



**DOĞRUSAL PROGRAMLAMA ÇÖZÜCÜLERİNİN  
PERFORMANS ANALİZİ**

**Merve CÖMERTOĞLU ARIK**



T.C.  
BURSA ULUDAĞ ÜNİVERSİTESİ  
FEN BİLİMLERİ ENSTİTÜSÜ

## DOĞRUSAL PROGRAMLAMA ÇÖZÜCÜLERİNİN PERFORMANS ANALİZİ

Merve CÖMERTOĞLU ARIK  
orcid ID: 0000-0002-9371-092X

Doç. Dr. Fatih ÇAVDUR  
(Danışman)

YÜKSEK LİSANS TEZİ  
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

BURSA – 2020  
Her Hakkı Saklıdır

## TEZ ONAYI

Merve CÖMERTOĞLU ARIK tarafından hazırlanan “DOĞRUSAL PROGRAMLAMA ÇÖZÜCÜLERİNİN PERFORMANS ANALİZİ” adlı tez çalışması aşağıdaki jüri tarafından oy birliği ile Bursa Uludağ Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalı’nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

**Danışman** : Doç. Dr. Fatih ÇAVDUR

**Başkan** : Doç. Dr. Fatih ÇAVDUR  
0000-0001-8054-5606  
Uludağ Üniversitesi,  
Mühendislik Fakültesi,  
Endüstri Mühendisliği Anabilim Dalı

İmza

**Üye** : Prof. Dr. Hüseyin Cenk ÖZMUTLU  
0000-0003-2540-9657  
Uludağ Üniversitesi,  
Mühendislik Fakültesi,  
Endüstri Mühendisliği Anabilim Dalı

İmza

**Üye** : Dr. Yunus DEMİR  
0000-0003-3868-1860  
Bursa Teknik Üniversitesi,  
Mühendislik ve Doğa Bilimleri Fakültesi,  
Endüstri Mühendisliği Anabilim Dalı

İmza

Yukarıdaki sonucu onaylarım

Prof. Dr. Hüseyin Aksel EREN  
Enstitü Müdürü

././....

**U.Ü. Fen Bilimleri Enstitüsü, tez yazım kurallarına uygun olarak hazırladığım bu tez çalışmada;**

- tez içindeki bütün bilgi ve belgeleri akademik kurallar çerçevesinde elde ettiğimi,
- görsel, işitsel ve yazılı tüm bilgi ve sonuçları bilimsel ahlak kurallarına uygun olarak sunduğumu,
- başkalarının eserlerinden yararlanması durumunda ilgili eserlere bilimsel normlara uygun olarak atıfta bulunduğumu,
- atıfta bulunduğum eserlerin tümünü kaynak olarak gösterdiğimi,
- kullanılan verilerde herhangi bir tahrifat yapmadığımı,
- ve bu tezin herhangi bir bölümünü bu üniversite veya başka bir üniversitede başka bir tez çalışması olarak sunmadığımı

**beyan ederim.**

18.02.2020



**Merve CÖMERTOĞLU ARIK**

## ÖZET

Yüksek Lisans Tezi

DOĞRUSAL PROGRAMLAMA ÇÖZÜCÜLERİNİN PERFORMANS ANALİZİ

**Merve CÖMERTOĞLU ARIK**

Bursa Uludağ Üniversitesi  
Fen Bilimleri Enstitüsü  
Endüstri Mühendisliği Anabilim Dalı

**Danışman:** Doç. Dr. Fatih ÇAVDUR

Matematiksel programlama modellerinin çözümü için kullanılabilir farklı çözümler mevcuttur. Bu çözümlerin performansları farklılık gösterebilmektedir. Alternatif çözümlerin ortaya çıkması sonucu daha hızlı çözüme ulaşan çözümler tercih edilmesi söz konusu olabilmektedir. Bu çalışma kapsamında bazı doğrusal programlama modelleri çözümleri karşılaştırılarak yetenek ve hız açısından performanslarının analiz edilmesi amaçlanmıştır. Ticari çözümlerin maliyetli olmaları ve lisans gereksinimleri nedeniyle, alternatif olarak açık kaynak kodlu çözümlerin kullanılıp kullanılmayacağını araştırılması da bu çalışmanın amaçlarından biridir. Elde edilen sonuçların konuyla ilgili çalışma yapacaklara, çözümler performanslarıyla ilgili fikir vermesi amaçlanmaktadır.

**Anahtar Kelimeler:** Matematiksel programlama, Doğrusal programlama, Doğrusal programlama çözümleri, Çözümler performansı, Ticari çözümler, Açık kaynak kodlu çözümler, Performans analizi

## ABSTRACT

MSc Thesis

PERFORMANCE ANALYSIS OF LINEAR PROGRAMMING SOLVERS

**Merve CÖMERTOĞLU ARIK**

Bursa Uludağ University  
Graduate School of Natural and Applied Sciences  
Department of Industrial Engineering

**Supervisor:** Doç. Dr. Fatih ÇAVDUR


There are different solvers that can be used for the solution of mathematical programming models. The performance of these solvers may differ. It may be preferable to use solvers that reach a faster solution as a result of the development of alternative solvers. In this study, it is aimed to analyze their performance in terms of ability and speed by comparing some linear programming models solvers. One of the aims of this study to search whether open-source solvers can be used as an alternative due to the cost of commercial solvers and license requirements. The results of this study obtained are intended to give ideas about solver performances to those who will work on the subject.

**Key words:** Mathematical programming, linear programming, linear programming solver, solver performance, commercial solver, open-source solver, performance analysis

## TEŐEKKÜR

Bu tezde, Uludađ Üniversitesi'nde yapmış olduđum matematiksel programlama çözücülerinin performans analizi konulu tez çalışmam sonucunda elde ettiđim bilgileri dikkatinize sunmaktayım. Bu çalışmayı hazırlarken geçirdiđim süreçte benden yardımlarını esirgemeyen başta Doç. Dr. Fatih ÇAVDUR olmak üzere Uludađ Üniversitesi Endüstri Mühendisliđi bölümündeki hocalarıma, ayrıca manevi desteđini her an yanımda hissettiđim eşime ve aileme teşekkürlerimi sunarım.

Merve CÖMERTOĐLU ARIK

18.02.2020  


## İÇİNDEKİLER

	<b>Sayfa</b>
ÖZET.....	i
ABSTRACT.....	ii
TEŞEKKÜR.....	iii
SİMGELER DİZİNİ.....	v
ŞEKİLLER DİZİNİ.....	vi
ÇİZELGELER DİZİNİ .....	vii
1. GİRİŞ .....	1
2. KURAMSAL TEMELLER ve KAYNAK ARAŞTIRMASI .....	3
2.1. Doğrusal Programlama.....	3
2.2. Tamsayılı Doğrusal Programlama .....	12
2.3. Doğrusal Programlama Çözücüleri .....	15
2.4. Kaynak Araştırması.....	17
2.5. Doğrusal Programlama Çözücüleri için Kullanılan Platformlar .....	22
3. MATERYAL VE YÖNTEM .....	23
3.1. Test Problemleri .....	23
3.2. Seçilen Çözücülerin Test Edilmesi .....	25
4. BULGULAR ve TARTIŞMA.....	27
5. SONUÇ .....	33
KAYNAKLAR .....	34
EKLER.....	36



## SİMGELER DİZİNİ

Simgeler	Açıklama
$z$	Amaç fonksiyonu değeri
$c$	Maliyet katsayıları
$x$	Karar değişkenleri
$a$	Teknolojik katsayılar
$B$	Temel değişkenler matrisi
$N$	Temel olmayan değişkenler matrisi
$w$	Simpleks çarpanları vektörü



## ŞEKİLLER DİZİNİ

	<b>Sayfa</b>
Şekil 2.1. Karışık tamsayılı programlama sınıfları .....	13
Şekil 3.1. Netlib problemlerinin kısıt ve değişken sayıları grafiği .....	25
Şekil 4.1. Netlib problemlerinin tüm çözümler için çözüm süreleri grafiği .....	28
Şekil 4.2. Tüm problemlerin en iyi sonuçları elde ettiği çözümler.....	30
Şekil 4.3. Tüm problemlerin en kötü sonuçları elde ettiği çözümler.....	30
Şekil 4.4. Tüm çözümlerin standart sapma değerleri.....	32
Şekil 4.5. “agg2” probleminin çözüm sonuçları.....	32



## ÇİZELGELER DİZİNİ

	<b>Sayfa</b>
Çizelge 2.1. Literatür çalışma özetleri .....	20
Çizelge 3.1. Netlib veri setindeki problemlerin temel özellikleri .....	24
Çizelge 4.1. Ip_solve çözücü ile optimal çözüme ulaşmayan problemler .....	27
Çizelge 4.2. Tüm çözücüler için en uzun sürede çözülen problem isimleri .....	28
Çizelge 4.3. Tüm çözücüler için çözüm süreleri istatistikleri .....	29
Çizelge 4.4. Tüm problemlerin en iyi ve en kötü sonucu elde ettiği çözücüler .....	29
Çizelge 4.5. Tüm çözücüler için tekrarlı çözüm süreleri istatistikleri .....	31



## 1. GİRİŞ

Bazaraa ve ark. (2010) tarafından doğrusal programlama, bir dizi doğrusal eşitlik ve / veya eşitsizlik kısıtlamalarını veya sınırları karşılarken doğrusal bir fonksiyonun optimizasyonu (minimizasyon veya maksimizasyon) şeklinde tanımlanmıştır. Doğrusal programlama problemi ilk olarak 1947 yılında George B. Dantzig tarafından, Birleşik Devletleri Hava Kuvvetleri biriminde zaman aşamalı bir dağıtım, eğitim ve lojistik tedarik programı için mekanize bir planlama aracı geliştirme konusunda matematik danışanı olarak çalışırken ortaya çıkmıştır. Sovyet matematikçi ve ekonomist L. V. Kantorovich, 1939'da organizasyon ve planlama ile ilgili bu tür bir problemi formüle edip çözmüş olsa da, çalışmaları 1959'a kadar bilinmemiştir. Dantzig, yayınlanan ilk makalesinde, Hava Kuvvetleri biriminde uygulanacak plan ve programları doğrusal yapıda bir programlama modeli olarak ele almıştır. Doğrusal programlama terimi, ilk kez 1948 yılında iktisatçı ve matematikçi T. C. Koopmans tarafından kullanılmıştır.

1949 yılında George B. Dantzig tarafından doğrusal programları çözmek için Simplex Yöntemi ortaya konulmuştur (Bazaraa ve ark. 2010). Bu tarihten itibaren, farklı kişiler teorik gelişmeler, hesaplama yönleri ve konunun yeni uygulamalarının araştırılması dahil olmak üzere birçok farklı yöntem ile doğrusal programlama alanına katkıda bulunmuştur. Simpleks doğrusal programlama yöntemi, önemli ve karmaşık yönetim karar problemlerini modelleme yeteneği ve makul bir sürede çözüm üretme kabiliyeti nedeniyle oldukça fazla kabul görmektedir.

J. Laderman, Ulusal Standartlar Bürosunda 9 eşitlik kısıtı ve 27 negatif olmayan değişken içeren bir diyet planlama doğrusal programını çözmüştür. Hesap makineleri kullanılarak bu problemin çözümü 120 adam – gün sürmüştür ve çalışmanın sayfaları birbirine yapıştırılarak muhafaza edilmiştir. Günümüzde ise modern çağdaki bilgisayar olanakları ve Simpleks yönteminin gelişmiş uygulamaları kullanılarak, on binlerce kısıt ve değişkene sahip doğrusal programlar kolayca çözülebilmektedir. Ne kadar Simpleks yönteminin çeşitli varyantları geliştirilmiş ve diğer yeni rakip algoritmalar ileri sürülmüş olsa da Simpleks yöntemi doğrusal problemlerini çözmek için uygulanabilir ve popüler bir araç olmaya devam etmektedir (Bazaraa ve ark. 2010).

Çalışmanın içeriği genel olarak beş bölüme ayrılmıştır. 2. bölümde doğrusal programlama modeli temelleri, Simpleks yöntemi, Dual programlama modeli temelleri, Dual Simpleks yöntemi, Primal – Dual yöntem, İç Nokta algoritması, tamsayı doğrusal programlama modeli temelleri teorik olarak açıklanmıştır. Bunlara ek olarak, matematiksel programlama modellerinin çözümünde kullanılan çözücüler, literatürde daha önce konuyla ilgili yapılmış çalışmalara genel bakış ve matematiksel programlama modellerinin çözücülerinin kullanıldığı platformlar yer almaktadır. Çalışmanın 3. bölümünde, test edilen Netlib (<https://www.netlib.org/>, 2020) veri seti ve problemlerin belirlenen çözücüler ile test edilmesi anlatılmıştır. 4. bölümde çalışma sonucu elde edilen bulgular ve 5. bölümde çalışma sonuçları sunulmuştur.



## 2. KURAMSAL TEMELLER ve KAYNAK ARAŞTIRMASI

### 2.1. Doğrusal Programlama

Doğrusal programlama problemi formülasyonu aşağıdaki gibi özetlenebilir. Aşağıda  $c_1x_1 + c_2x_2 + \dots + c_nx_n$  ile gösterilen ve minimize edilmek istenen amaç fonksiyonudur.  $c_1, c_2, \dots, c_n$  katsayıları bilinen maliyet katsayıları ve  $x_1, x_2, \dots, x_n$  karar değişkenleridir.

$$\begin{array}{llllllllll} \min & z & = & c_1x_1 & + & c_2x_2 & + & \dots & + & c_nx_n \\ \text{kısıtlar} & & & a_{11}x_1 & + & a_{12}x_2 & + & \dots & + & a_{1n}x_n & \geq & b_1 \\ & & & a_{21}x_1 & + & a_{22}x_2 & + & \dots & + & a_{2n}x_n & \geq & b_2 \\ & & & \vdots & & \vdots & + & \dots & + & \vdots & & \vdots \\ & & & a_{m1}x_1 & + & a_{m2}x_2 & + & \dots & + & a_{mn}x_n & \geq & b_m \\ & & & x_1 & , & x_2 & , & \dots & , & x_n & \geq & 0 \end{array} \quad (2.1)$$

$\sum_{j=1}^n a_{ij}x_j \geq b_i$  eşitsizliği,  $i$ . kısıtı ifade etmektedir.  $i = 1, \dots, m, j = 1, \dots, n$  için  $a_{ij}$  katsayıları teknolojik katsayılar olarak adlandırılmaktadır. Bu katsayılar  $\mathbf{A}$  matrisinde yer almaktadır.

