının en küçüğünden büyük değildir, yani $U^* \le \bar{U}$.

Özellik 7. Amaç fonksiyonunun optimal değeri aşağıdaki sınırlarda yerleşir: $\bar{U} \le U^* \le \bar{U}$.

Özellik 8. $\bar{U} = u_c^d$ olduğunu varsayalım. Eğer $u_c^d = \min_{i=1,n}\{u_i^d\}$ olursa, o zaman $x_c^d = 1$ ve diğer elemanlar 0 alınır ve bu MADAK probleminin optimal çözümüdür.

Özellik 9. $\bar{U} = u_m^s$ olduğunu varsayalım. Eğer $u_m^s = \max_{j=1,p}\{u_m^j\}$ olursa, o zaman $x_m^s = 1$ ve diğer elemanlar 0 alınır ve bu MADAK probleminin optimal çözümüdür.

Özellik 10. Eğer $\bar{U} = \bar{U}$ ise, o zaman $U^* = \bar{U} = \bar{U}$ eşitliği doğrudur.

7. MADAK PROBLEMİ İÇİN YENİ BİR ÇÖZÜM ALGORİTMASI

Bu bölümde MADAK Problemini çözmek için problemin özellikleri gözönüne alınarak çok aşamalı bir yöntem önerilmiştir:

Birinci aşamada (A1-A2 adimları) U matrisinin her sütunu için en büyük dominant eleman (U_i, (i = 1, 2,...,n); özellik 5 e göre bunların her biri bir olurlu çözümdür.) bulunarak onlardan en küçüğü $\left(\bar{U}\right)$ seçilir ve bu başlangıç çözüm olarak kabul edilir($\hat{U} = \bar{U}$). Eğer bu çözüm üstün kritik elemanlardan en büyüğüne eşit ise özellik 9 a göre optimal çözüm bulunmuştur ve algoritma işini bitirir (A6 adımı)

İkinci aşamada (A3-A9 adımları) her satır için kritik elemanlar ve her sütun için üstün kritik elemanlar belirlenerek ve üstün kritik elemanlar kendi aralarında, kalan kritik elemanlar kendi aralarında ve üstün kritik elemanlar başta olacak şekilde azalan sırada dizilir. (NOT: Bu sıralama ((10) sırası) bundan sonraki tüm kısımlarda greedy kriteri olarak kullanılır). Sonra üstün kritik elemanlardan en büyüğü seçilerek bu sıralamaya göre bir olurlu çözüm bulunur ve bu çözüm önceden bulunmuş çözümle kıyaslanarak bunlardan küçüğü daha iyi bir çözüm olarak saklanır.

Sonraki aşamalarda 2-7 özellikleri gözönüne alınarak bulunmuş çözüm iyileştirilmeye çalışılır (A10-A31 adımları). Bunun için en son belirlenmiş çözüme dahil olan elemanlar sırasıyla onlardan daha iyi uygun dominant elemanlarla değiştirilerek bulunmuş çözüm iyileştirilmeye çalışılır.

7.1 ALGORİTMA A

Algoritma A ana algoritmadır ve yukarıdaki aşamalar bu algoritma ile gerçekleştirilir. Algoritma B ise yardımcı algoritma olarak kullanılır.

Algoritma daki $i(v_k)$ ve $j(v_k)$, v_k elemanının uygun olarak yerleştiği sütun ve satır numaralarını, kg ise en son bulunmuş çözümdeki elemanlardan olan incelenecek elemanın (10) sırasındaki yerini gösterir.

A0: U=0, $X=(x_i^j)$=0, ($i=1,...,n$; $j=1,2,...,p$); $P=\{1,\ 2,...,p\}$.

A1: Her bir sütun için en büyük dominant eleman bulunur. Yani $u_i=\max\limits_{j=1,p}\{u_i^j\}$ ve $j_i=arg\ \max\limits_{j=1,n}\{u_i^j\}, i=1,2,...,n$ bulunur. Açıktır ki, $u_i=u_i^{j_i}$ olacaktır. Özellik 5 e göre bunların her biri bir olurlu çözümdür.

A2: Bu dominantlardan en küçüğü $(\overline{u})$ bulunur, yani o an için olurlu çözümlerden en iyisi belirlenir. Başka bir deyişle $\hat{U}=\min\limits_{i=1,n}\{u_i\}$, $i_{j_i}=arg\ \min\limits_{i=1,n}\{u_i\}$ bulunur. Açıktır ki, $\hat{U}=u_{i_{j_i}}^{j_i}$ olacaktır ve $ig=i_{j_i}$, $jg=j_i$, $kg=0$ olarak gösterelim

bulunur. Açıktır ki, $u^j=u_{i_j}^j$, $j=1,...,p$ olacaktır. Bu elemanlardan her birisi için dominantlık vektörü $V(u^j)$ oluşturulur.

