Eskişehir Osmangazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi Cilt : XXV, Sayı : 1, 2012
Journal of Engineering and Architecture Faculty of Eskişehir Osmangazi University, Vol : XXV, No: 1, 2012

Makalenin Geliş Tarihi : 10.08.2011
Makalenin Kabul Tarihi : 18.04.2012

0-1 TAMSAYILI DOĞRUSAL OLMAYAN MATEMATİKSEL MODELLERİN UYGUN ÇÖZÜM TEMELLİ GENİŞLETİLMİŞ SUBGRADİENT ALGORİTMASI İLE ÇÖZÜLMESİ

Tuğba SARAÇ

ÖZET : *Uygun Çözüm Temelli Genişletilmiş Subgadient Algoritması (UÇT-GSA) doğrusal olmayan matematiksel modeller için, 2004 yılında Gasimov ve diğerleri tarafından önerilmiştir. Sivri, genişletilmiş Lagrange fonksiyonu ile kurulmuş ikil problemin çözümüne yönelik bir yaklaşımdır. Bu yöntemin önemli üstünlükleri, çözüm sürecinin yakınsak olması, sıfır ikil aralığın elde edilebilmesi ve sürekli problem üzerine herhangi bir dışbükeylik veya türevlenebilirlik şartı olmaması olarak sayılabilir. Bu çalışmada 0-1 tamsayılı doğrusal olmayan matematiksel modellerin UÇT-GSA ile çözülebilmeleri için bir GAMS kodu geliştirilmiştir ve algoritmanın 0-1 tamsayılı doğrusal olmayan problemlerin çözümündeki başarısı karesel sırt çantası, hücre oluşturma ve dinamik yerleşim problemleri kullanılarak araştırılmıştır.*

ANAHTAR KELİMELER : *0-1 Doğrusal olmayan programlama, Uygun çözüm temelli genişletilmiş subgradient algoritması (UÇT-GSA), Karesel sırt çantası problemi, Hücre oluşturma problemi, Dinamik yerleşim problemi.*

SOLVING THE 0-1 NONLINEAR PROGRAMMING MODELS BY USING THE MODIFIED SUBGRADIENT ALGORITHM BASED ON FEASIBLE VALUES

ABSTRACT : *A modified subgradient algorithm based on feasible values (F-MSG) has been proposed for solving nonlinear mathematical models in 2004 by Gasimov et.al. It is an approach to solve dual problems constructed by sharp augmented Lagrangian function. It has some remarkable features. For example, it is convergent, and it guarantees zero duality gap for the problems such that its objective and constraint functions are all Lipschtz. In this study, a GAMS program has been developed for solving the nonlinear models by using FMSG. Success of the algorithm on solving the 0-1 nonlinear programming problems has been examined by using the quadratic knapsack, cell formation and dynamic layout problems.*

KEY WORDS : *0-1 Nonlinear programming, Modified sub-gradient algorithm based on feasible values (F-MSG), Quadratic knapsack problem, Cell formation problem, Dynamic layout problem.*

Eskişehir Osmangazi Üniversitesi, Mühendislik Mimarlık Fakültesi, Endüstri Müh.Bölümü, Meşelik Kamp., 26480 ESKİŞEHİR

I. GİRİŞ

Doğrusal olmayan modellerin çözümünde uygulanan temel yaklaşım Karush-Kuhn-Tucker koşulları iken, 60'lı yılların ortalarından itibaren subgradient temelli yaklaşımlar önerilmiş ve bu alanda değişik yöntemler geliştirilmiştir. 1998 yılında Rockafellar ve Wets [1] çalışmalarında, sivri (*sharp*) Lagrange fonksiyonunu vermiştir. 2002'de Gasimov [2] tarafından bu fonksiyonu kullanan Genişletilmiş Subgradient Algoritması (GSA) önerilmiş ve bu algoritma ile doğrusal olmayan modellerin etkin bir şekilde çözülebileceği gösterilmiştir. Yöntemin temel avantajları, çözüm sürecinin yakınsak olması, sıfır ikil aralığın elde edilebilmesi ve sürekli problem üzerine herhangi bir dışbükeylik veya türevlenebilirlik şartı olmamasıdır. Ancak sayılan üstünlüklerin yanı sıra yöntemin kullanılmasında karşılaşılan, GSA'nın temel adımındaki kısıtsız problemin çözümünde kullanılan tekniklerin yerel en iyi noktalara takılması ve adım uzunluğu hesabında verilen üst sınırın belirlenmesi gibi bazı zorluklar da mevcuttur. Bu zorlukları ortadan kaldırmak amacıyla yöntem geliştirilerek, 2004 yılında Gasimov ve diğerleri [3] tarafından Uygun Çözüm Temelli Genişletilmiş Subgadient Algoritması (UÇT-GSA) adıyla verilmiştir.

Bu çalışmanın amacı, UÇT-GSA algoritmasının 0-1 tamsayılı doğrusal olmayan modellerin çözümündeki performansını araştırmaktır. Deneysel çalışmalarda örnek problem olarak doğrusal olmayan amaç fonksiyonuna sahip 0-1 tamsayılı programlama problemleri olan karesel sırt çantası, hücre oluşturma ve rota seçimi ve dinamik yerleşim problemleri kullanılmıştır. Yazından alınan test problemleri öncelikle sadece GAMS çözücüleri kullanılarak çözülmüş daha sonra UÇT-GSA, GAMS'te kodlanarak aynı test problemlerinin çözümleri araştırılmıştır. Elde edilen sonuçlar karşılaştırılmıştır. Bu çerçevede, çalışmanın ikinci bölümünde uygun çözüm temelli genişletilmiş subgradient algoritması anlatılmıştır. Üçüncü bölümde algoritmanın GAMS'te nasıl kodlandığına değinilmiş, dördüncü bölümde ise çalışmada kullanılan örnek problemler tanıtılmıştır. Beşinci bölümde elde edilen deneysel sonuçlar tartışılmış, son bölümde ise çalışmanın genel sonuçları ve gelecek çalışmalara yönelik öneriler sunulmuştur.