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

$i$ . bileşenin sütun vektörü olan ve sağ taraf vektörü olarak adlandırılan  $b$  vektörü karşılanacak minimum gereklilikleri sağlamaktadır.  $x_1, x_2, \dots, x_n \geq 0$  kısıtı ise negatif olmama kısıtı olarak adlandırılmaktadır. Tüm kısıtları sağlayan  $x_1, x_2, \dots, x_n$  değişken değerleri seti uygun çözüm olarak adlandırılmaktadır. Bu değerlerin tamamına ise uygun alan ismi verilmektedir. Tüm uygun çözümlerin içinde amaç fonksiyonunu minimize veya maksimize eden karar değişkenlerini bulmak doğrusal programlama problemlerinin amacıdır.

Bir doğrusal programlama problemini sonlu olarak çözebilen algoritmanın hesaplama karmaşıklığını analiz etmek için, bu problemin çözmek için gereken eforun üst sınırının

belirlenmesi gerekmektedir. Bu efor, problemi çözmek için gereken, toplama, çarpma ve karşılaştırmalar gibi temel işlemlerin sayısı dikkate alınarak hesaplanabilir. Problemin boyutları açısından  $g(m, n, L)$  fonksiyonunun belirlenmesi gerekmektedir. Bu fonksiyonda  $L$ , problemin tüm verilerini kaydetmek için gereken ikili bitlerin sayısıdır ve problemin girdi uzunluğu olarak bilinir. Yeterince büyük bir sabit olan  $\tau > 0$  dikkate alınarak, problemi çözmek için algoritmanın gerektirdiği toplam temel işlem sayısı  $\tau g(m, n, L)$  fonksiyonundan daha küçük olmak zorundadır. Böyle bir durumda, algoritmanın karmaşıklık derecesi  $O(g(m, n, L))$  ile hesaplanmaktadır.

$\mathbf{A}$ ,  $m \times n$  boyutlu bir matris ve  $\mathbf{b}$ ,  $m$ -boyutlu bir matris iken,  $\mathbf{Ax} = \mathbf{b}$  ve  $\mathbf{x} \geq \mathbf{0}$  olan bir sistem dikkate alındığında,  $\text{rank}(\mathbf{A}, \mathbf{b}) = \text{rank}(\mathbf{A}) = m$  sıra olduğu varsayılırsa,  $\mathbf{A}$  matrisinin sütunları yeniden düzenlendikten sonra,  $\mathbf{A} = [\mathbf{B}, \mathbf{N}]$  olur. Burada  $\mathbf{B}$ ,  $m \times m$  boyutlu tersi alınabilir bir matris ve  $\mathbf{N}$ ,  $m \times (n - m)$  boyutlu bir matristir.  $\mathbf{Ax} = \mathbf{b}$  denkleminde  $\mathbf{x} = \begin{bmatrix} \mathbf{x}_B \\ \mathbf{x}_N \end{bmatrix}$  olan çözüm, temel uygun çözüm olarak adlandırılır. Bu çözümde;  $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}$  ve  $\mathbf{x}_N = \mathbf{0}$ 'dır. Eğer  $\mathbf{x}_B \geq \mathbf{0}$  ise,  $\mathbf{x}$ , sistemin temel uygun çözümü olur. Burada  $\mathbf{B}$ , temel değişkenler matrisi ve  $\mathbf{N}$ , temel olmayan değişkenler matrisidir.  $\mathbf{x}_B$  elemanları temel değişkenler (veya bağımlı değişkenler) ve  $\mathbf{x}_N$  elemanları temel olmayan değişkenler (veya bağımsız değişkenler) olarak adlandırılmaktadır. Temel uygun çözüm sayısı,  $\binom{n}{m} = \frac{n!}{m!(n-m)!}$  ile hesaplanır.

Amaç fonksiyonu aşağıda  $z_0$  ile belirtilmiş bir doğrusal programlama problemi için temel uygun çözümün  $\begin{pmatrix} \mathbf{B}^{-1}\mathbf{b} \\ \mathbf{0} \end{pmatrix}$  olduğu varsayılır.

$$z_0 = \mathbf{c} \begin{pmatrix} \mathbf{B}^{-1}\mathbf{b} \\ \mathbf{0} \end{pmatrix} = (\mathbf{c}_B, \mathbf{c}_N) \begin{pmatrix} \mathbf{B}^{-1}\mathbf{b} \\ \mathbf{0} \end{pmatrix} = \mathbf{c}_B \mathbf{B}^{-1}\mathbf{b}$$

Bu durumda uygunluk için  $\mathbf{x}_B \geq \mathbf{0}, \mathbf{x}_N \geq \mathbf{0}$  ve  $\mathbf{b} = \mathbf{Ax} = \mathbf{Bx}_B + \mathbf{Nx}_N$  olması gerekmektedir. Bu eşitlik  $\mathbf{B}^{-1}$  ile çarpıldığında;

$$\begin{aligned}
\mathbf{x}_B &= \mathbf{B}^{-1}\mathbf{b} - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_N \\
&= \mathbf{B}^{-1}\mathbf{b} - \sum_{j \in J} \mathbf{B}^{-1}a_j x_j \\
&= \bar{\mathbf{b}} - \sum_{j \in J} (y_j) x_j \quad (2.2)
\end{aligned}$$

temel deęişken seti elde edilir. Bu denklemde  $J$ , temel olmayan deęişkenlerin indislerinin kümesidir. Bu deęişkenlere göre, amaç fonksiyonu ařaęıdaki gibi ifade edilebilir:

$$\begin{aligned}
z &= \mathbf{c}\mathbf{x} \\
&= \mathbf{c}_B\mathbf{x}_B + \mathbf{c}_N\mathbf{x}_N \\
&= \mathbf{c}_B(\mathbf{B}^{-1}\mathbf{b} - \sum_{j \in J} \mathbf{B}^{-1}a_j x_j) + \sum_{j \in J} c_j x_j \\
&= z_0 - \sum_{j \in J} (z_j - c_j) x_j \quad (2.3)
\end{aligned}$$

Bu durumda her temel olmayan deęişken için  $z_j = \mathbf{c}_B\mathbf{B}^{-1}a_j$  ifadesine eřittir. Bu dönüşümler kullanılarak, doğrusal programlama problemi ařaęıdaki gibi yazılabilir:

$$\begin{aligned}
\min \quad z &= z_0 - \sum_{j \in J} (z_j - c_j) x_j \\
\text{kısıtlar} \quad \sum_{j \in J} (y_j) x_j + \mathbf{x}_B &= \bar{\mathbf{b}} \\
x_j &\geq 0, \quad j \in J, \quad \text{ve} \quad \mathbf{x}_B \geq \mathbf{0} \quad (2.4)
\end{aligned}$$

Temel uygun çözüm ve bu çözüme karşılık gelen bir temel deęişken dikkate alındığında,  $x_k$  ile ifade edilen bazı temel olmayan deęişkenler için  $z_k - c_k > 0$  ise geliştirilebilir veya tüm temel olmayan deęişkenler için  $z_j - c_j \leq 0$  ise optimal nokta ile durdurulabilir. Eęer  $z_k - c_k > 0$  ve  $y_k$  vektörü en az bir pozitif bileşen içeriyorsa,  $x_k$  deęişkenindeki artış mevcut temel deęişkenlerden biri tarafından engellenir, deęişken 0'a düşer ve temel deęişken olmaktan çıkar. Diğer taraftan eęer  $z_k - c_k > 0$  ve  $y_k \leq 0$  ise  $x_k$  süresiz olarak artırılabilir ve optimal amaç fonksiyonu deęeri sınırsızdır ( $-\infty$ ). Bu konu tam olarak Simpleks yönteminin özüdür.

Simpleks yönteminde başlangıç adımı olarak  $\mathbf{B}$  temel deęişkenler matrisi ile bir başlangıç temel uygun çözümü seçilir.



Temel adımlar ise aşağıdaki gibidir:

**Adım 1:**  $\mathbf{B}\mathbf{x}_B = \mathbf{b}$  olan sistem çözülür ( $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b} = \bar{\mathbf{b}}$  olan eşsiz çözüm ile).

$\mathbf{x}_B = \bar{\mathbf{b}}$ ,  $\mathbf{x}_N = \mathbf{0}$  ve  $z = \mathbf{c}_B\mathbf{x}_B$  olsun.

**Adım 2:**  $\mathbf{w}\mathbf{B} = \mathbf{c}_B$  olan sistem çözülür ( $\mathbf{w} = \mathbf{c}_B\mathbf{B}^{-1}$  olan eşsiz çözüm ile). ( $\mathbf{w}$  vektörü simpleks çarpanların vektörü olarak adlandırılır, çünkü bileşenleri amaç fonksiyonunu kanonik forma dönüştürmek için eklenen  $\mathbf{A}$  matrisinin satırlarıdır.)

Tüm temel olmayan değişkenler için  $z_j - c_j = \mathbf{w}\mathbf{a}_j - c_j$  hesaplanır.

$$\left| z_k - c_k = \max_{j \in J} \{z_j - c_j\} \right|$$

olsun.  $z_k - c_k \leq 0$  ise, mevcut temel uygun çözüm, optimal çözüm olarak durdurulur. Aksi takdirde, giriş değişkeni olarak  $x_k$  ile Adım 3'e gidilir. (Giriş değişkeni seçimi için kullanılan bu strateji, "Dantzig kuralı" olarak bilinir.)

**Adım 3:**  $\mathbf{B}\mathbf{y}_k = \mathbf{a}_k$  olan sistem çözülür ( $\mathbf{y}_k = \mathbf{B}^{-1}\mathbf{a}_k$  olan eşsiz çözüm ile). Eğer  $y_k \leq 0$  ise, optimal çözümün sınırsız olduğu sonucuna varılır.

$$\left\{ \begin{bmatrix} \bar{\mathbf{b}} \\ \mathbf{0} \end{bmatrix} + x_k \begin{bmatrix} -y_k \\ \mathbf{e}_k \end{bmatrix} : x_k \geq 0 \right\}$$

$\mathbf{e}_k$ ,  $k$ . pozisyondaki 1 hariç  $(n - m)$ 'lik sıfır-vektördür. Eğer  $y_k \not\leq 0$  ise, Adım 4'e gidilir.

**Adım 4:**  $x_k$ , giriş değişkeni olarak seçilir.  $r$ , engelleme değişkeninin indisi olmak üzere,  $x_{B_r}$  değişkeni, aşağıdaki minimum oran testi ile belirlenmektedir:

$$\frac{\bar{b}_r}{y_{rk}} = \min_{1 \leq i \leq m} \left\{ \frac{\bar{b}_i}{y_{ik}} : y_{ik} > 0 \right\}$$

$a_k$ 'nın  $a_{B_r}$  ile yer değiştirdiği noktada  $\mathbf{B}$  temel değişkeni güncellenir,  $J$  indis kümesi güncellenir ve Adım 1 tekrarlanır.

Her bir doğrusal programlama problemiyle ilişkili olan ve dual olarak adlandırılan bir başka doğrusal programlama problemi mevcuttur. Dual doğrusal program orijinal primal doğrusal programa göre bazı avantajlara sahip olabilir. Orijinal programın çözümünü elde etmek için kullanılabilir ve değişkenleri, orijinal doğrusal programın optimal çözüm kümesi hakkında çok yararlı bilgiler sağlamaktadır. Dual teorem, primal ve dual problemin eşit optimal amaç fonksiyonu değerlerine sahip olduğunu belirtmektedir (eğer problemler optimal çözümlere sahipse). Bu teoreme dayanarak, primal problemin  $m$  kısıtlı,  $n$  değişkenli normal maksimizasyon problemi olduğunu varsayarsak, dual problemin  $m$  değişkenli ve  $n$  kısıtlı normal minimizasyon problemi olacağı söylenebilir.

Primal doğrusal programın aşağıdaki gibi kanonik formda verildiği varsayılırsa:

$$\begin{aligned}
 \text{P:} \quad & \min \quad \mathbf{cx} \\
 & \text{kısıtlar} \quad \mathbf{Ax} \geq \mathbf{b} \\
 & \mathbf{x} \geq \mathbf{0} \quad (2.5)
 \end{aligned}$$

Dual doğrusal program aşağıdaki gibi tanımlanır:

$$\begin{aligned}
 \text{D:} \quad & \max \quad \mathbf{wb} \\
 & \text{kısıtlar} \quad \mathbf{wA} \leq \mathbf{c} \\
 & \mathbf{w} \geq \mathbf{0} \quad (2.6)
 \end{aligned}$$

Her bir primal kısıt için bir dual değişken ve her primal değişken için bir dual kısıt mevcuttur.

Eğer primal doğrusal program aşağıdaki gibi standart formda verildiği varsayılırsa:

$$\begin{aligned}
 \text{P:} \quad & \min \quad \mathbf{cx} \\
 & \text{kısıtlar} \quad \mathbf{Ax} = \mathbf{b} \\
 & \mathbf{x} \geq \mathbf{0} \quad (2.7)
 \end{aligned}$$

Dual doğrusal program aşağıdaki gibi tanımlanır:

$$\begin{aligned} D: \quad & \max \quad \mathbf{wb} \\ & \text{kısıtlar} \quad \mathbf{wA} \leq \mathbf{c} \\ & \mathbf{w}, \text{ kısıtlanmamış} \\ & \text{değişken} \end{aligned} \quad (2.8)$$

Bir maksimizasyon problemini çözmek için Simpleks yöntemi kullanıldığında (maksimizasyon problemi primal problem olarak adlandırılır), çözüme ilk uygun çözüm ile başlanır (çünkü başlangıç tablosundaki her kısıtın sağ taraf değeri negatif olmayan değerlerden oluşur). Başlangıç tablosunun amaç fonksiyonu satırındaki en az bir değişken negatif katsayıya sahiptir, bu nedenle başlangıçtaki primal çözüm dual uygun çözüm değildir. Simpleks tablosu aracılığıyla primal uygunluk korunur ve dual uygunluğa ulaşıldığında (amaç fonksiyonu satırında bir negatif olmayan değer olduğunda) optimal bir çözüm elde edilir. Bununla birlikte, 0-sütunundaki her değişkenin negatif olmayan katsayı olduğu (dual uygun tablo) ve en az bir kısıtın sağ taraf değerinin negatif değer aldığı (ilk uygun olmayan) tablo ile doğrusal programlama problemi çözümüne başlamak daha kolaydır. Dual Simpleks metodu, negatif olmayan amaç fonksiyonu satırını sağlar ve sonunda sağ taraf değerlerinin negatif olmadığı bir tablo elde eder. Bu noktada optimal tablo elde edilir. Bu teknik dual uygunluğu sağladığı için Dual Simpleks metodu olarak adlandırılır.