A3: Her satır için kritik eleman bulunur. Yani $u^j=\min\limits_{i=1,n}\{u_i^j\}$ ve $i_j=arg\ \min\limits_{i=1,n}\{u_i^j\}, j=1,2,...,p$.

A4: Her sütun için üstün kritik eleman $(\tilde{u}_i, i=1,...,n)$ bulunur. (Eğer herhangi bir i sütununda kritik eleman yoksa yani bağımsız sütun ise, bu sütun için $\tilde{u}_i=0$ kabul edilir.

A5: Kritik elemanlar azalan sırada dizilip, üstün kritik elemanlar başa alınarak yeniden sıralama yapılır.

$$u_{i_1}\geq u_{i_2}\geq ...\geq u_{i_q},\ u_{i_{q+1}}\geq ...\geq u_{i_p} \qquad (9)$$

sırasında ilk q eleman üstün kritik elemandır. (Özellik 2 ye göre q≤n dir), $u_{i_{q+1}},...,u_{i_p}$ ise kalan satırların kritik elemanlarıdır. Bundan sonraki sunumu kolaylaştırmak için (9) sırasını aşağıdaki gibi yeniden işaretleyelim :

$$v_1\geq v_2\geq ...\geq v_q,\ v_{q+1}\geq ...\geq v_p \qquad (10)$$

Yani burada $v_1=u_{i_1}$, $v_2=u_{i_2}$,..., $v_p=u_{i_p}$

A6: Eğer $v_1=\hat{U}$ ise $\hat{X}$ optimal çözümdür (Ö9 özelliğine göre), yani $x_{ig}^{jg}=1$, $UG=\hat{U}$ ve SON a geçilir.

A7: PK=P ve $U=v_1$, $P=P-V(v_1)$ kabul edip, B algoritmasını kullanarak (10) sıralamasına göre bir olurlu çözüm bulunur.

A8: Bulunmuş yeni çözüm için amaç fonksiyonunun değeri $U\geq\hat{U}$ ise A10'a git.

A9: $\hat{U}=U$, $ig=i(v_1), jg=j(v_1), kg=ks$.

A10: UG=0, uy = $\hat{U}-v_p$, k = 1, pk = p.

A11: s=1.

A12: j = PK(s)

A13: i = 1

A14: Eğer $u_i^j>uy$ veya $u_i^j\leq v_k$ ise A21'e git.

A15: $U=u_i^j, P=PK-V(u_i^j)$ kabul edip,

A16: Eğer $P=\varnothing$ ise ks=0 ve A18' e git.

A17: B algoritmasını kullanarak (10) sırasına göre bir çözüm bulunur.

A18: Eğer $U \geq \hat{U}$ ise A21'e git.

A19: $\hat{U} = U$, ig=i, jg=j, kg=ks.

A20: Eğer k=1 ise $uy = \hat{U} - v_p$

A21: i = i + 1

A22: Eğer $i \leq n$ ise A14'e git.

A23: s = s +1

A24: Eğer $s \leq pk$ ise A12'ye git.

A25: $UG = UG + u_{ig}^{jg}$, $x_{ig}^{jg} = 1.$

A26:Eğer kg=0 ise A29'a git.

A27: $PK = PK - V(u_{ig}^{jg})$, $pk = |PK|$, $uy = u_{ig}^{jg}$, $\hat{U} = \hat{U} - uy$

A28: k=kg, ig=i(v_k), jg=j(v_k) , A11'e git.

A29: PRINT X, UG.

A30: SON.

7.2 ALGORİTMA B

B0: U, P parametreleri çağıran program tarafından belirlenmiş olur.

B1: k = 1, ks=0.

B2: Eğer $j(v_k) \notin P$ ise o zaman adım B7 ye git.

B3: Eğer ks=0 ise ks=k.

B4: $U = U + v_k$

B5: P=P \ $V(v_k)$

B6: Eğer P=Ø ise adım B9 a git.

B7: k = k + 1

B8: Eğer k ≤ p ise adım B2 ye git.

B9: U, ks çıktıları ile algoritma biter.

Burada $j(u_k)$ u_k elemanının yerleştiği satırın numarasıdır.