II. UYGUN ÇÖZÜM TEMELLİ GENİŞLETİLMİŞ SUBGRADİENT ALGORİTMASI

Bilindiği gibi 0-1 tamsayılı modeller, subgradient yöntemler yardımıyla çözülebilmektedirler ancak bu yöntemler, modelin en iyi çözümünü bulamayıp bir yerel en iyi noktasına takılabilmektedir. Bu yöntemler hakkında ayrıntılı bilgilere Bazaraa ve diğerleri [4] ile Bertsekas'ın [5] kitaplarından erişmek mümkündür. 2002 yılında Gasimov [2] tarafından yapılan çalışmalarla sivri, genişletilmiş (*sharp augmented*) Lagrange fonksiyonu kullanan GSA geliştirilmiştir. Yöntemin temel avantajı, çözüm sürecinin yakınsak olduğunun ispatlanmış olması, yani her ardıştırmada bir öncekinden daha iyi bir çözüme erişileceğinin garantilenmesidir. Ayrıca diğer Lagrange fonksiyonlarından farklı olarak, çözüm koniler aracılığıyla taranmakta ve böylece sıfır ikil aralığın elde edilmesi, yani en iyi çözümün bulunabilmesi sağlanmaktadır. Sürekli problem üzerine herhangi bir dışbükeylik veya türevlenebilirlik şartı olmaması da yöntemin önemli bir avantajıdır. Ancak sayılan üstünlüklerin yanı sıra yöntemin kullanılmasında karşılaşılan bazı zorluklar da mevcuttur. Bu zorluklar, daha çok GSA'nın temel adımındaki kısıtsız problemin çözümünde kullanılan tekniklerin yerel en iyi noktalara takılmasından kaynaklanmaktadır. Diğer bir zorluk da, adım uzunluğu hesabında verilen üst sınırın belirlenmesiyle ilgilidir. Bu zorlukları ortadan kaldırmak amacıyla yöntem geliştirilerek, 2004 yılında Gasimov ve diğerleri [3] tarafından UÇT-GSA adıyla verilmiştir. 2006 yılında Saraç ve Sipahioğlu [6] UÇT-GSA'nın karesel sırt çantası problemlerinin çözümünde kullanılabileceğini göstermişlerdir. 2007 yılında Gasimov ve Üstün [7] algoritmanın karesel atama problemlerini çözmedeki başarısını ortaya koymuşlardır. Bu çalışmada yazarlar UÇT-GSA'nın karesel atama probleminin çözümündeki performansını göstermişlerdir.

UÇT-GSA, sivri, genişletilmiş Lagrange fonksiyonu ile kurulmuş ikil problemin çözümüne yönelik bir yaklaşımdır. Ele alınan doğrusal olmayan (P) probleminin matematiksel modeli aşağıda verildiği gibi olsun.

$$(P)\quad \begin{array}{l} f(x)=0 \\ x\in S \\ k.a. \\ enk\ f_0(x) \end{array}$$

Burada S çözüm kümesi ve $f_0 : X \rightarrow R$ ve $f : X \rightarrow R^p$ sırasıyla amaç ve kısıtları gösteren fonksiyonlardır. R_+, pozitif sayılar, $\|\cdot\|$, öklit normu ve $\langle\cdot,\cdot\rangle$, R^p'de iç çarpım olmak üzere, (P) problemi için sivri, genişletilmiş Lagrange fonksiyonu $L : S\times R^p\times R_+ \rightarrow R$ ve ikil fonksiyon $H : R^p\times R_+ \rightarrow R$ aşağıda verilmiştir.

$$L(x,u,c)= f_0(x)+c\|f(x)\|-\langle u, f(x)\rangle \qquad (1)$$

$$H(u,c)=\underset{x\in S}{enk}\,L(x,u,c) \qquad (2)$$

İkil fonksiyonu kullanarak ikil problem (P^*) aşağıdaki gibi yazılabilir.

$$(P^*)\quad \underset{(u,c)\in R^p\times R_+}{enb} H(u,c) \qquad (3)$$

Bu aşamada problem (P^*) ikil modelinin çözümüne dönüşmüştür. Bu problemin çözümüne yönelik olarak geliştirilmiş olan UÇT-GSA'nin adımları [7] ve kullanılan gösterimlerin açıklamaları aşağıda verilmiştir:

$f_0(x)$	: amaç fonksiyonu
$f(x)$	: kısıt fonksiyonları
k	: ikil değişkenlerin güncellenme sayısı
n	: ardıştırma sayısı
u_k ve c_k	: k. güncellemede ikil değişkenlerin değerleri
H_n	: n. ardıştırmada üst sınır değeri
ε_1 ve ε_2	: kabul edilebilir hata miktarları
M	: H_n üst sınır değerinden küçük ya da eşit bir uygun çözümün varlığını araştırırken durdurma toleransı olarak kullanılacak büyük pozitif sayı. ($l(k)$ değeri M değerini aştığında, kısıt sağlama probleminin uygun bir çözümü olmadığı varsayılarak, arama sonlandırılır.)
Δ_n	: üst sınırı güncellemede kullanılan pozitif adım büyüklüğü parametresi
α ve δ	: adım katsayıları
q	: kısıt sağlama probleminin çözüldüğü ve asıl problemin kısıtlarını sağladığı durum sayısı
t	: kısıt sağlama probleminin çözülemediği durum sayısı

Adım 1: ε_1, ε_2, Δ_0 ve M için pozitif, H_0 için herhangi bir sayı ata. $n = 0$, $t = 0$ ve $q = 0$ ata.

Adım 2: Herhangi bir $(u_1^n, c_1^n) \in R^p \times R_+$ seç ve $0 < l(1) < M$ ve k = 1, $u_k = u_1^n$, $c_k = c_1^n$ ata.

Adım 3: Verilen (u_k, c_k) için izleyen kısıt sağlama problemini $P(H_n)$ çöz.