Belirli durumlarda yapay değişkenler eklemeyen uygun bir başlangıç temel çözümü (tüm  $\bar{b}_i \geq 0$  olan) bulmak zordur. Bu durumlarda, uygun olmayan fakat dual uygun olan (bir minimizasyon problemi için tüm  $z_j - c_j \leq 0$  olan) başlangıç temel çözümü bulmak mümkündür. Bu gibi durumlarda dual uygulanabilirlik ve tamamlayıcı gevşekliği sürdüren ve primal uygunluğa doğru yönelen bir dizi Simpleks tablonun geliştirileceği Simpleks yönteminin bir varyantının kullanılması yararlı olacaktır.

Dual Simpleks yönteminde başlangıç adımı olarak tüm  $j$  değerleri için  $z_j - c_j = \mathbf{c}_B \mathbf{B}^{-1} \mathbf{a}_j - c_j \leq 0$  olan primal  $\mathbf{B}$  temel çözümü bulunur.

Temel adımlar ise aşağıdaki gibidir:

**Adım 1:** Eğer  $\bar{\mathbf{b}} = \mathbf{B}^{-1}\mathbf{b} \geq 0$ , ise durulur; mevcut çözüm optimaldir. Aksi takdirde,  $\bar{b}_r < 0$  olan  $r$  pivot satırı seçilir;  $\bar{b}_r = \text{minimum}\{\bar{b}_i\}$ .

**Adım 2:** Eğer tüm  $j$  değerleri için  $y_{rj} \geq 0$  ise, durulur; dual sınırsızdır ve primal uygun değildir. Aksi takdirde, aşağıdaki minimum oran testiyle  $k$  pivot sütunu seçilir:

$$\frac{z_k - c_k}{y_{rk}} = \min_j \left\{ \frac{z_j - c_j}{y_{rj}} : y_{rj} < 0 \right\}$$

**Adım 3:**  $y_{rk}$  değeri için pivot tablo yapılır ve Adım 1'e dönülür.

Dual Simpleks metoduna benzer olarak, dual uygunluk ile başlayan ve tamamlayıcı gevşekliği korurken, primal uygunluğu elde etmeye devam eden bir başka yöntem de Primal-Dual Algoritma yöntemidir. Bununla birlikte, Dual Simpleks metodu ve Primal-Dual metod arasındaki fark, Primal-Dual algoritmanın dual uygun çözümünün temel çözüm olmasının zorunlu olmamasıdır. Verilen bir dual uygun çözüme göre, zor dual kısıtlara karşılık gelen primal değişkenler belirlenir. Simpleks yönteminin ilk aşaması kullanılarak belirlenmiş olan primal değişkenler ile birlikte primal uygunluğa ulaşılmaya çalışılır. Eğer primal uygunluk elde edilemezse, Simpleks yönteminin ilk aşama problemine en az bir yeni değişken alınarak dual uygun çözüm değiştirilir. Primal çözüm uygun hale gelene kadar veya dual çözüm sınırsız hale gelene kadar bu aşama devam ettirilir.

Primal – dual yönteminde başlangıç adımı olarak tüm  $j$  değerleri için  $\mathbf{w}a_j - c_j \leq 0$  olan bir  $\mathbf{w}$  vektörü seçilir.  $Q$ , pozitif olmasına izin verilen primal değişkenlerin indislerinin kümesidir ve  $Q = \{j: \mathbf{w}a_j - c_j = 0\}$  olarak ifade edilir.

Temel adımlar ise aşağıdaki gibidir:

**Adım 1:**  $Q = \{j: \mathbf{w}a_j - c_j = 0\}$  olarak kabul edilir ve aşağıdaki sınırlı primal problem çözülür:

$$\begin{aligned} \min \quad & \sum_{j \in Q} 0x_j + \mathbf{1}\mathbf{x}_a \\ \text{kısıtlar} \quad & \sum_{j \in Q} a_j x_j + \mathbf{x}_a = \mathbf{b} \\ & x_j \geq 0 \quad \forall j \in Q \\ & \mathbf{x}_a \geq \mathbf{0} \end{aligned} \quad (2.9)$$

$x_0$  ile optimal amaç fonksiyonu değeri belirlenir. Eğer  $x_0 = 0$  ise, durulur; optimal çözüm elde edilmiştir. Aksi takdirde;  $\mathbf{v}^*$ , yukarıda belirtilen sınırlı primal problem için optimal dual çözüm olarak ifade edilir.

**Adım 2:** Eğer tüm  $j$  değerleri için  $\mathbf{v}^* a_j \leq 0$  ise, durulur; dual problem sınırsızdır ve primal çözüm de uygun değildir. Aksi takdirde;

$$Q = \min_j \left\{ \frac{-(\mathbf{w}a_j - c_j)}{\mathbf{v}^* a_j} : \mathbf{v}^* a_j > 0 \right\} > 0$$

olarak belirlenir ve  $\mathbf{w}$  değeri,  $\mathbf{w} + Q\mathbf{v}^*$  ile değiştirilir. Adım 1 tekrarlanır (Bazaraa ve ark. 2010).

Doğrusal programlama problemlerinin çözümünde önemli bir yer tutan Simpleks algoritması, 1984 yılında N. Karmarkar tarafından yeni bir polinom – zaman algoritması önermesiyle ciddi bir rakip elde etmiştir. Karmarkar algoritması, uygun alanda belirlenen bir alanı tamamen eniyilemek için izlenen izdüşümsel dönüşümlerin yinelenmesine dayanır. Doğrusal programlama problemlerinin çözümünde Karmarkar yönteminin kullanılması için problemin aşağıdaki biçimde bulunması gerekmektedir:

$$\begin{aligned}
\min \quad & \mathbf{c}\mathbf{x} \\
\text{kısıtlar} \quad & \mathbf{A}\mathbf{x} = \mathbf{0} \\
& \mathbf{1}\mathbf{x} = \mathbf{1} \\
& \mathbf{x} \geq \mathbf{0} \quad (2.10)
\end{aligned}$$

Bu gösterimde  $\mathbf{A}$ ,  $m, n \geq 2$  olan, tüm elemanları tamsayı olan  $m \times n$  boyutlu bir matris,  $\mathbf{c}$ , tüm elemanları tamsayı olan bir matris,  $\mathbf{1}$ ,  $n$  adet 1'den oluşan bir satır vektörüdür ve burada aşağıdaki varsayımlar geçerli olmalıdır:

- Verilen doğrusal programlama problemi için  $x^0 = \left[\frac{1}{n} \frac{1}{n} \dots \frac{1}{n}\right]$  noktası uygun olmalıdır.
- Verilen doğrusal programlama problemi için optimal z-değeri 0'a eşittir.

$\mathbf{x} = [x_1 \ x_2 \ \dots \ x_n]^T$  matrisinde  $\mathbf{K}$ ,  $m \times n$ 'lik bir matristir.  $\mathbf{c} = [c_1 \ c_2 \ \dots \ c_n]$  matrisi ise  $n$ -boyutlu sütun vektörüdür ve 0'a eşittir. Karmarkar algoritmasının adımları aşağıdaki gibidir:

**Adım 1:**  $x^0 = \left[\frac{1}{n} \frac{1}{n} \dots \frac{1}{n}\right]^T$  uygun noktasından başlanır ve  $k = 0$  olarak alınır.

**Adım 2:** Eğer  $cx^k < \epsilon$  ise durulur, değilse 3. adıma gidilir.

**Adım 3:** Aşağıda verilen denklem ile dönüştürülmüş birim simpleks  $y^{k+1} = [y_1^{k+1} \ y_2^{k+1} \ \dots \ y_n^{k+1}]^T$  noktası elde edilir.

$$y^{k+1} = \left[\frac{1}{n} \frac{1}{n} \dots \frac{1}{n}\right]^T - \frac{\theta(I - P^T(PP^T)^{-1}P)[\text{Diag}(x^k)]c^T}{\|c_p\|\sqrt{n(n-1)}}$$

Verilen denklemde  $\|c_p\|$ ,  $(I - P^T(PP^T)^{-1}P)[\text{Diag}(x^k)]c^T$  uzunluğuna eşittir ve  $P$  ise ilk  $m$  satırı  $A[\text{Diag}(x^k)]$  olan son satırı 1 vektörü olan ve  $0 < \theta < 1$  değerleri arasında algoritmanın yakınsamasını sağlamak için seçilen  $(m+1) \times n$ 'lik matristir. Yakınsamayı sağlamak için  $\theta = \frac{1}{4}$  olarak seçilebilir. Ardından orijinal alanda  $y^{k+1}$  noktasına karşılık gelen noktayı belirlemek için merkezleme

dönüşümü kullanılarak yeni bir  $x^{k+1}$  noktası elde edilir.  $x^{k+1} = f^{-1}(y^{k+1}/x^k)$   
Adım 1'e dönülerek  $k$  artırılır ve Adım 2 ile devam edilir.

## 2.2. Tamsayılı Doğrusal Programlama

Doğrusal programlama problemlerinde değişkenlerin tamsayı değerleri olması durumunda, probleme tamsayılı doğrusal programlama problemi adı verilir. Değişkenlerin yalnızca bir kısmının tamsayı değerlerine sahip olması durumunda bu model, karışık tamsayılı programlama problemi olarak adlandırılır. Eğer değişkenlerin tamamı tamsayılardan oluşuyor ise, saf tamsayılı programlama problemi olarak adlandırılır. Yalnızca 0 ve 1 ikili değişkenlerini içeren tamsayılı problemlere ise ikili (veya 0-1 tamsayı) programlama problemi adı verilir.

Karışık tamsayılı programlama problemlerinin genel formülasyonu aşağıdaki gibidir:

$$\begin{aligned} \max \quad & \sum_{j=1}^n c_j x_j \\ \text{kısıtlar} \quad & \sum_{j=1}^n a_{ij} x_j \leq b, \quad i = 1, 2, \dots, m \\ & x_j \geq 0, \quad j = 1, 2, \dots, n \\ & x_j \text{ tamsayı, } j = 1, 2, \dots, l \quad l \leq n \quad (2.11) \end{aligned}$$

Bu formülasyonda  $l = n$  olduğunda, problem saf tamsayılı programlama problemine dönüşmektedir.

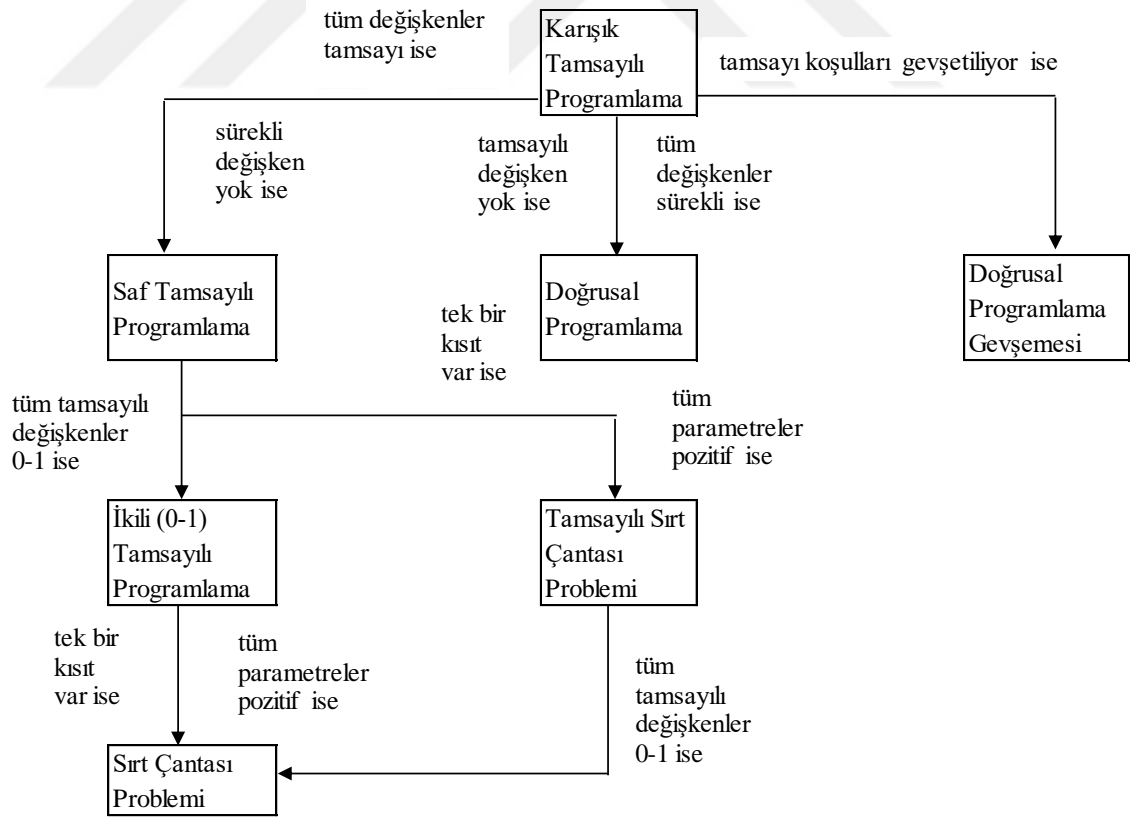
Problem değişkenlerinin 0 ve 1 ile temsil edildiği ikili programlama problemleri için değişkenler aşağıdaki gibi ifade edilir (Hillier ve Lieberman 2001):

$$x_j = \begin{cases} 1, & \text{eğer } j. \text{ karar evet ise} \\ 0, & \text{eğer } j. \text{ karar hayır ise} \end{cases}$$

Karışık tamsayılı programlama modellerinde tüm değişkenler tamsayılı ise ve sürekli değişken bulunmuyorsa, problem saf tamsayılı programlama modeline dönüşmektedir. Eğer tamsayılı değişken bulunmuyorsa ve tüm değişkenler sürekli ise, problem doğrusal

programlama modeline dönüşmektedir. Doğrusal programlama ayrıca, belirli bir karışık tamsayılı programlama probleminin tamsayı koşullarının gevşetilmesi (veya yok sayılması) yoluyla da elde edilir. Bu nedenle elde edilen doğrusal programlama modeline doğrusal programlama gevşemesi (belirli bir tamsayılı programlama modelinin) adı verilir. Doğrusal programlama gevşemesi,  $x$  ve  $y$  değişkenlerinin ikisini de içerir ve  $y$  vektörünü, sürekli değişkenlerin bir vektörü olarak davranır.

Tamsayı değişkenlerinin 0 veya 1 ile kısıtlandığı bir tamsayı programına 0-1 (ikili) tamsayılı programlama modeli adı verilir. Amaç fonksiyonu ve kısıt katsayıları pozitif olan tek bir  $<$  doğrusal kısıt içeren ikili doğrusal programlama problemine sırt çantası (knapsack) problemi denir. Tek bir kısıt ve tüm kısıtların katsayılarının pozitif olduğu, tamsayılı değişken değerlerinin 0-1 ile kısıtlı olmadığı tamsayılı programlama modeline tamsayılı sırt çantası problemi ismi verilir. Belirli koşullar altında çeşitli karışık tamsayılı programlama sınıfları arasındaki ilişki Şekil 2.1’de verilmiştir.