8. HESAPLAMA DENEMELERİ

Önerilen algoritmanın pratik effektifliğini kontrol etmek için IBM Pentium S CPU 150 MHz de C++ dilinde hazırlanmış programla hesaplama denemeleri yapılmıştır. Bunun için elemanları rasgele sayılar olan farklı boyutlu (u_i^j) matrisleri üretilmiştir. Testlerin gerçek durumlara uygunluğunu tatmin etmek için matrislerin elemanları aşağıdaki kısıtlamaları sağlamak koşulu ile üretilmiştir.

1. Hiçbir j_1 satırı başka bir j_2 satırını majorite etmez, yani matriste $\forall i,\ l_i^{j_1} \geq l_i^{j_2}$ koşulunu sağlayan iki $j_1 \ ve \ j_2$ satırları yoktur.

2. $0 \leq u_i^j \leq 10000,\ i = \overline{1,n},\ j = \overline{1,p}$ ve u_i^j ler tamsayıdırlar.

Üç tür denemeler yapılmıştır. Birinci tür denemelerde küçük boyutlu matrisler için problem önce Dal & Buda yöntemi ile, sonra ise önerilmiş algoritma ile çözülür ve amaç fonksiyonun değerleri

karşılaştırılarak nispi hatalar $\alpha = \frac{U^* - \hat{U}}{U^*}$ formülü ile hesaplanır. Bu zaman p*n boyutlu matrislere bakılır, n' in 5 ve 10 ; p' nin ise 5, 10, 15; 20 değerleri ele alınır (Tablo 1- 2).

Deneme sonuçları aşağıdaki tablolarda gösterilmiştir

Tablo 1. *Küçük girdi matrisleri için sonuçlar*.

No	n=5, p=5			n=5, p=10			n=5, p=15			n=5, p=20		
	Amaç Fonksiyonun Değeri		α	Amaç Fonksiyonun Değeri		α	Amaç Fonksiyonun Değeri		α	Amaç Fonksiyonun Değeri		α
	B&B	Heuristic		B&B	Heuristic		B&B	Heuristic		B&B	Heuristic	
1	543	543	0.00	875	875	0.00	878	878	0.00	326	326	0.00
2	451	451	0.00	675	675	0.00	794	794	0.00	478	478	0.00
3	717	717	0.00	537	537	0.00	938	938	0.00	689	689	0.00
4	478	478	0.00	807	807	0.00	648	648	0.00	934	934	0.00
5	744	744	0.00	662	662	0.00	786	786	0.00	671	671	0.00
6	341	341	0.00	765	765	0.00	806	806	0.00	858	858	0.00
7	565	565	0.00	654	673	0.02	751	751	0.00	736	755	0.02
8	454	454	0.00	521	521	0.00	548	548	0.00	874	874	0.00
9	683	683	0.00	889	889	0.00	693	693	0.00	781	802	0.02
10	789	789	0.00	931	931	0.00	818	818	0.00	915	915	0.00

Tablo 2. *Küçük girdi matrisleri için sonuçlar*.

No	n=10, p=5			N=10, p=10			n=10, p=15			N=10, p=20		
	Amaç Fonksiyonun Değeri		α	Amaç Fonksiyonun Değeri		α	Amaç Fonksiyonun Değeri		α	Amaç Fonksiyonun Değeri		α
	B&B	Heuristic		B&B	Heuristic		B&B	Heuristic		B&B	Heuristic	
1	232	232	0.00	421	421	0.00	737	737	0.00	855	855	0.00
2	352	352	0.00	463	528	0.00	611	611	0.00	778	796	0.02
3	300	300	0.00	617	617	0.00	706	706	0.00	628	628	0.00
4	259	259	0.00	584	584	0.00	768	768	0.05	724	724	0.00
5	374	374	0.00	338	338	0.00	763	763	0.00	854	859	0.01
6	350	350	0.00	694	694	0.00	711	711	0.00	872	880	0.01
7	557	557	0.00	642	642	0.02	403	419	0.03	543	543	0.00
8	147	147	0.00	599	599	0.00	806	849	0.05	685	685	0.00
9	117	117	0.00	508	508	0.00	638	638	0.00	790	808	0.02
10	316	316	0.00	500	500	0.00	548	551	0.01	890	890	0.00

İkinci tür denemelerde ise n=25 ve p=35 kabul ederek problemler Dal&Buda yöntemi ve önerilen algoritma ile çözülerek zaman kıyaslaması verilmiştir (Tablo 3).

Tablo 3. *Zaman Kıyaslaması*

	Heur.	Zaman	B&B	Zaman
1	99.32	0.01 sn.	99.32	9.84 sn.
2	97.83	0.00 sn.	97.83	15.60 sn.
3	99.76	0.00 sn.	99.76	22.36 sn.
4	99.27	0.00 sn.	99.27	85.00 sn.
5	93.32	0.01 sn.	93.32	13.19 sn.
6	99.28	0.01 sn.	99.28	8.96 sn.
7	99.69	0.00 sn.	99.69	6.48 sn.