$P(H_n)$: $f_0(x) + c_k \| f(x) \| - \langle u_k, f(x) \rangle \le H_n$ olacak şekilde $x \in S$'i bul. Eğer kısıtı sağlayan bir çözüm bulunamamışsa (örneğin $l(k) > M$ ise) Adım 6'ya git. Eğer çözüm bulunmuşsa, asıl problemin kısıtları sağlanıyor mu kontrol et. Eğer $f(x_k^n) = 0$ (ya da $\| f(x_k) \le \varepsilon_1 \|$) ise, Adım 5'e git.

Adım 4: İkil değişkenleri aşağıdaki formülleri kullanarak güncelle,

$$u_{k+1} = u_k - \alpha s_k f(x_k),$$

$$c_{k+1} = c_k + (1+\alpha) s_k \| f(x_k) \|,$$

Burada s_k pozitif adım parametresidir ve (4) ya da (5) numaralı formüllerden birisi ile hesaplanabilir. Bu çalışmada s_k'nın hesaplanmasında (5) numaralı formül kullanılmıştır.

$$0 < s_k = \frac{\delta\alpha \left(H_n - L(x_k, u_k, c_k) \right)}{\left(\alpha^2 + (1+\alpha)^2 \right) \| f(x_k) \|^2}\ , \tag{4}$$

$$0 < s_k = \frac{\delta\left(\alpha\left(H_n - L(x_k, u_k, c_k)\right) + (\bar{c} - c_k)\|f(x_k)\|\right)}{\left(\alpha^2 + (1+\alpha)^2\right)\|f(x_k)\|^2} \tag{5}$$

Burada $\alpha > 0$, $0 < \delta < 2$ ve $\bar{c} \geq c_k$'dır. Ayrıca adım parametresi s_k'nın ikil değişkenler (u_k, c_k) için izleyen koşulu sağlaması gereklidir.

$$s_k\|f(x_k)\| + c_k - \|u_k\| > l(k),$$

Eğer bu koşul sağlanmıyorsa, $c_k = \gamma\ (c_k+1)$, $\gamma > 1$ olacak şekilde c_k'yı arttır. Burada $l(k)$ fonksiyonu $k \to +\infty$ gittiğinde $l(k) \to +\infty$ gidecek şekilde tariflenmiş herhangi bir fonksiyondur, $l(k)$ fonksiyonunu güncelle, $k = k+1$ ata ve Adım 3'e git.

Adım 5: x_k, $f(x_k) = 0$ olacak şekilde $P(H_n)$ probleminin çözümü iken, $L(x_k, u_k, c_k) = f_0(x_k)$'dir. $q = q + 1$ ata ve t'yi kontrol et. Eğer $t = 0$ ise, $\Delta_{n+1} = \Delta_n$, $t \neq 0$ ise, $\Delta_{n+1} = \frac{1}{2}\Delta_n$ ata. Δ_{n+1}'i kontrol et eğer $\Delta_{n+1} \leq \varepsilon_2$ ise **DUR**. $f(x_k)$ yaklaşık eniyi çözüm değeri, x_k asıl modelin yaklaşık çözümü ve (u_k, c_k)'da ikil problemin yaklaşık çözümüdür. $\Delta_{n+1} > \varepsilon_2$ ise, $H_{n+1} = enk\{f_0(x_k),\ H_n - \Delta_{n+1}\},\ n = n+1$ ata ve Adım 2'ye git.

Adım 6: $t = t + 1$ ata. Eğer $q = 0$ ise, $\Delta_{n+1} = \Delta_n$, değilse, $\Delta_{n+1} = \frac{1}{2}\Delta_n$ ata. $\Delta_{n+1} \leq \varepsilon_2$ ise **DUR**. $H_{n+1} = H_n + \Delta_{n+1}$, $n = n+1$ ata ve Adım 2'ye git.

III. UÇT-GSA ALGORİTMASININ GAMS'TE KODLANMASI

GAMS, matematiksel modellerin ortak bir yapıda kodlanarak farklı çözücülerle çözümlerinin araştırılmasına ve yalnızca modellerin kapalı formda yazılmalarına değil aynı zamanda algoritmaların da kodlanabilmelerine olanak tanıyan bir yazılımdır. Doğrusal, doğrusal olmayan, karma tamsayılı, doğrusal olmayan tamsayılı gibi farklı yapılara sahip eniyileme problemleri, farklı çözücüler kullanılarak çözülebilmektedir.

Klasik programlama dillerindeki döngü yapıları ve eğerli ifadeler kullanılabildiğinden, model çözüldüğünde elde edilen karar değişkeni değerlerini kullanabilmek ve gerektiğinde modeli tekrar başka parametre değerleri ile çözdürebilmek mümkün olduğundan GAMS, algoritmaların kodlanabilmesi için de uygun bir yazılımdır. Bu çalışmada, UÇT-GSA algoritması GAMS kullanılarak kodlanmıştır. Bu kodda, algoritma adımları, **Çizelge 1**'de de verildiği gibi koşullu bir döngü içinde yazılmıştır. Bu döngü ya algoritmanın durma kriterleri sağlandığında (DUR$\neq$1) ya da ardıştırma sayısı (n), izin verilen en büyük değere (M) ulaştığında durmaktadır. Çizelgede adımlar için yazılan kodlar kapalı verilmiştir ve durdurma parametrelerinin güncellenmesini içermektedirler. Herhangi bir algoritma adımının uygulanma koşulu gerçekleştiğinde, ilgili adıma gidilmesini sağlamak üzere bir *adim* değişkeni kullanılmıştır. Ve hangi adıma gidilmesi gerekiyorsa değişkene bu adımın numarası atanmıştır (*adim*=adım no). Her adıma ait kodlar ancak *adim* değişkeni ilgili adımın numarasına eşit olduğunda çalışmaktadır.