Şekil 2.1. Karışık tamsayılı programlama sınıfları (Chen ve ark. 2010)



Karışık tamsayı programlama problemlerinin çözümünde Dal–Sınır algoritması kullanılabilir. Dal–Sınır algoritmasının uygulanmasında başlangıç olarak  $z^* = -\infty$  olarak belirlenir, burada  $z^*$ ,  $z$  fonksiyonunun güncel değeridir. Aşağıdaki dallanma adımı, sınırlama adımı, budama (fathoming) adımı ve optimallik testi tüm probleme uygulanır.

**Dallanma Adımı:** Budama adımı uygulanmamış alt problemler içinden en son oluşturulan alt problem seçilir. Alt problemin doğrusal programlama gevşemesi için optimal çözümde tamsayı olmayan bir değere sahip tamsayı kısıtlı değişkenler arasında dallanma değişkeni olarak değişkenlerin doğal sıralamasında ilki seçilir. Bu değişken  $x_j$  ve bu değişkenin çözüm değeri  $x_j^*$  olsun. Alt problem için düğümden dallanma ile iki yeni alt problem yaratma amacıyla;  $x_j \leq [x_j^*]$  ve  $x_j \geq [x_j^*] + 1$  kısıtları eklenir.

**Sınırlama Adımı:** Her yeni alt problem için doğrusal programlama gevşemesine Simpleks yöntemi (veya yeniden optimize ederken Dual Simpleks yöntemi) uygulayarak ve elde edilen optimal çözüm için  $Z$  değerini kullanarak sınır elde edilir.

**Budama Adımı:** Her yeni alt problem için, aşağıda verilen üç budama testi uygulanır ve testlerin herhangi biriyle budanan alt problemler çıkarılır.

Test 1: Sınır değeri  $\leq Z^*$  midir? Buradaki  $Z^*$ , mevcut  $Z$  fonksiyonunun güncel değeridir.

Test 2: Doğrusal programlama gevşemesinin uygun bir çözümü yok mudur?

Test 3: Doğrusal programlama gevşemesi için optimal çözüm, tamsayı kısıtlı değişkenler için tamsayı değerlere sahip midir? (Bu çözüm mevcut çözümden daha iyi ise, yeni mevcut çözüm olarak belirlenir ve test 1, daha iyi  $Z^*$  ile tüm budanmamış alt problemlere yeniden uygulanır.)

**Optimallik Testi:** Alt problem kalmadığında durulur, mevcut çözüm optimaldir. Aksi takdirde, başka bir iterasyon ile devam edilir.

### 2.3. Doğrusal Programlama Çözücüleri

Tez çalışması kapsamında farklı çözücüler kullanılarak çeşitli testler gerçekleştirilmiştir. Bu kısımda, öncelikle mevcut çeşitli doğrusal programlama çözücüleri hakkında bilgi verilmiştir. Bu çalışma kapsamında ticari çözücülerden CPLEX ve GUROBI, açık kaynak kodlu çözücülerden ise CLP, LP\_SOLVE ve GLPK kullanılarak testler gerçekleştirilmiştir.

**LINGO:** Lingo, doğrusal, doğrusal olmayan (dışbükey ve dışbükey olmayan / küresel), ikinci dereceden, ikinci dereceden kısıtlı, ikinci dereceden konik, yarı belirli, stokastik ve tamsayılı optimizasyon modellerini daha hızlı, daha kolay ve daha verimli çözmek için tasarlanmış kapsamlı bir araçtır. Lingo, doğrusal, doğrusal olmayan ve tamsayılı problemleri hızla okunabilir bir formda formüle etmeye olanak sağlamaktadır. Doğrudan veritabanından veya elektronik tablolardan bilgi alan modeller oluşturmaya olanak sağlar. Modeller Lingo içinde oluşturulabilir, çözülebilir veya doğrudan yazılan bir uygulamadan çağrılabilir (<https://www.lindo.com/index.php/products/lingo-and-optimization-modeling>).

**MATLAB:** Matlab Optimization Toolbox, doğrusal programlama, karışık tamsayılı doğrusal programlama, ikinci dereceden programlama, doğrusal olmayan programlama, kısıtlı doğrusal en küçük kareler, doğrusal olmayan en küçük kareler ve doğrusal olmayan denklemler için çözücüleri içermektedir. Sürekli ve kesikli problemlere en uygun çözümleri bulmak, trade-off analizi yapmak ve algoritma ve uygulamalara optimizasyon metodlarını dahil etmek için Matlab çözücüleri kullanılabilir. Toolbox ile parametre tahmini, bileşen seçimi ve parametre ayarını içeren optimizasyon görevleri tasarlanabilir. Portföy optimizasyonu, kaynak atama ve üretim planlama, çizelgeleme gibi uygulamalarda en uygun çözümü bulmak için kullanılabilir (<https://www.mathworks.com/products/matlab.html>).

**XPRESS:** Xpress Optimizasyon paketi - sadece Xpress olarak da adlandırılır – karışık tamsayılı doğrusal problemleri çözmek için tasarlanmış ticari tescilli bir yazılımdır. Xpress, bilgisayar platformlarında çok yaygın kullanılmaktadır ve ayrıca birkaç

programlama dili için çağrılabilir bir kütüphane API'si (Application Programming Interface) ve bağımsız bir komut satırı arayüzü de dahil olmak üzere çeşitli arayüzler sunmaktadır (<https://www.fico.com/en/products/fico-xpress-optimization>).

**CPLEX:** Genellikle sadece CPLEX olarak adlandırılan IBM ILOG CPLEX Optimization Studio, büyük ölçekli karışık tamsayılı doğrusal problemleri çözmek için tasarlanmış ticari bir çözücüdür. CPLEX, IBM tarafından geliştirilmiştir. Yazılım ayrıca çeşitli arayüzlere sahiptir, böylece çözücü ile farklı programlama dilleri veya programlar ile bağlantı kurmak mümkündür. Bununla birlikte programın tek başına da çalıştırılabilmesi mümkündür (<https://www.ibm.com/tr-tr/analytics/cplex-optimizer>).

**GUROBI:** GUROBI optimizasyon, karışık tamsayılı matematiksel optimizasyon problemlerinin yanı sıra doğrusal olmayan problemler için de kullanılan modern bir çözücüdür. GUROBI optimizasyon C dilinde yazılmıştır, tüm bilgisayar platformlarında geçerlidir ve çeşitli programlama dillerinden erişilebilir. Standart bağımsız modelleme sistemleri, problemleri tanımlamak ve modellemek için kullanılabilir (<http://www.gurobi.com/>).

**LP\_SOLVE:** lp\_solve, doğrusal (karışık-tamsayılı) programları çözmek için kullanılan ve tamsayılı programları Revize Simpleks ve Dal-Sınır metodu ile çözen ücretsiz ve açık kaynak kodlu programlama çözücüsüdür. Temel olarak lp\_solve, karışık tamsayılı programlama problemlerini çözmek için neredeyse her programlama dilinden çağrılabilen API adı verilen bir dizi kütüphanedir (<http://lpsolve.sourceforge.net/5.5/>).

**GLPK:** GLPK (GNU Doğrusal Programlama Kiti) paketi, büyük ölçekli doğrusal programlama, karışık tamsayılı programlama ve ilişkili diğer problemlerin çözümü için tasarlanmıştır. GLPK paketi; Primal ve Dual Simpleks yöntemleri, Primal ve Dual İç Nokta yöntemi, Dal-Kesim yöntemi, GNU MathProg için çevirmen, API bileşenlerini içeren ücretsiz açık kaynak kodlu programlama çözücüsüdür (<https://www.gnu.org/software/glpk/glpk.html>).

**CLP:** Coin - OR projesi yöneylem araştırması topluluğuna açık kaynak kodlu yazılım oluşturmayı amaçlamaktadır. Coin - OR projesine bağlı olan projelerden bazıları: karışık tamsayı doğrusal programlama problemlerini çözmek için kullanılan Symphony, doğrusal programlama tabanlı Dal–Kesim kütüphanesi olan CBC, doğrusal optimizasyon problemlerini çözmek için kullanılan CLP kütüphanesi şeklinde özetlenebilir (<https://www.coin-or.org/>).

Tez çalışması kapsamında belirlenen LP\_SOLVE, GLPK ve CLP çözücülerinin kaynak kodu kamuya açık olduğu ve genellikle çok iyi belgelendiği için bir yazılımda herhangi bir kısıtlama olmaksızın kullanılmıştır. Böylece farklı platformlarda çözücülerini derlemek mümkün olmuştur. Ayrıca, hemen hemen tüm açık kaynak kodlu programlama kitlerinin herhangi bir yazılım ürünü ve bir doğrusal programlama çözücüsü arasında veri alışverişi için kullanılacak bir çeşit yapılandırılmış API'ye sahip olduğuna da dikkat edilmelidir. Ek olarak bu doğrusal programlama çözücülerini için mevcut uygulama programı arayüzleri R programlama diliyle çağırılacak şekilde kullanılmıştır.

#### **2.4. Kaynak Araştırması**

Literatürde çözücü performanslarını analiz eden geçmiş çalışmalara rastlanmaktadır. Bunlara örnek olarak Bongartz ve ark. (1997) çalışması verilebilir. Büyük ölçekli problemlerin çözümünde LANCELOT ve MINOS çözücülerinin karşılaştırılmasını ele alan bir çalışmada, 913 problem seçilmiştir. Problemlerden doğrusal olanlar Netlib veri setleri kullanılarak, bunun dışındakiler ise CUTE veri seti kullanılarak elde edilmiştir. Toplamda 112 doğrusal problem, 225 kuadratik problem, 300 kısıt problemi (bound – constrained problem), 58 doğrusal kısıtlı problem ve 218 doğrusal olmayan kısıtlı problem çözücülerinin test edilmesi için kullanılmıştır. Çalışmanın sonucunda LANCELOT çözücüsünün kısıtsız ve doğrusal olmayan kısıtlı problemlerde avantajlı olmasının yanı sıra MINOS çözücüsünün doğrusal kısıtlı problemlerde çok iyi performans sergilediği belirtilmiştir. Ayrıca serbestlik derecesinin büyük olduğu durumlarda da LANCELOT çözücüsünün daha iyi sonuç verdiği belirtilmiştir.

Bir başka çalışmada ise Dolan ve ark. (2001) optimizasyon yazılımlarının karşılaştırılması için performans profilleri belirlenmiştir. Performans profilleri,

çözücünün hesaplama süresi ve tüm çözümler içindeki en iyi süreye bağlı olarak hesaplanmıştır. Çözümlerin test edilmesi için COPS 3.0 problem seti (Anonim 2020) ve Mittelman tarafından oluşturulan doğrusal programlama problemleri kullanılmıştır. LANCELOT, MINOS, SNOPT ve LOQO çözümleri karşılaştırılmıştır.

Bir başka çalışma kapsamında ise Meindl ve Templ (2012) çözümlerin karışık tamsayılı doğrusal programlama problemleri üzerindeki performansları karşılaştırılmıştır. Bu çalışmada CBC (branch - and - cut tabanlı doğrusal programlama çözümlüsü), CPLEX (<https://www.ibm.com/tr-tr/analytics/cplex-optimizer>), GLPK (<https://www.gnu.org/software/glpk/>), GUROBI (<http://www.gurobi.com/>), LP\_SOLVE (<http://lpsolve.sourceforge.net/5.5/>), SCIP (C - L - S) (<https://scip.zib.de/>) ve XPRESS (<https://www.fico.com/en/products/fico-xpress-optimization>) çözümleri test edilmiştir. Bu çalışmada kullanılan problemler de Mixed Integer Linear Programming kütüphanesinden alınmıştır. Bu kütüphanede bulunan problem örnekleri, karmaşıklık seviyesi (complexity) farklı olan ve farklı çözümleri karşılaştırmak için kullanılacak karışık tamsayılı doğrusal problemleri içermektedir. Çalışmada kullanılan veri seti, iki saat içinde en az bir çözümlü tarafından optimal çözüme ulaşabilen 87 problemden oluşmaktadır. Sonuç olarak tüm çözümler arasında CPLEX'ten sonra en iyi çözümlünün GUROBI olduğu ve açık kaynaklı çözümlerin ticari çözümlere oranla çok daha yavaş çözüme ulaşmalarına rağmen GLPK – CLP ve lp\_solve olarak sıralandığı belirtilmiştir.

Gearhart J. ve ark. (2013) tarafından yapılan bir başka çalışmada ise, açık kaynaklı doğrusal programlama çözümleri karşılaştırması incelenmiştir. Çalışmada öncelikle potansiyel açık kaynaklı çözümleri belirlemek için doğrusal programlama araçları araştırması yapılmıştır. Araştırma sonucu COIN-OR Doğrusal Programlama (CLP), GNU Doğrusal Programlama Kiti (GLPK), lp\_solve ve MINOS çözümlerinin test edilmesi karşılaştırılmıştır. Bu çözümlerin test edilmesi için kullanılan toplam 201 problem, üç farklı veri kaynağından elde edilmiştir. Netlib veri kaynağının LP kütüphanesinden 138 adet problem, çalışmanın desteklendiği CCO (Contingency Contractor Optimisation) projesi dahilinde bulunan 24 problem ve Hans Mittelman tarafından oluşturulan veri setinden (Anonim, 2019) 39 doğrusal programlama problemi, çözümler tarafından test edilmek üzere seçilmiştir. Problemler CPLEX çözümlünde çözüm

sürelerine göre zor ve kolay olarak gruplanmıştır. 180 problem kolay, 21 problem zor olarak belirlenmiştir. Sonunda bir endüstri standardı olan CPLEX ile karşılaştırılmıştır. Çalışma sonucunda hiçbir açık kaynaklı çözücünün CPLEX'ten daha iyi performans göstermediği ve çözücüleri içinden CLP'nin hız ve yetenek açısından en iyi performansı gösteren çözücü olduğu bulunmuştur.

Kronqvist J. ve ark. (2018) tarafından yapılan bir başka çalışmada ise, konveks karışık tamsayılı doğrusal olmayan programlama (MINLP – mixed integer nonlinear programming) problemlerini çözmek için deterministik yazılımların incelenmesi ve yaygın olarak kullanılan çözücülerin karşılaştırması incelenmiştir. Bu çalışmada test seti olarak MINLPLib kütüphanesindeki konveks olarak sınıflandırılan 335 problemin tümü kullanılmıştır. Çözücüler arasındaki farkların daha iyi vurgulanması için konveks MINLP problemlerinin çözümünde kullanılan en yaygın yöntemlerin özeti verilmiştir. Çözücülerin farklı özelliklere sahip problemlerde nasıl performans gösterdiğini incelemek için test seti sürekli gevşeme farkına, doğrusal olmama derecesine ve göreceli bağımsız değişken sayısına bağlı alt kümelerle ayrılmıştır. Çalışma sonuçları, belirli bir çözücü veya yöntemin belirli bir MINLP problem tipi için ne kadar uygun olduğuna dair bilgiler sunmaktadır.