8	99.35	0.00 sn.	99.35	3.74 sn.
9	99.14	0.00 sn.	99.14	13.74 sn.
10	98.82	0.01 sn.	98.82	27.86 sn.

Üçüncü tür denemelerde büyük boyutlu matrisler ele alınır ve hesaplama zamanı kontrol edilir. Bu zamab p=1000 ve n' in15, 25, 35, 45, 55, 65, 75, 85, 95 değerleri ele alınır (Tablo 4).

Tablo 4. *Büyük girdi matrisleri için sonuçlar.*

No	n	p	Zaman
1	5	1000	0.04 sn.
2	15	1000	0.05 sn.
3	25	1000	0.15 sn.
4	35	1000	0.12 sn.
5	45	1000	0.18 sn.
6	55	1000	0.17 sn.
7	65	1000	0.24 sn.
8	75	1000	0.26 sn.
9	85	1000	0.27 sn.
10	95	1000	0.31 sn.

Denemeler gösterir ki, önerilmiş algoritma çoğunlukla optimal çözümü bulur ve diğer durumlarda ise nispi hata çok küçük olur, başka bir ifadeyle optimal çözümden çok az farklı olur. Tablo 3 ve 4 den ise problemin büyük boyutlu matrisler için de kısa zamanda çözüldüğü görülmektedir.

9. SONUÇ

Bu çalışmada Global Optimizasyon Probleminin çözümünde yardımcı problem gibi kullanılan Minimal Ağırlıklı Dominant Altküme Problemi olarak adlandırdığımız Bir Boole Değişkenli problemin özelliklerine dayanarak yeni bir heuristik algoritma önerilmiştir. Algoritmadaki işlemlerin sayısı $O\left(n \cdot p \cdot \log_2 p\right)$ ile sınırlıdır. C++ dilinde hazırlanmış programla yapılmış denemeler algoritmanın effektifliğinin yüksek olduğunu göstermektedir. Algoritmanın Global Optimizasyon Problemlerini çözmek için Kesen Açılar yöntemi ile birlikte kullanılması düşünülmektedir.

REFERANSLAR

[1] M.Yu.Andramonov, A.M.Rubinov and B.M.Glover , Cutting Angle methods in Global Optimization, *Applied Mathematics Letters*, Vol.12, pp. 95-100, 1999.

[2] Dj.A.Babayev, An Exact Method for Solving the Subproblem of the Cutting Angle Method of Global Optimization In book "Optimization and Related Topics", in Kluwer Academic Publishers, series "Applied Optimization", Vol 47, December, Dordrecht/ Boston/ London, pp. 472-482, 2000..

[3] U.G. Nuriyev , On the solving of a combinatoric problem, Proceeding of XXII National Meeting on Operational Research and Industrial Engineering, Ankara, p.29, 2001.

[4] U.G. Nuriyev , On Transformation of Global Optimization Subproblem., In proc. of 3-th Joint Seminar on Appied Mathematics, Baku State University & Zanjan University, Baku, p. 96, 2002

[5] U.G. Nuriyev , On the solving of a Global Optimization Subproblem, Proceeding of XXIII National Meeting on Operational Research and Industrial Engineering, Istanbul, p.33, 2002.

[6] U.G. Nuriyev and O. Şen , Dominating Subset with Minimal Weigth Problem., Proceeding of the Third International Conference on Matematical & Computational

Applications (ICMCA'2002). Konya, pp. 54–60, 2002.
[7] U.G. Nuriyev , B, Ordin , Dominating Subset with Minimal Weight Problem and the survey of its complexity, Proceeding of XXIII National Meeting on Operational Research and Industrial Engineering, Istanbul, p.33, 2002.
[8]. U.G. Nuriyev , On Complexity of a Global Optimization Problem, Mathematical & Computational Applications, Vol.8, No.1, pp. 27-34, 2003.
[9] Ordin B., On Some Properties of the Solution of the Dominating Subset with Minimal Weight Problem, A Euro Conference for young OR researchers and practitioners ORP^3, September 21-26, Kaiserslautern, Germany, 351-360, 2003.
[10] C.H. Papadimitriou, K. Steiglitz , Combinatorial Optimization:Algorithms and Complexity, Prentice-Hall, 1982.
[11] A.M. Rubinov, *Abstract Convexity and Global Optimization*, Dordrecht, Kluwer Academic Publishers, 2000.