Çizelge 1. *UÇT-GSA'nın GAMS kodunun temel yapısı.*

```
While ( (DUR=0) and (n < M)
        if ( adim=2, "adım 2 için yazılan kodlar" );
        if (adim =3, "adım 3 için yazılan kodlar" );
        if (adim =4, "adım 4 için yazılan kodlar" );
        if (adim =5, "adım 5 için yazılan kodlar" );
        if (adim =6, "adım 6 için yazılan kodlar" )
);
```

IV. ÖRNEK PROBLEMLER

Algoritmanın etkinliğini araştırmak üzere örnek problem olarak, 0-1 tamsayılı ve doğrusal olmayan yapıya sahip olan karesel sırt çantası, hücre oluşturma-rota seçimi ve dinamik yerleşim problemleri seçilmiştir. Bu bölümde seçilen örnek problemler tanıtılmıştır.

IV.1.Karesel Sırt Çantası Problemi

Karesel Sırt Çantası Problemi (KSÇP), 0-1 sırt çantası probleminin iki parçanın sırt çantasına birlikte seçilmelerinden dolayı oluşacak ek karları da göz önünde bulunduracak şekilde genelleştirilmiş halidir. Matematiksel modeli aşağıda verilmiştir.

Kümeler

$J = \{j|\ j=1,..,n\}$ Parça dizin kümesi $i, j \in J$

Karar değişkeni

x_j : *j.* parçanın sırt çantasına atanması durumunda 1, diğer durumda 0.

Parametreler

p_j : *j.* parçanın herhangi bir sırt çantasına atanmış olmasının sağlayacağı katkı

q_{ij} : *i.* ve *j.* parçanın aynı sırt çantasına atanmış olmasının sağlayacağı katkı

w_j : *j.* parçanın ağırlığı

c : sırt çantasının kapasitesi

(KSÇP)

$$\sum_{j=1}^{n} w_j x_j \leq c \tag{6}$$

$$x_j \in \{0,1\}, \quad j = 1,......., n \tag{7}$$

$k.a.$

$$enb \quad z = \sum_{j=1}^{n} p_j x_j + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} q_{ij} x_i x_j \tag{8}$$

Modeldeki (6) numaralı kısıt, sırt çantasına atanan parçaların ağırlıkları toplamının ilgili sırt çantasının kapasitesini aşmamasını sağlamaktadır. (7) numaralı kısıt karar değişkenlerinin 0 ya da 1 tamsayı değer almaları gerektiğini gösteren kısıttır. Amaç (8) ise, parçaların sırt çantasına atanmış olmasından dolayı elde edilecek kazançların ve aynı sırt çantasına birlikte seçilmiş olmalarından doğan kazançların toplamlarını enbüyüklemektir.

IV.2.Hücre Oluşturma ve Rota Seçimi Problemi

Hücre Oluşturma ve Rota Seçimi Problemi (HORSP), alternatif rotaların söz konusu olduğu bir üretim sisteminde, parça ve tezgahların hücrelere atanması ve ayrıca parçaların en uygun rotalarının seçilmesi problemidir. Matematiksel modeli aşağıda verilmiştir.

Kümeler

$P = \{p|p=1,..,n\}$ Parça dizin kümesi

$T = \{t|t=1,..,m\}$ Tezgah dizin kümesi

$H = \{h|h=1,..,k\}$ Hücre dizin kümesi

$R = \{r|r=1,.., R_p\}$ Rota dizin kümesi

Karar değişkenleri

x_{phr} : Eğer p parçası r rotası ile h hücresine atandıysa 1, diğer durumda 0.

y_{th} : Eğer t tezgahı h hücresine atandıysa 1, diğer durumda 0.

Parametreler

n : parça sayısı

m : tezgah sayısı

k : hücre sayısı $k \in [1, m]$

R_p : p parçasının rota sayısı

a_{ptr} : eğer p parçası r rotası ile t. tezgaha atandı ise 1, diğer durumda 0.

w_1 : amaç fonksiyonunda hücre dışı eleman sayısı teriminin ağırlığı

w_2 : amaç fonksiyonunda kullanılmayan eleman sayısı teriminin ağırlığı

(HORSP)

$$\sum_r \sum_h x_{phr} = 1 \qquad p = 1,\ldots,n \tag{9}$$

$$\sum_h y_{th} = 1 \qquad t = 1,\ldots,m \tag{10}$$

$$x_{phr},\ y_{th} \in \{0,1\} \qquad p = 1,\ldots,n,\ t = 1,\ldots,m,\ h = 1,\ldots,k,\ r = 1,\ldots,R_p \tag{11}$$

$k.a.,$

$$enk\,z = w_1 \sum_p \sum_t \sum_h \sum_r a_{ptr} x_{phr} (1 - y_{th}) + w_2 \sum_p \sum_t \sum_h \sum_r (1 - a_{ptr}) x_{phr} y_{th} \tag{12}$$

Burada kısıt (9), her parçanın mutlaka bir hücreye atanmasını ve alternatif rotalardan sadece birisinin seçilmesini garanti etmektedir. Kısıt (10), her tezgâhın mutlaka bir hücreye atanmasını sağlamaktadır. Kısıt (11) karar değişkenlerinin tümünün 0,1 tamsayı değişken olduğunu göstermektedir. Modelin amaç fonksiyonu (12) numaralı eşitlikte verilmiştir. Burada ilk terim hücre dışı ve ikinci terim de kullanılmayan eleman sayısını göstermektedir. Amaç, bu ağırlıklandırılmış toplamın en küçüklenmesi olarak belirlenmiştir.

IV.3.Dinamik Yerleşim Problemi

Dinamik Yerleşim Problemi (DYP), klasik yerleşim (karesel atama) probleminin taşıma ve yer değiştirme maliyetlerinin dönemler bazında farklılık gösterdiği çok dönemli yapıya genelleştirilmiş halidir. Matematiksel modeli aşağıda verilmiştir.