Literatür çalışması kapsamında incelenen makalelerde kullanılan çözücüler ve çözücüler ile ilgili temel bilgiler (erişilebilir olup olmadıkları, lisans gerekip gerekmediği, hangi programlama dilinin kullanıldığı bilgileri) Çizelge 2.1'de yer almaktadır.

**Çizelge 2.1.** Literatür çalışma özetleri

No	Çalışmanın İsmi	Kullanılan yazılım / solver	Erişim	Gerekli Lisanslar	Kullanılan Programlama Dili
1	Comparison of Open-Source Linear Programming Solvers	-COIN-OR Linear Programming (CLP)	Var	Eclipse Public License Version 1.0	C++
		-GNU Linear Programming Kit (GLPK)	Var	GNU General Public License	C
		-Lp_solve	Var	GNU Lesser General Public License	C
		-MINOS	Yok	Ücretli bir lisans gerekli	Fortran 77
2	Analysis of Commercial and Free and Open Source Solvers for Linear Optimization Problems	-GNU Linear Programming Kit (GLPK)	Var	GNU General Public License	C
		-Lp_solve	Var	GNU Lesser General Public License	C
		-COIN-OR Linear Programming (CLP)	Var	Eclipse Public License Version 1.0	C++
		-CPLEX	Var	Akademik lisans gerekli	Kendi dilinde yazılmıştır
		-GUROBI	Var	Akademik lisans gerekli	C
3	Benchmarking Optimization Software with Performance Profiles	-LANCELOT	Var	Akademik lisans gerekli	AMPL
		-MINOS	Yok	Ücretli bir lisans gerekli	AMPL
		-SNOPT	Var	Akademik lisans gerekli	AMPL
		-LOQO	Var	Ücretli bir lisans gerekli	AMPL

**Çizelge 2.1.** Literatür çalışma özetleri (devam)

No	Çalışmanın İsmi	Kullanılan yazılım / solver	Erişim	Gerekli Lisanslar	Kullanılan Programlama Dili
4	A Review and Comparison of Solvers for Convex MINLP	-AlphaECP (Alpha Extended Cutting Plane)	Var	Ücretli bir lisans gerekli	GAMS
		-ANTIGONE (Algorithms for Continuous Integer Global Optimization of Nonlinear Equations)	Var	Ücretli bir lisans gerekli	GAMS
		-AOA (AIMMS Outer Approximation)	Var	Ücretli bir lisans gerekli	AIMMS
		-BARON (Branch and Reduce Optimization Navigator)	Var	Ücretli bir lisans gerekli	GAMS
		-BONMIN (Basic Open-source Nonlinear Mixed Integer Programming)	Var	Eclipse Public License Version 1.0	GAMS
		-Couenne (Convex Over and Under Envelopes for Nonlinear Estimation)	Var	Eclipse Public License Version 1.0	GAMS
		-DICOPT (Discrete Continuous Optimizer)	Var	Ücretli bir lisans gerekli	GAMS
		-Juniper	Var	Açık kaynak kodlu	JuMP
		-Knitro	Var	Ücretli bir lisans gerekli	GAMS
		-LINDO	Var	Ücretli bir lisans gerekli	GAMS
		-Minotaur (Mixed-Integer Nonlinear Optimization Toolkit: Algorithms, Underestimators and Relaxations)	Var	Açık kaynak kodlu	Kendi dilinde yazılmıştır
		-Muriqui	Var	Açık kaynak kodlu	Kendi dilinde yazılmıştır
		-Pavito	Var	Açık kaynak kodlu	JuMP
		-SBB (Simple Branch and Bound)	Var	Ücretli bir lisans gerekli	GAMS
		-SCIP (Solving Constraint Integer Programs)	Var	Akademik lisans gerekli	GAMS
-SHOT (Supporting Hyperplane Optimization Toolkit)	Var	Eclipse Public License Version 2.0	GAMS		



## 2.5. Doğrusal Programlama Çözücüleri için Kullanılan Platformlar

Tez çalışması kapsamında kullanılan açık kaynak kodlu çözücüler için R-Studio platformu, ticari çözücüler için ise Maximal Software Mathematical Programming Language (MPL), OptiMax kütüphanesi kullanılmıştır.

RStudio, doğrudan kod yürütmeyi destekleyen bir konsol, sözdizimi vurgulama düzenleyicisi ve çizim, geçmiş, hata ayıklama ve çalışma alanı yönetimi araçları ile R için entegre bir geliştirme ortamıdır (<https://rstudio.com/>). RStudio ortamında GLPK, LP\_SOLVE ve CLP çözücülerini kullanmak için ROI (R Optimization Infrastructure) kütüphanesine bağlı ilgili çözücülerin ve çözümünü test edilecek Netlib datasetinin kütüphaneleri kullanılmıştır.

MPL (Matematiksel Programlama Dili), model geliştiricinin karmaşık optimizasyon modellerini açık, özlü ve verimli bir şekilde formüle etmesini sağlayan gelişmiş bir modelleme sistemidir. MPL'de geliştirilen modeller, bugün piyasada bulunan çok sayıda ticari optimize ediciden herhangi biri ile çözülebilir (<http://www.maximalsoftware.com/mpl/>).

MPL OptiMax kütüphanesi, optimizasyon modellerini son kullanıcı uygulamalarına entegre etmek için özel olarak tasarlanmış, nesne yönelimli bir bileşen kütüphanesidir. OptiMax tasarımı Microsoft Activex / Automation bileşen yazılım teknolojilerine dayanmaktadır. OptiMax, MPL modellerini Excel / Access için VBA, Visual Basic, Visual C++, Delphi, Java ve Web için standart kodlama dilleri gibi çeşitli farklı programlama platformlarına sorunsuz bir şekilde entegre etmek için kullanılabilir.

### 3. MATERYAL VE YÖNTEM

#### 3.1. Test Problemleri

Çalışmanın bu bölümünde problemlerin bazı (veya tümü) değişkenlerinin tamsayı (ikili tamsayı) olduğu karışık tamsayılı matematiksel problemler için sağlanan sonuçlar analiz edilecektir. Netlib (<https://www.netlib.org/>, 2020) veri seti, karışık tamsayılı problemler için veri setleri sunmaktadır. Test senaryoları LP kütüphanesinden alınmıştır. Bu kütüphanede bulunan problem örnekleri karmaşıklık bakımından farklılık gösteren ve farklı çözümleri karşılaştırmak için kullanılabilen bir dizi gerçek dünya karışık tamsayılı doğrusal problemleri içermektedir.

Tez çalışması kapsamında Netlib veri seti kullanılarak farklı çözümlerin performans değerlendirilmesi yapılması amaçlanmıştır. Netlib veri setinin LP kütüphanesindeki 93 adet problem dikkate alınmıştır.

Çizelge 3.1’de Netlib veri setinin LP kütüphanesinden seçilmiş olan problemlerin kısıt sayısı, değişken sayısı ve sıfır olmayanların sayısı yer almaktadır. Bu veriler doğrultusunda problemlerin kısıt ve değişken sayılarının grafik üzerinde gösterimi Şekil 3.1’de yer almaktadır. Problemlerden kısıt sayısı en fazla olan problem, 8388 kısıt ve 8426 değişken bulunan “*stocfor3*” problemidir. Değişken sayısı en fazla olan problem ise 3000 kısıt ve 13525 değişken bulunan “*fit2p*” problemidir. Kısıt sayısı ve değişken sayısı en az olan problem, 2 kısıt ve 8 değişkene sahip “*seba*” isimli problemidir.

**Çizelge 3.1.** Netlib veri setindeki problemlerin temel özellikleri

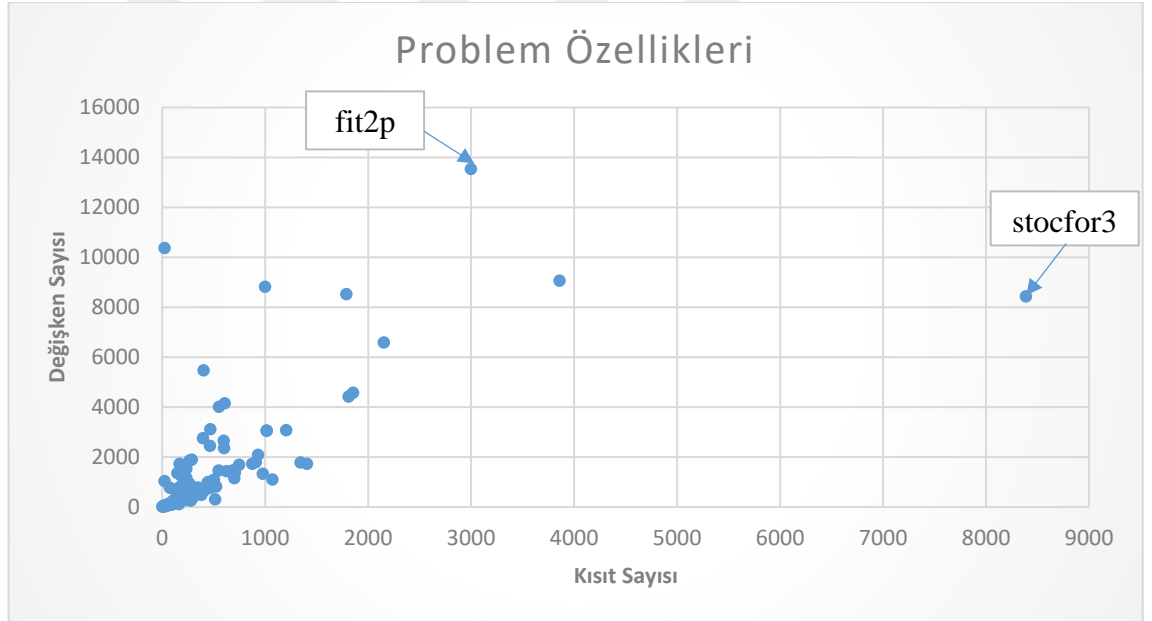
Problem İsmi	Kısıt Sayısı	Değişken Sayısı	SFR OLM	Problem İsmi	Kısıt Sayısı	Değişken Sayısı	SFR OLM
adlittle	53	94	372	pilot.we	602	2346	8234
afiro	11	14	36	pilot4	348	770	4603
agg	164	107	867	pilot87	1811	4416	70189
agg2	280	250	2267	pilotnov	748	1686	11390
agg3	516	302	4300	recipe	48	71	359
bandm	173	223	1485	sc105	32	31	212
beaconfd	16	39	86	sc205	62	61	503
blend	51	58	396	sc50a	17	16	72
bnl1	446	990	4613	sc50b	15	15	56
bnl2	932	2083	10177	scagr25	240	391	1223
boeing1	287	418	2764	scagr7	60	103	305
boeing2	122	160	811	scfxm1	233	379	2147
bore3d	40	60	333	scfxm2	467	759	4317
brandy	104	173	1641	scfxm3	701	1139	6487
capri	142	203	1241	scorpion	57	83	327
cycle	909	1784	12791	scrs8	174	799	2542
czprob	464	2433	4866	scsd1	77	760	2388
d2q06c	1855	4561	30506	scsd6	147	1350	4316
d6cube	402	5467	34332	scsd8	397	2750	8584
degen2	382	473	3851	sctap1	269	339	1444
degen3	1406	1721	24422	sctap2	977	1326	5717
df1001	3861	9052	31400	sctap3	1344	1767	7630
e226	146	249	2261	seba	2	8	11
etamacro	290	474	1913	share1b	93	194	1028
fffff800	292	636	4792	share2b	93	79	691
finnis	296	363	1361	shell	236	1157	2320
fit1d	24	1024	13386	ship04l	288	1886	4267
fit1p	627	1427	9618	ship04s	188	1238	2804
fit2d	25	10364	127769	ship08l	470	3099	7100
fit2p	3000	13525	50284	ship08s	234	1526	3508
ganges	409	647	3123	ship12l	609	4147	9222
gfrd.pnc	277	742	1580	ship12s	267	1847	4121
greenbea	1015	3048	22970	sierra	877	1723	6202
greenbeb	1015	3039	22870	stair	241	269	3518
grow15	300	645	5620	standata	164	331	757
grow22	440	946	8252	standmps	266	925	2167
grow7	140	301	2612	stocfor1	44	59	313

SFR OLM: Sıfır Olmayanlar

**Çizelge 3.1.** Netlib veri setindeki problemlerin temel özellikleri (devam)

Problem İsmi	Kısıt Sayısı	Değişken Sayısı	SFR OLM	Problem İsmi	Kısıt Sayısı	Değişken Sayısı	SFR OLM
israel	163	141	2256	stocfor2	1070	1088	5994
kb2	37	30	260	stocfor3	8388	8426	46700
lotfi	117	282	596	truss	1000	8806	27836
maros.r7	2152	6578	80167	tuff	142	388	4047
maros	524	809	5717	vtp.base	40	61	177
modszk1	550	1455	2936	wood1p	170	1716	44572
nesm	598	2645	12869	woodw	551	4006	14468
perold	503	1074	5346	25fv47	682	1446	9895
pilot.ja	708	1369	10840	80bau3b	1789	8510	18645
pilot	1204	3066	40102				

SFR OLM: Sıfır Olmayanlar



**Şekil 3.1.** Netlib problemlerinin kısıt ve değişken sayıları grafiği

### 3.2. Seçilen Çözücülerin Test Edilmesi

Çalışmanın bu bölümünde amacı, ticari ve ticari olmayan (açık kaynak kodlu) çözücülerle ilgili kıyaslama karşılaştırmalarının sonuçları özetlenmiştir.

Tüm problemler, Windows 10 Home Single Language 64-bit (10.0, Build 18362) (18362.19h1\_release.190318-1202) işletim sistemine sahip, Intel® Core™ i5-7200U CPU @ 2.50GHz (4 CPUs), ~2.7GHz işlemcili, 8192MB RAM'e sahip bilgisayar ile çözdürülmüştür.

Çalışma kapsamında kullanılan iki ticari ve üç ticari olmayan çözücü için aşağıdaki versiyonlar kullanılmıştır:

- CPLEX (12.6.2.0)
- GUROBI (7.5.2)
- CLP (Coin – OR) (0.4)
- GLPK (0.3-0)
- LP\_SOLVE (0.3-2)

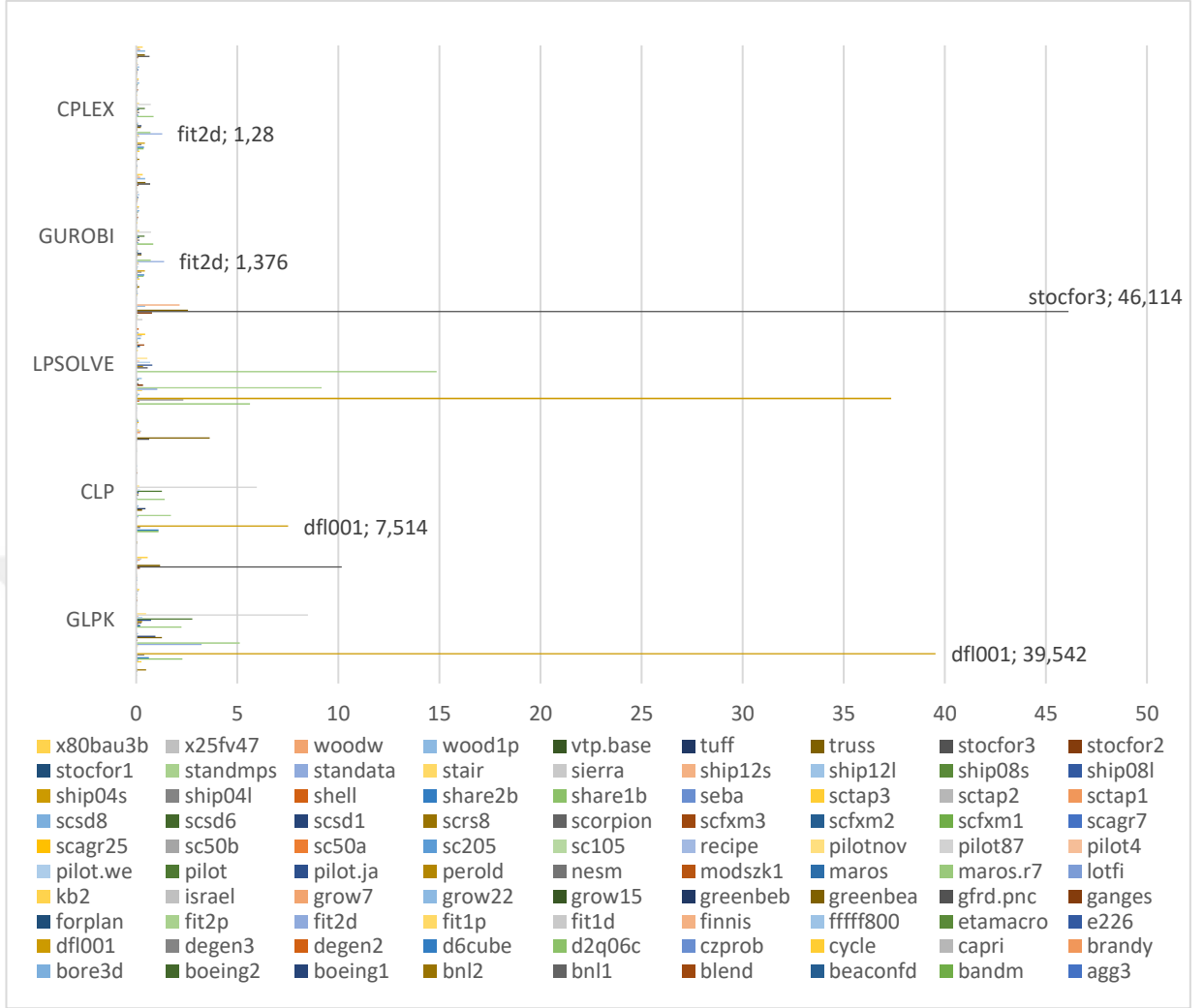
#### 4. BULGULAR ve TARTIŞMA

Tüm test problemlerinin içinde Lp\_solve çözücüsü için optimal çözüme ulaşmayan toplam 28 problem mevcuttur. Bu problemlerin isimleri Çizelge 4.1’de verilmiştir.

**Çizelge 4.1.** lp\_solve çözücüsü ile optimal çözüme ulaşmayan problemler

Problem İsimleri	
bnl1	sc205
bnl2	sc50a
boeing1	sc50b
boeing2	ship04l
brandy	ship04s
cycle	ship08l
czprob	ship08s
d6cube	ship12l
forplan	ship12s
greenbea	tuff
greenbeb	pilot
maros	pilot87
modszk1	x25fv47
sc105	x80bau3b

Tez çalışması kapsamında öncelikle tüm problemler 5 çözücü için 1’er kez çözdürülmüştür. Saniye cinsinden çözüm sonuçları Şekil 4.1’de verilmiştir. Bu verilere göre tüm problemlerin 5 farklı çözücü tarafından birer kez çözdürülmesi sırasında en uzun sürede optimal çözüme ulaşan problemler Çizelge 4.2’de mevcuttur.



Şekil 4.1. Netlib problemlerinin tüm çözümler için çözüm süreleri grafiği

Çizelge 4.2. Tüm çözümler için en uzun sürede çözülen problem isimleri

Çözücü ismi	Problem İsmi	Çözüm Süreleri (sn)
GLPK	df1001	39,542
CLP	df1001	7,514
LPSOLVE	stocfor3	46,114
GUROBI	fit2d	1,376
CPLEX	fit2d	1,280

Tez çalışması kapsamında kullanılan tüm çözümler için problem setindeki problemlerin çözüm sürelerine göre en uzun, en kısa sürede elde edilen çözüm süreleri, bu en uzun ve en kısa çözümler arasındaki süreler, ortalama süreler ve sürelerin standart sapmaları ile ilgili detaylı bilgiler Çizelge 4.3'te yer almaktadır. Bu veriler hesaplanırken, tüm

problemlerin tek çözüm sonuçları dikkate alınarak standart sapmalar hesaplanmıştır. Bu verilere göre standart sapması en yüksek olan çözücü Lp\_solve iken, en düşük olan çözücünün CPLEX çözücüsü olduğu tespit edilmiştir.

**Çizelge 4.3.** Tüm çözücüler için çözüm süreleri istatistikleri

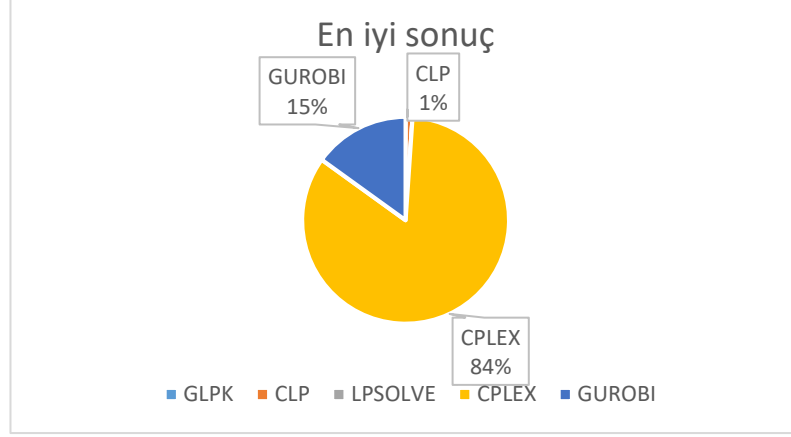
Çözücü ismi	En uzun süre	En kısa süre	Süre farkı	Ortalama	Standart Sapma
GLPK	39,542	0,007	39,535	0,901	4,306
CLP	7,514	0,008	7,507	0,300	1,071
LPSOLVE	46,114	0,000	46,114	1,389	6,306
GUROBI	1,376	0,013	1,363	0,143	0,211
CPLEX	1,280	0,000	1,280	0,136	0,204

Tüm çözücülerin 1'er kez çözüm denemelerinin en iyi sonucu ve en kötü sonucu sağladığı çözücülerin listesi Çizelge 4.4'te yer almaktadır. Bu çizelgeye göre, tüm problemlerin 78 adedi en iyi sonucu CPLEX çözücüsü ile, 14 adedi GUROBI çözücüsü ile ve 1 adedi CLP çözücüsü ile elde edilmiştir. Tüm problemlerin 64 adedi ise en kötü sonucu LPSOLVE çözücüsü ile ve 29 adedi GLPK çözücüsü ile elde edilmiştir. Sonuçlar ile ilgili grafikler Şekil 4.2 ve Şekil 4.3'te verilmiştir.

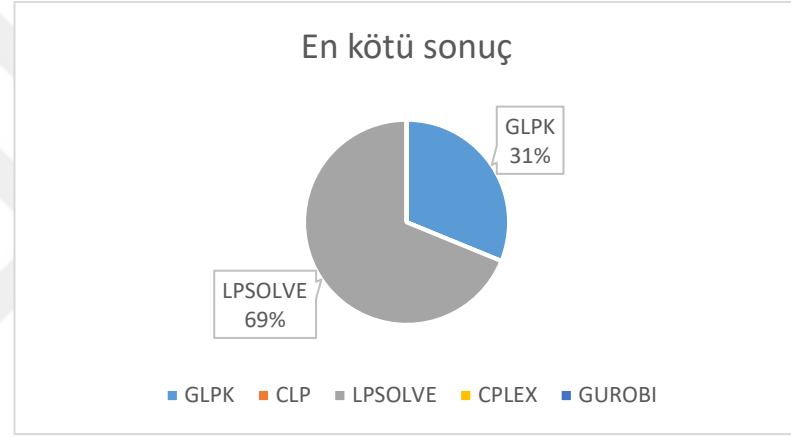
**Çizelge 4.4.** Tüm problemlerin en iyi ve en kötü sonucu elde ettiği çözücüler

Çözücü İsmi	En iyi sonuç	En kötü sonuç
GLPK	0	29
CLP	1	0
LPSOLVE	0	64
CPLEX	78	0
GUROBI	14	0





Şekil 4.2. Tüm problemlerin en iyi sonuçları elde ettiği çözümler



Şekil 4.3. Tüm problemlerin en kötü sonuçları elde ettiği çözümler

Netlib verisetine ait test edilmesi belirlenmiş problemler, Visual Basic for Applications platformunda OptiMax kütüphanesi kullanılarak MPL ortamında CPLEX ve GUROBI çözümleri ile 100'er kez çözdürülmüştür. CPLEX çözümleri için çözüm sürelerinin ortalamalarının bulunduğu grafik Ek-1'de, GUROBI çözümleri için çözüm sürelerinin ortalamalarının bulunduğu grafik Ek-2'de verilmiştir.

Ticari çözümler ile çözümler elde edildikten sonra, problemler RStudio platformunda ROI (R Optimization Infrastructure) kütüphanesi kullanılarak CLP, GLPK ve Lp\_solve çözümleri ile 100'er kez çözdürülmüştür. CLP çözümleri için çözüm sürelerinin ortalamalarının bulunduğu grafik Ek-3'te, GLPK çözümleri için çözüm sürelerinin ortalamalarının bulunduğu grafik Ek-4'te ve Lp\_solve çözümleri için çözüm sürelerinin ortalamalarının bulunduğu grafik Ek-5'te verilmiştir.

100'er tekrarlı çözüm sonuçlarında ticari çözümler olan CPLEX ve GUROBI çözümleriyle en uzun sürede optimal çözüme ulaşan problemler sırasıyla, 1,17 sn ile "fit2d" ve 1,30 sn ile "fit2d" problemidir. CLP, GLPK ve lp\_solve açık kaynak kodlu çözümleriyle en uzun sürede optimal çözüme ulaşan problemler ise sırasıyla şu şekildedir; 7,62 sn ile "df1001", 38,96 sn ile "df1001" ve 44,63 sn ile "stocfor3" problemleridir.

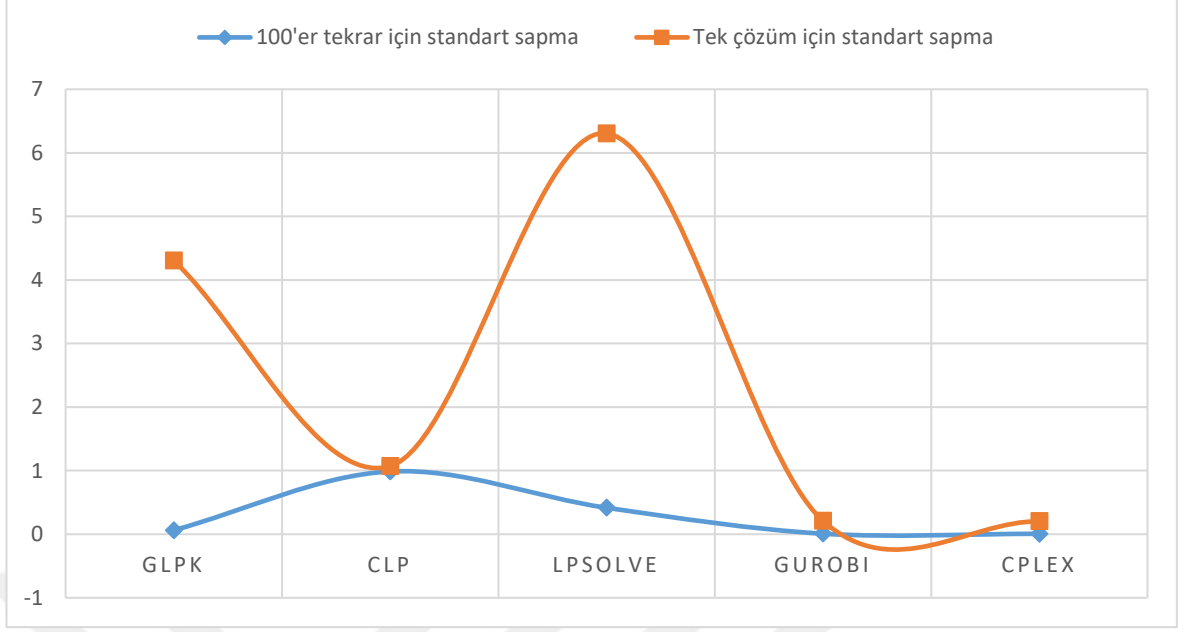
Tüm çözümlerin 100'er kez çözdürülmesi sonucu çözümlerin karşılaştırılması amacıyla istatistik değerleri Çizelge 4.5'te verilmiştir. Burada tüm problemler için en kısa ve en uzun çözüm süreleri, bu süreler arasındaki farklar, sürelerin ortalamaları ve standart sapmaları detaylı olarak verilmiştir.