Kümeler

$D=\{t|t=1,..,T\}$ Dönem dizin kümesi

$Y=\{j|j=1,..,N\}$ Yer dizin kümesi $j, l \in Y$

$B=\{i|i=1,.., N\}$ Bölüm dizin kümesi $i, k \in B$

Karar değişkenleri

x_{tij} : Eğer i bölümü t döneminde j yerine atandıysa 1, diğer durumda 0.

Parametreler

N : bölüm ya da yer sayısı

T : dönem sayısı

f_{tik} : t döneminde i ve k bölümleri arasındaki akış miktarı

d_{jl} : j ve l yerleri arasındaki uzaklık

A_{tijl} : t. dönemde i bölümünü, j yerinden l yerine taşıma maliyeti

C_{tijkl} : t. dönemde j yerindeki i bölümünden l yerindeki k bölümüne taşıma maliyeti

$$C_{tijkl} = f_{tik} \times d_{jl}$$

$$y_{tijl} = x_{(t-1)ij} \times x_{til}$$

(DYP)

$$\sum_{j=1}^{N} x_{tij} = 1, \quad i = 1,\ldots,N, \quad t = 1,\ldots,T \tag{13}$$

$$\sum_{i=1}^{N} x_{tij} = 1, \quad j = 1,\ldots,N, \quad t = 1,\ldots,T \tag{14}$$

$$x_{tij} \in \{0,1\}, \quad i, j = 1,\ldots,N, \quad t = 1,\ldots,T \tag{15}$$

$k.a.,$

$$enk\, z = \sum_{t=2}^{T} \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{l=1}^{N} A_{tijl}\ y_{tijl} + \sum_{t=1}^{T} \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} \sum_{l=1}^{N} C_{tijkl}\ x_{tij}\ x_{tkl} \tag{16}$$

Kısıt (13) her bölümün, her bir dönemde bir yere atanmasını garanti etmektedir. Kısıt (14) ise her yerin her bir dönemde bir bölgeye tahsis edilmesini sağlamaktadır. Kısıt (15) karar değişkenlerinin tümünün 0,1 tamsayı değişken olduğunu göstermektedir. Modelin amacı (16), tüm dönemler boyunca toplam taşıma ve bölümlerin yer değiştirme maliyetlerinin enküçüklenmesidir.

V. DENEYSEL SONUÇLAR

Algoritmanın etkinliğinin araştırılmasına yönelik olarak her bir örnek problem için yazından alınan test problemleri kullanılmıştır. Öncelikle bu test problemleri doğrudan GAMS çözücüleri kullanılarak çözülmüştür. Problemler hem doğrusal olmayan hem de 0-1 tamsayılı olduğundan MINLP çözücülerini kullanmak gereklidir. Ancak MINLP çözücüler alt problemleri çözerken MIP, LP, ve NLP çözücüleri de çağırabildiklerinden, bu çözücülerin de belirlenmesi gerekmektedir. Problemlerin çözümünde kullanılmak üzere DICOPT, SBB ve BARON MINLP çözücüleri seçilmiştir. Burada tüm MINLP çözücüler için kullanılacak alt çözücüler MIP ve LP için CPLEX, NLP için MINOS olarak belirlenmiştir.

Test problemlerini UÇT-GSA ile çözebilmek için öncelikle sürekli hale getirilmeleri gerekmektedir. Bu çalışmada tüm örnek problemlerin sürekli hale dönüştürülmesinde Li [8] tarafından önerilen (17) numaralı kısıt modellere eklenmiştir.

$$\sum_{i=1}^{n}(x_i - x_i^2) = 0 \tag{17}$$

Burada x_i 0-1 tamsayı değişkenleri temsil etmektedir ve ilgili değişkenlerin işaret kısıtları da $0 \le x_i \le 1$ olarak değiştirilmelidir. Böylece sürekli hale dönüştürülmüş olan test problemleri daha sonra GAMS'te kodlanmış olan UÇT-GSA kullanılarak çözülmüş ve elde edilen sonuçlar her bir örnek problem için ayrı başlıklar altında sunulmuştur. Bu bölümde sonuçları sunulan tüm testler, Pentium Core2 Duo, 1,87 GB RAM, 2 GHz özelliklerini taşıyan bir bilgisayar ve 21,5 versiyonlu GAMS paket programı kullanılarak gerçekleştirilmiştir.

V.1.KSÇP ile ilgili testler

Bu bölümde kullanılan test problemleri ve eniyi çözümleri literatürden [9] alınmıştır. Ele alınan problemlerde parça sayısı (n) 100'dür ve bu problemlerin yoğunlukları (sıfırdan farklı p_j ve q_{ij} parametre oranı) d = 0,25'tir. Bu problemlerin eniyi çözümleri Billionnet and Soutif [10], tarafından bulunmuştur. Çizelge 2'de ilgili test problemlerinin örnek numaraları (*no*) ve eniyi çözüm değerleri (z_{enb}) verilmiştir.

Çizelge 2. *KSÇP Test Problemlerinin En İyi Çözümleri.*

no	1	2	3	4	5	6	7	8	9	10
z_{enb}	18558	56525	3752	50382	61494	36360	14657	20452	35438	24930

Öncelikle test problemleri doğrudan GAMS çözücüleri kullanılarak çözülmüştür. Elde edilen sonuçlar Çizelge 3'te verilmiştir. Çizelge 3'ün sol tarafında test problemlerinin numarası (*no*) yer almaktadır. Çizelgenin ikinci, üçüncü ve dördüncü bölümlerinde ise sırasıyla DICOPT, SBB ve BARON ile elde edilmiş çözüm değerleri (z), eniyi çözümden yüzde olarak fark (*%E*) ve saniye olarak çözüm süreleri verilmiştir. Problemlerin eniyi çözüm değerlerinin elde edilebildiği sonuçlar koyu ve altı çizili olarak yazılmıştır.

Çizelge 3. *KSÇP Test Problemlerinin GAMS Çözücüleri ile Elde Edilen Çözümleri.*

	Doğrusal Olmayan Modeller için GAMS Çözücüleri								
	DICOPT			**SBB**			**BARON**		
no	*z*	*%E*	*süre*	*z*	*%E*	*süre*	*z*	*%E*	*süre*
1	18511	0,25	0,74	18485	0,39	0,82	18514	0,24	0,78
2	**56525**	0,00	0,53	55717	1,43	1,62	55578	1,68	4,97
3	**3752**	0,00	0,37	3717	0,93	1,31	3538	5,70	2,33
4	**50382**	0,00	0,08	**50382**	0,00	0,13	**50382**	0,00	0,89
5	**61494**	0,00	0,31	61213	0,46	0,36	60983	0,83	0,25
6	**36360**	0,00	0,46	36137	0,61	0,63	36137	0,61	0,89
7	**14657**	0,00	0,35	14072	3,99	1,65	14282	2,56	0,78
8	20369	0,41	0,78	19932	2,54	1,27	19932	2,54	0,66
9	**35438**	0,00	0,37	**35438**	0,00	0,66	34924	1,45	0,81
10	**24915**	0,06	0,59	24503	1,71	1,07	24748	0,73	0,78
ortalama		0,07	0,46		1,21	0,95		1,63	1,31

Çizelge 3'den de görülebileceği gibi GAMS DICOPT çözücüsü ile 10 test probleminin 7'sinde eniyi çözümüne ulaşılmış, sadece 3 test probleminde ise %1'in oldukça altında bir hata payı ile oldukça yaklaşılmıştır. GAMS SBB çözücüsü ile iki ve GAMS BARON çözücüsü ile de yalnızca bir test probleminin eniyi değerlerine ulaşılabilmiştir. Bu iki çözücü, çözüm süresi açısından da DICOPT'un gerisinde kalmışlardır.

Daha sonra aynı test problemleri, GAMS'te kodlanmış olan UÇT-GSA algoritması kullanılarak çözülmüş ve elde edilen sonuçlar Çizelge 4'de sunulmuştur. NLP çözücüsü olarak MINOS ve LP için de CPLEX seçilmiştir. Çizelge 4'ün en solunda test problemlerinin çözümünde kullanılan parametre değerleri verilmiştir. Çizelgenin ikinci bölümünde, UÇT-GSA ile elde edilmiş amaç fonksiyonu değeri (*z*), eniyi çözümden yüzde olarak fark (*%E*) ve saniye olarak çözüm süreleri verilmiştir.

Çizelge 4'ten de görülebileceği gibi, elde edilen çözümler GAMS MINLP sonuçları ile karşılaştırıldığında hem eniyi çözüme yaklaşma hem de çözüm süresi yönüyle çözücülerin gerisinde kalmaktadır. Öte yandan, UÇT-GSA, beşinci ve sekizinci test problemlerinin eniyi çözümüne ulaşabilmiştir. DICOPT ile sekizinci problemin eniyi çözümüne ulaşılamamış olması dikkat çekicidir.

Çizelge 4. KSÇP Test Problemlerinin UÇT-GSA ile Çözümleri.

	parametre değerleri						UÇT-GSA		
no	δ	α	$\overline{H}$	γ	*M*	Δ	*z*	*%E*	*süre*
1	1,75	4,5	-1500	3	20	1000	17633	4,98	645
2	1,90	10	-100	9	20	2000	56452	0,13	275
3	1,30	10	-10	9	20	200	3717	0,93	4.0
4	1,95	5	-5000	3	20	2000	48768	3,20	387
5	1,55	3	-100	3	20	2000	61494	0,00	625
6	1,95	9	-500	2	20	1000	36041	0,88	434
7	1,55	3	-100	3	20	2000	14049	4,15	14
8	1,95	4	-500	2	20	1000	20452	0,00	254
9	1,95	2	-500	2	20	1000	35294	0,41	476
10	1,95	2,5	-100	2	20	1000	24912	0,07	141
						ortalama		1,48	361

***Çizelge 5.** Farklı NLP çözücüleri kullanılarak elde edilen UÇT-GSA Çözümleri.*
(δ=1.95, α=10, $\overline{H}$ =0, γ=3, M=20, Δ=1000)

	(*n*=100, *d*=0.25)	conopt			minos		
no	*eniyi değer*	*z*	*E%*	*süre*	*z*	*E%*	*süre*
1	18558	-	-	-	**18528**	0,16%	328
2	56525	**56452**	0,13%	5,94	**56452**	0,13%	268
3	3752	**3656**	2,56%	4,37	3532	5,86%	93
4	50382	48768	3,20%	4,08	**50242**	0,28%	285
5	61494	**60955**	0,88%	4,43	60245	2,03%	151
6	36360	33819	6,99%	4,47	**36197**	0,45%	213
7	14657	13845	5,54%	2,7	**14158**	3,40%	136
8	20452	-	-	-	**20032**	2,05%	2510
9	35438	**35325**	0,32%	3,92	34924	1,45%	204
10	24930	**24748**	0,73%	3,57	**24748**	0,73%	578
	ortalama		2,54%	4,19		1,66%	476,6

Aynı test problemleri son olarak sabit parametre değerleri (δ=1,95, α=10, $\overline{H}$ =0, γ=3, *M=20,* Δ=1000) kullanılarak UÇT-GSA ile çözülmüştür. Testlerde NLP çözücüsü olarak MINOS ve CONOPT, LP için de CPLEX kullanılmıştır. Elde edilen sonuçlar Çizelge 5'te verilmiştir. Çizelge 5'ten de görülebileceği gibi CONOPT çözücüsü kullanıldığında iki test problemine çözüm bulunamamıştır. Yine CONOPT çözücüsü kullanılarak elde edilen çözümler MINOS kullanılarak elde edilen çözümlere kıyasla eniyi çözüme daha uzaktır. Ancak çözüm süreleri dikkat çekici bir oranda daha kısadır. Bu testlerden çıkarılabilecek bir başka sonuçta parametre değerleri probleme özel değil de sabit alındığında ortalama hata oranlarının artmış olmasıdır. Bu da aslında UÇT-GSA'nın parametre değerlerine duyarlı olduğunun ve uygun parametre değerleri seçildiğinde daha iyi sonuçlara erişilebileceğinin bir göstergesidir.