**Çizelge 4.5.** Tüm çözümler için tekrarlı çözüm süreleri istatistikleri

Çözümlü ismi	En uzun süre	En kısa süre	Süre farkı	Ortalama	Standart Sapma
GLPK	41,368	0,006	41,362	0,894	0,063
CLP	95,325	0,007	95,318	0,329	0,987
LPSOLVE	53,940	0,000	53,940	1,413	0,416
GUROBI	1,372	0,000	1,372	0,130	0,008
CPLEX	1,283	0,000	1,283	0,118	0,007

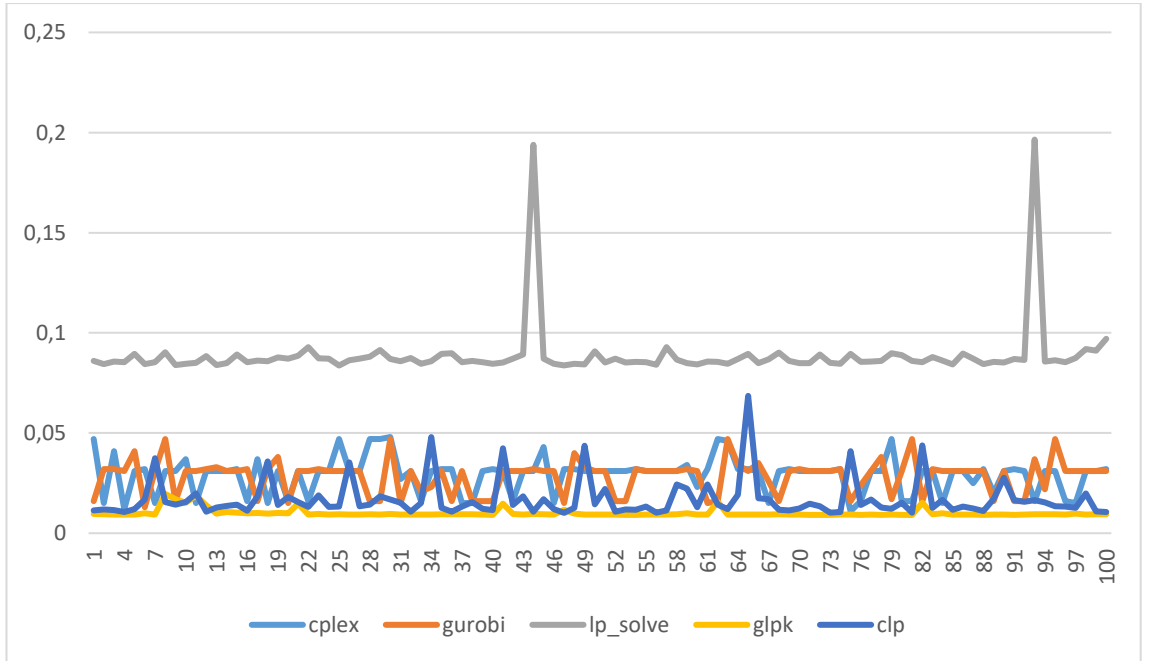
Çalışma sonucunda elde edilen verilere göre, birer kez çözüm testi sonucunda standart sapma değerleri küçükten büyüğe doğru sırasıyla şu şekildedir: CPLEX – GUROBI – CLP – GLPK – LP\_SOLVE. Tüm problemlerin 100'er kez çözüm testi sonucunda standart sapma değerleri küçükten büyüğe doğru sırasıyla şu şekildedir: CPLEX – GUROBI – GLPK – LP\_SOLVE – CLP.

Tek çözüm ve tekrarlı çözümler sonunda elde edilen standart sapma değerleri Şekil 4.4'te sunulmuştur. Bu değerlere göre lp\_solve ve GLPK çözümleri için tekrarlı çözümler standart sapmanın büyük ölçüde azalmasını sağlamıştır. CLP, GUROBI ve CPLEX çözümleri için standart sapma değerleri tek çözüm ve tekrarlı çözümler için çok değişiklik göstermemektedir.



Şekil 4.4. Tüm çözücülerin standart sapma değerleri

Netlib veri setinde bulunan “agg2” probleminin tüm çözücüler ile elde edilen çözüm süreleri saniye cinsinden Şekil 4.5’te verilmiştir. Buradan da anlaşılacağı gibi LPSOLVE çözücüsü dışındaki tüm çözücülerde yakın sonuçlar elde edilmesine rağmen, LPSOLVE çözücüsü için çözüm sonuçları tüm denemelerde daha yüksek çıkmıştır.



Şekil 4.5. “agg2” probleminin çözüm sonuçları

## 5. SONUÇ

Tez çalışması kapsamında literatürde geçerli olan Netlib datasetinin doğrusal programlama kütüphanesine ait 93 adet problem, belirlenmiş olan çözümlerin çözüm süreleri açısından performanslarının değerlendirilmesi amacıyla test edilmiştir. Belirlenmiş matematiksel programlama problemi çözümlerinden ticari olan çözümler CPLEX ve GUROBI iken, açık kaynak kodlu çözümler, GLPK, CLP ve Lp\_solve çözümleridir. Öncelikle tüm test problemlerinin birer kez çözdürülmesi, ardından 100'er kez çözdürülmesi ile istatistiksel veriler elde edilmiştir.

Çalışma kapsamında lp\_solve çözümlüsü dışındaki tüm çözümler tüm problemler için optimal çözümlü elde etmiştir. Elde edilen optimal çözüm değerleri, Netlib datasetlerinin literatürde geçerli olan optimal çözüm değerlerine eşittir.

Literatürde yapılan çalışmalara paralel olarak ticari çözümlerinin, çözüm süresi performansı açısından açık kaynak kodlu çözümlerden daha iyi sonuçlar elde ettiği tespit edilmiştir. Tek çözüm ve tekrarlı çözüm testlerinin her iki test için de en iyi çözüm sonuçlarının sağlandığı çözümlü CPLEX'dir. Hem ticari hem de açık kaynak kodlu çözümler arasında sapması en az olan çözümlüdür. Tez çalışması kapsamında yapılan performans analizinin, konuyla ilgili çalışacak olan kişilere fikir vermesi amaçlanmıştır.

## KAYNAKLAR

- Anonim, 2012a.** GLPK (GNU Linear Programming Kit). <https://www.gnu.org/software/glpk/glpk.html> - (Erişim tarihi: 12.01.2020)
- Anonim, 2016a.** COIN – OR (CLP). <https://www.coin-or.org/> - (Erişim tarihi: 12.01.2020)
- Anonim, 2016b.** MPL Maximal Software. <http://www.maximalsoftware.com/mpl/> - (Erişim tarihi: 13.01.2020)
- Anonim, 2019a.** Benchmarks for Optimization Software. <http://plato.asu.edu/bench.html> - (Erişim tarihi: 13.01.2020)
- Anonim, 2020a.** CPLEX Optimizer. <https://www.ibm.com/tr-tr/analytics/cplex-optimizer> - (Erişim tarihi: 12.01.2020)
- Anonim, 2020b.** COPS: Large Scale Optimization Problem. <https://www.mcs.anl.gov/~more/cops/> - (Erişim tarihi: 12.01.2020)
- Anonim, 2020c.** GUROBI Optimizer. <https://www.gurobi.com/> - (Erişim tarihi: 12.01.2020)
- Anonim, 2020d.** LINGO 18.0. <https://www.lindo.com/index.php/products/lingo-and-optimization-modeling> - (Erişim tarihi: 12.01.2020)
- Anonim, 2020e.** lp\_solve 5.5.2.5. <http://lpsolve.sourceforge.net/5.5/> - (Erişim tarihi: 12.01.2020)
- Anonim, 2020f.** MATLAB. <https://www.mathworks.com/products/matlab.html> - (Erişim tarihi: 12.01.2020)
- Anonim, 2020g.** NETLIB <https://www.netlib.org/> - (Erişim tarihi: 28.01.2020)
- Anonim, 2020.** RStudio. <https://rstudio.com/> - (Erişim tarihi: 13.01.2020)
- Anonim, 2020h.** XPRESS Optimization. <https://www.fico.com/en/products/fico-xpress-optimization> - (Erişim tarihi: 12.01.2020)
- Bazaraa, M.S., Jarvis, J.J, Sherali, H.D. 2010.** Linear Programming and Network Flows. John Wiley & Sons, Inc., Canada, 748 pp.
- Bongartz, I., Conn, A.R., Gould, N.I.M., Saunders, M.A., Toint, L. 1997.** A Numerical Comparison Between the LANCELOT and MINOS Packages for Large-Scale Constrained Optimization. Council for the Central Laboratory of the Research Councils.
- Chen, D.S, Batson, R.G., Dang, Y. 2010.** Applied Integer Programming. A John Wiley & Sons, Inc. New Jersey, 468 pp.

**Dolan, E.D, More, J.J., Munson, T.S. 2004.** Benchmarking Optimization Software with COPS 3.0. Mathematics and Computer Science.

**Gearhart, J.L., Adair, K.L., Detry, R.J., Durfee, J.D., Jones, K.A., Martin, N. 2013.** Comparison of Open-Source Linear Programming Solvers.

**Hillier, F.S., Lieberman G.J. 2001.** Introduction to Operations Research. McGraw-Hill Companies Inc., New York, 1213 pp.

**Kronqvist, J., Bernal, D.E., Lundell, A., Grossmann, I.E. 2018.** A Review and Comparison of Solvers for Convex MINLP. Optimization and Engineering.

**Meindl, B., Templ, M. 2012.** Analysis of Commercial and Free and Open Source Solvers for Linear Optimization Problems. Vienna University of Technology.

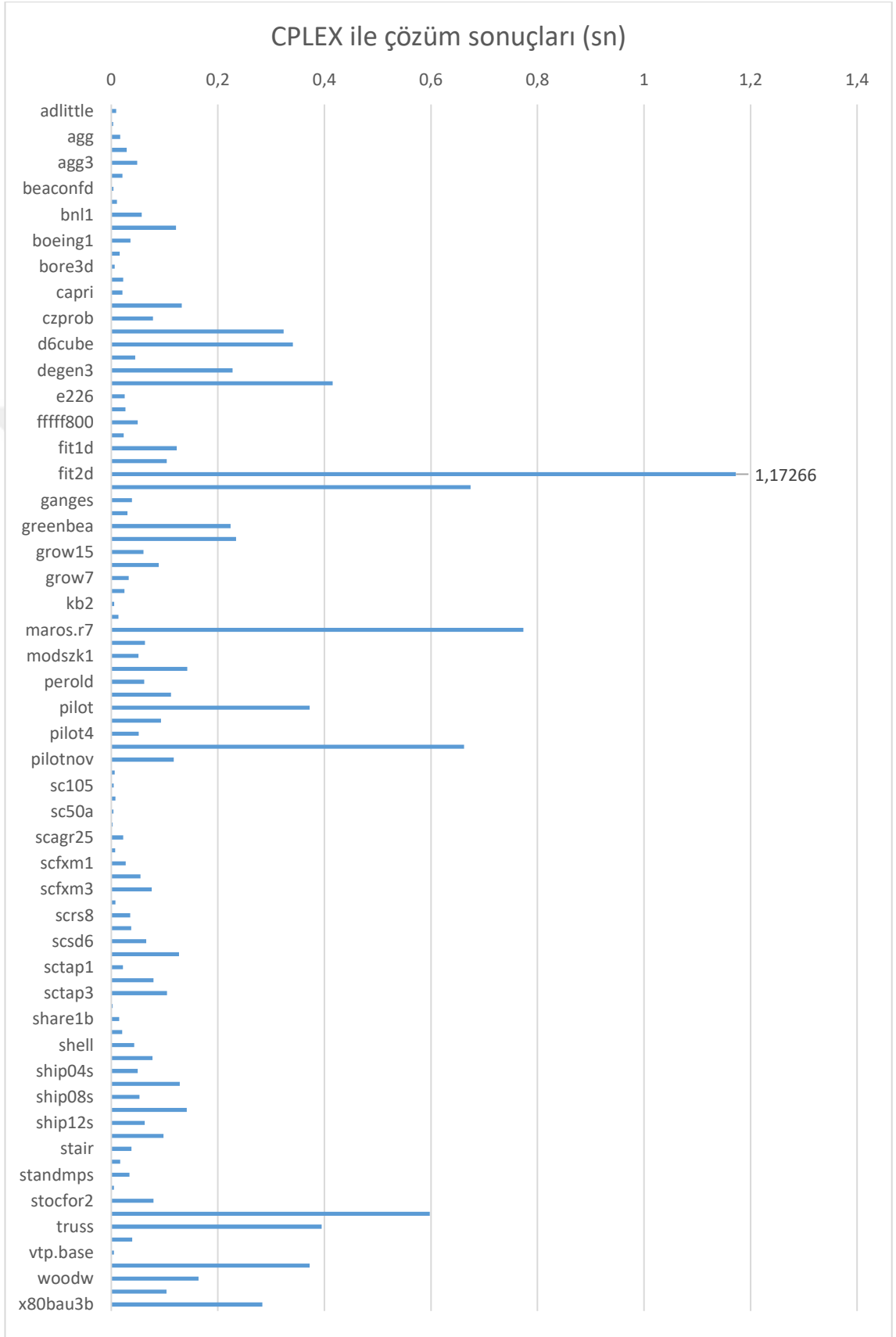


## EKLER

- EK 1** Problemlerinin CPLEX çözücüsü için çözüm süreleri grafiđi
- EK 2** Problemlerinin GUROBI çözücüsü için çözüm süreleri grafiđi
- EK 3** Problemlerinin CLP çözücüsü için çözüm süreleri grafiđi
- EK 4** Problemlerinin GLPK çözücüsü için çözüm süreleri grafiđi
- EK 5** Problemlerinin LP\_SOLVE çözücüsü için çözüm süreleri grafiđi

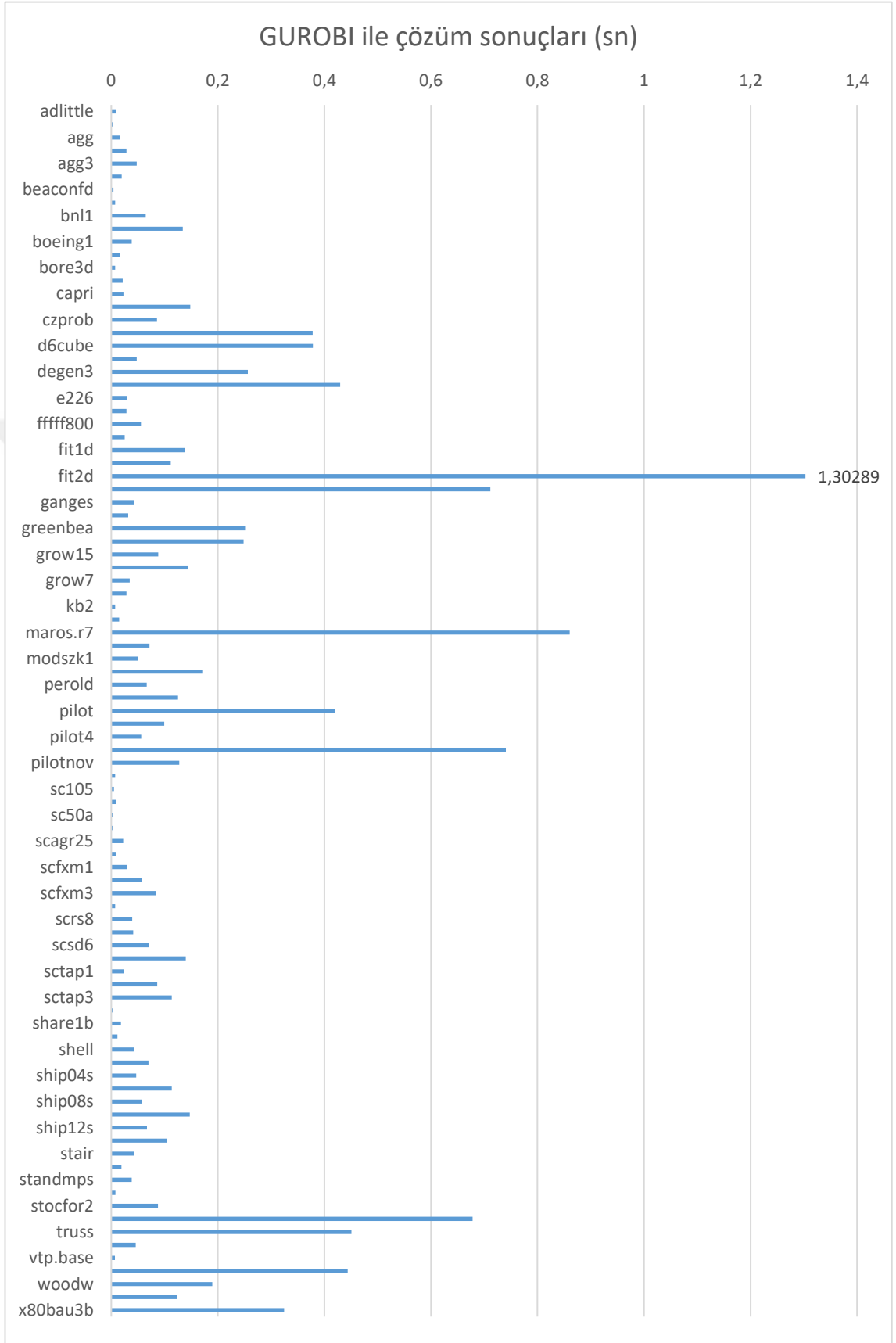


**EK 1** Problemlerinin CPLEX çözücüsü için çözüm süreleri grafiği

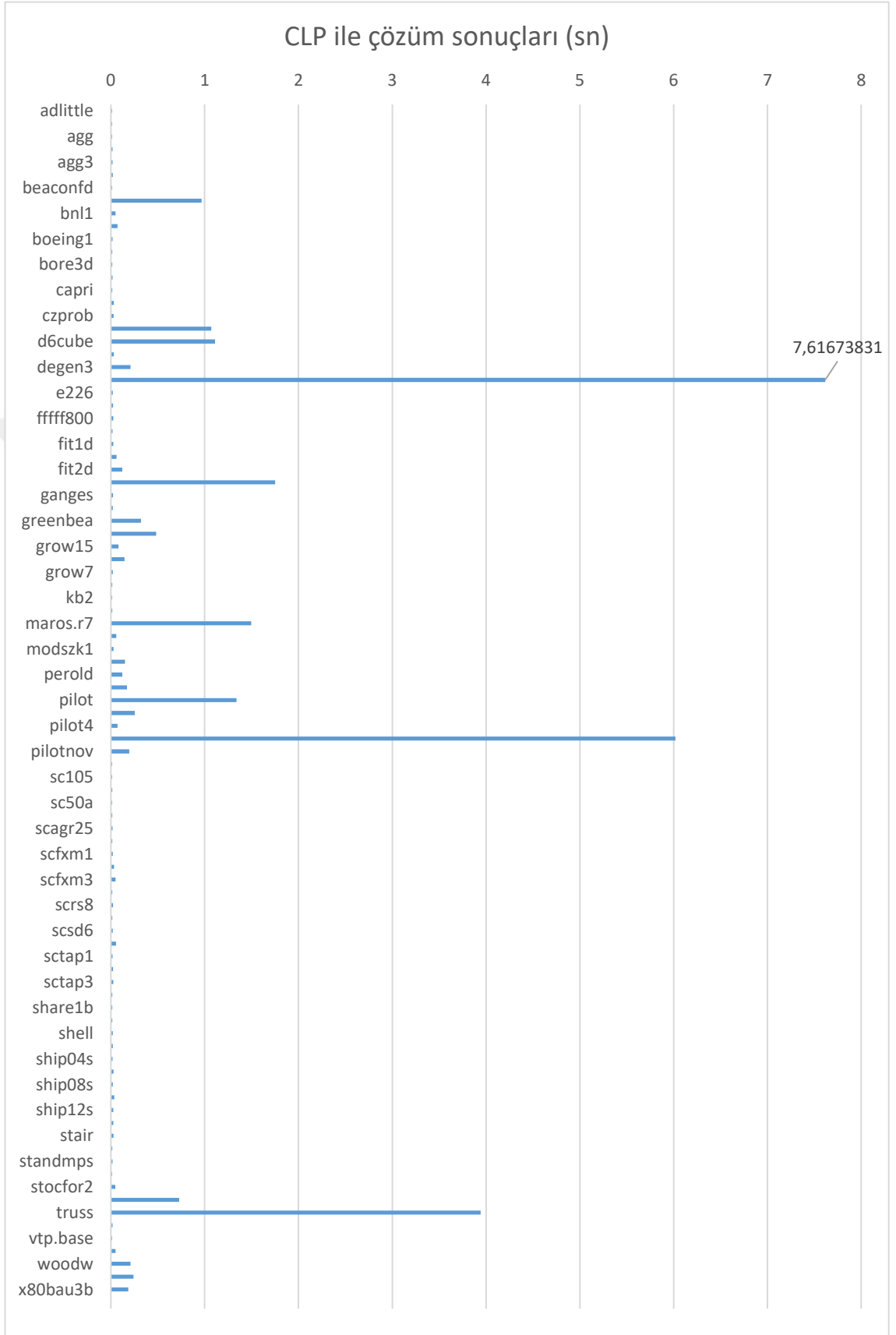




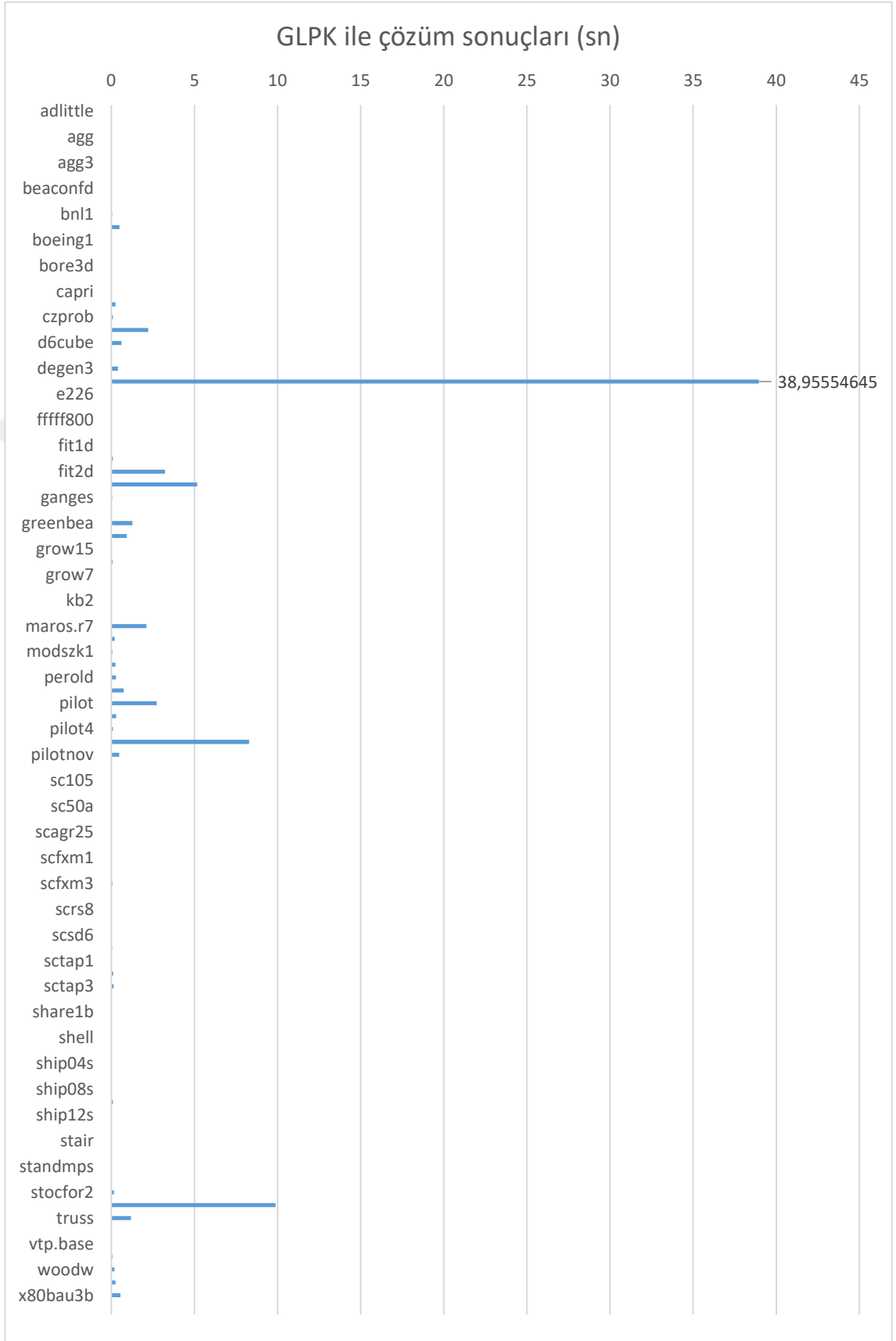
**EK 2** Problemlerinin GUROBI çözücüsü için çözüm süreleri grafiği



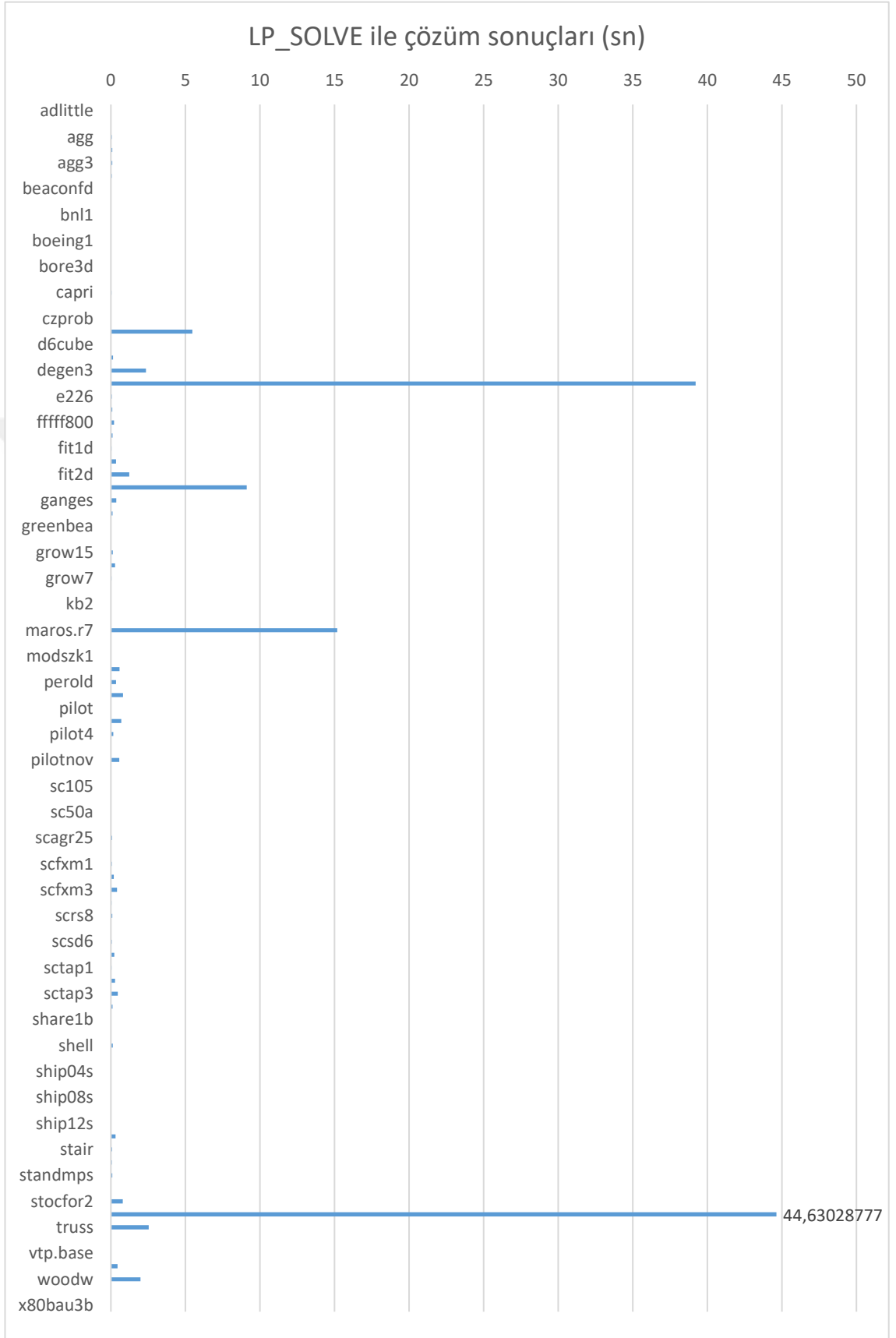
**EK 3** Problemlerinin CLP çözücüsü için çözüm süreleri grafiği



**EK 4** Problemlerinin GLPK çözücüsü için çözüm süreleri grafiği



**EK 5** Problemlerinin LP\_SOLVE çözücüsü için çözüm süreleri grafiği



## ÖZGEÇMİŞ

Adı Soyadı : Merve CÖMERTOĞLU ARIK  
Doğum Yeri ve Tarihi : BURSA / 11.12.1993  
Yabancı Dil : İngilizce

Eğitim Durumu  
Lise : Bursa Nilüfer Milli Piyango Anadolu Lisesi  
Lisans : Uludağ Üniversitesi Endüstri Mühendisliği  
Yüksek Lisans : Uludağ Üniversitesi Endüstri Mühendisliği

Çalıştığı Kurum/Kurumlar : Valeo Otomotiv Sistemleri

İletişim (e-posta) : [mervecmertoglu@gmail.com](mailto:mervecmertoglu@gmail.com)

Yayımları