V.2.HORSP ile ilgili testler

Bu bölümde kullanılan test problemlerinin kaynağı, tezgah sayısı (*M*), parça sayısı (*P*), toplam rota sayısı (*R*) ve problemin kaynağından alınan amaç fonksiyonu değeri (*z*) (w_1=0,5, w_2=0,5) Çizelge 6'da verilmiştir.

Çizelge 6. *HORSP örneklerinin özellikleri.*

no	*problemin kaynağı*	*M / P / R*	*z*
1	[11]	4 / 5 / 11	0,5
2	[12]	4 / 5 / 12	0,5
3	[13]	6 / 6 / 13	1,5
4	[14]	11 / 10 / 22	3,0
5	[14]	26 / 28 / 71	20,5

Öncelikle test problemleri w_1=0,5, w_2=0,5 alınarak, doğrudan GAMS çözücüleri ile çözülmüştür. Elde edilen sonuçlar Çizelge 7'de verilmiştir. Çizelge 7'nin sol tarafında test problemlerinin numarası (*no*) yer almaktadır. Çizelgenin ikinci, üçüncü ve dördüncü bölümlerinde ise sırasıyla DICOPT, SBB ve BARON ile elde edilmiş çözüm değerleri (*z*) ve saniye olarak çözüm süreleri verilmiştir. Problemlerin elde edilen en başarılı sonuçlar koyu ve altı çizili olarak yazılmıştır. Çizelge 7'den de görülebileceği gibi GAMS BARON çözücüsü ile 5 test probleminin ilk 4'ünde literatürde verilen çözümlere ulaşılmıştır, ayrıca BARON çözücüsü ile elde edilen çözümlerin tamamı diğer iki çözücüden daha başarılıdır.

Çizelge 7. *HORSP Test Problemlerinin GAMS Çözücüleri ile Elde Edilen Çözümleri.*

	Doğrusal Olmayan Modeller için GAMS Çözücüleri					
	DICOPT		**SBB**		**BARON**	
no	*z*	*süre*	*z*	*süre*	*z*	*süre*
1	2,5	0,16	2,5	0,06	**<u>0,5</u>**	0,08
2	2,5	0,09	2,5	0,16	**<u>0,5</u>**	0,05
3	2,0	0,19	2,0	0,10	**<u>1,5</u>**	0,17
4	8,0	0,23	8,0	0,25	**<u>3,0</u>**	120,09
5	53,5	0,22	53,5	0,57	**<u>34,5</u>**	1006,03
ort		0,18		0,23		225,28

Daha sonra aynı test problemleri, GAMS'te kodlanmış olan UÇT-GSA kullanılarak çözülmüş ve elde edilen sonuçlar Çizelge 8'de sunulmuştur. NLP çözücüsü olarak MINOS ve LP için de CPLEX seçilmiştir. Çizelge 8'in ilk bölümünde test problemlerinin çözümünde kullanılan parametre değerleri verilmiştir. Çizelgenin ikinci bölümünde ise, UÇT-GSA ile elde edilmiş amaç fonksiyonu değeri (*z*) ve saniye olarak çözüm süreleri verilmiştir.

Çizelge 8. *HORSP Test Problemlerinin UÇT-GSA ile Çözümleri .*

	parametre değerleri						UÇT-GSA	
no	δ	α	$\overline{H}$	γ	M	Δ	z	Süre
1	1,75	3,0	0	2	20	10	0,5	4,80
2	1,75	3,0	0	2	20	10	0,5	6,38
3	1,75	3,0	0	2	20	10	1,5	9,19
4	1,55	5,0	0	2	20	10	3,0	18,31
5	1,55	5,0	0	2	20	10	26	314,79
						ortalama		70,69

Çizelge 8'den de görülebileceği gibi, elde edilen çözümler, HORSP için en başarılı çözümleri üreten GAMS MINLP çözücüsü olan BARON'un sonuçları ile karşılaştırıldığında hem eniyi çözüm kalitesi hem de çözüm süresi yönüyle öne çıkmıştır.

V.3.DYP ile ilgili testler

Bu bölümde literatürden alınan iki test problemi : (1), 6 bölüm ve 5 dönemin olduğu Rosenblatt [15] ve (2), 9 bölüm ve 5 dönemin olduğu Conway ve Ventakaraman [16] kullanılmıştır.

Çizelge 9. *DYP'nin GAMS Çözücüleri ile Elde Edilen Çözümleri.*

	Doğrusal Olmayan Modeller için GAMS Çözücüleri					
	DICOPT		**SBB**		**BARON**	
no	z	Süre	z	Süre	z	Süre
(1)	82389	0,47	**76992**	37,45	87625	1000,08
(2)	648161	2,25	**631466**	79,55	721127	1001,89
ortalama		1,36		58,5		1000,99

Öncelikle test problemleri doğrudan GAMS çözücüleri kullanılarak çözülmüştür. Elde edilen sonuçlar Çizelge 9'da verilmiştir. Çizelge 9'un sol tarafında test problemlerinin numarası (*no*) yer almaktadır. Çizelgenin ikinci, üçüncü ve dördüncü bölümlerinde ise sırasıyla DICOPT, SBB ve BARON ile elde edilmiş çözüm değerleri (z) ve saniye olarak çözüm süreleri verilmiştir. Problemlerin elde edilen en başarılı çözümler koyu ve altı çizili olarak yazılmıştır.

Çizelge 9'dan da görülebileceği gibi GAMS SBB çözücüsü ile her iki test problemi için de en başarılı çözümler elde edilmiştir.

Daha sonra aynı test problemleri, GAMS'te kodlanmış olan UÇT-GSA algoritması kullanılarak çözülmüş ve elde edilen sonuçlar Çizelge 10'da sunulmuştur. NLP çözücüsü olarak MINOS ve LP için de CPLEX seçilmiştir. Çizelge 10'nun en solunda test problemlerinin çözümünde kullanılan parametre değerleri verilmiştir. Çizelgenin ikinci bölümünde, UÇT-GSA ile elde edilmiş amaç fonksiyonu değeri (z) ve saniye olarak çözüm süreleri verilmiştir.

Çizelge 10. *DYP Test Problemlerinin UÇT-GSA ile Çözümleri.*

parametre değerleri						**UÇT-GSA**	
δ	α	$\overline{H}$	γ	M	Δ	z	Süre
1,95	2	50000	3	20	10000	86864	142,63
1,95	10	300000	10	20	100000	536227	560,11
					ortalama		351,37

Çizelge 10'dan da görülebileceği gibi, elde edilen çözümler GAMS MINLP sonuçları ile karşılaştırıldığında (1) numaralı test problemi için BARON çözücüsünden, (2) numaralı test problemi için ise tüm çözücülerden daha başarılı bir çözüm makul süreler içinde elde edilmiştir.

VI. SONUÇ VE ÖNERİLER

UÇT-GSA, doğrusal olmayan matematiksel modellerin, sivri, genişletilmiş Lagrange fonksiyonu ile kurulmuş ikil problemin çözümüne dayalı bir yaklaşımdır. Çözüm sürecinin yakınsak olması, sıfır ikil aralığın elde edilebilmesi ve sürekli problem üzerine herhangi bir dışbükeylik veya türevlenebilirlik şartının olmaması algoritmanın önemli özelliklerindendir. Doğrusal olmayan 0-1 tamsayılı problemler de sürekli hale dönüştürülerek [8] bu yöntem yardımı ile çözülebilmektedirler.

GAMS, matematiksel modellerin ortak bir yapıda kodlanarak farklı çözücülerle çözümlerinin araştırılmasına olanak tanıyan ve algoritmaların da kodlanabildiği bir yazılımdır. UÇT-GSA gibi karmaşık algoritmaların GAMS'te kodlanması birçok farklı problemin, birçok farklı çözücü kullanılarak çözümünün araştırılabilmesini sağlayacağından önemlidir.

Bu çalışmada 0-1 tamsayılı doğrusal olmayan matematiksel modellerin GAMS çözücüleri kullanılarak UÇT-GSA ile çözülebilmeleri için bir GAMS kodu geliştirilmiştir. Örnek problem olarak KSÇP, HORSP ve DYP çözülerek literatür sonuçları ile karşılaştırılmıştır. Ümit verici sonuçlar elde edilmiştir.
Gelecekte UÇT-GSA'nın farklı yapıdaki problemleri çözmedeki performansı da araştırılabilir. Özellikle, UÇT-GSA'nın parametre değerlerinin performansı arttıracak şekilde belirlenebilmesine yönelik çalışmalar çok daha başarılı çözümlere erişilebilmesine olanak sağlayacaktır.

KAYNAKLAR

[1] R.T. Rockafellar, R.J.-B Wets, "*Variational Analysis*", Springer, Berlin, 1998.

[2] R.N. Gasimov, "Augmented Lagrangian duality and nondifferantiable optimization methods in nonconvex programming", *Journal of Global Optimization*, Vol.24, pp.187-203, 2002.

[3] R.N. Gasimov, A.M. Rubinov, O. Ustun, "The Modified Subgradient Algorithm Based on Feasible Dual Values and Solving the Quadratic Assignment Problems", *International Conference on Continuous Optimization* (ICCOPT-I) August 2-4, Rensselaer Polytechnic Institute, Troy, New York, pp.31-32, 2004.

[4] M.S. Bazaraa, H.D. Sherali, C.M. Shetty, "*Nonlinear Programming. Theory and Algorithms*", John Wiley & Sons, Inc., New Jersey, 2006.

[5] D.P. Bertsekas, "*Nonlinear Programming*", Athena Scientific, Belmont, MA, 1995.

[6] T. Saraç, A. Sipahioğlu, "*Karesel sırt çantası problemi için subgradient temelli bir çözüm yaklaşımı*", Yöneylem Araştırması ve Endüstri Mühendisliği 26. Ulusal Kongresi, 3-5 Temmuz, İzmit-Kocaeli, pp.202-205, 2006.

[7] R.N. Gasimov, O. Ustun, "Solving the quadratic assignment problem using F-MSG algorithm", *Journal of Industrial and Management Optimization*, Vol.3, No.2, pp.173-191, 2007.

[8] H.L. Li, "An approximate method for local optima for nonlinear mixed integer programming problems", *Computers and Operations Research*, Vol.19, No.5, pp.435-444, 1992.

[9] http://cedric.cnam.fr/~soutif/QKP/

[10] A. Billionnet, E. Soutif, "An exact method based on Lagrangean decomposition for the 0-1 quadratic knapsack problem", *European Journal of Operational Research*, Vol.157, No.3, pp.565-575, 2003.

[11] A. Kusiak, "The generalized group technology concept", *International Journal of Production Research*, Vol.25, pp.561-569, 1987.

[12] G.K. Adil, D. Rajamani, D. Strong, “Cell formation considering alternate routings”. *International Journal of Production Research*, Vol.34, pp.1361–1380, 1996.

[13] Y.B. Moon, S.C. Chi, “Generalized part-family formation using neural network techniques”, *Journal of Manufacturing Systems,* Vol.11, pp.149-160, 1992.

[14] Y.K. Won, S.H. Kim, “Multiple criteria clustering algorithm for solving the group technology problem with multiple process routings”. *Computers and Industrial Engineering*, Vol.32, pp.207–220, 1997.

[15] M.J. Rosenblatt, “The Dynamics of Plant Layout”, *Management Science*, Vol.3, pp.76-86, 1986.

[16] D.G. Conway, M.A. Ventakaraman, “Genetic search and the dynamic facility layout problem”, *Computers and Operations Research*, Vol.21, No.8, pp.955-960, 1994.