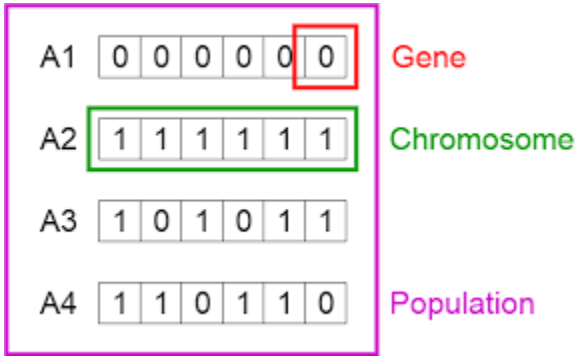
YBS519
Yapay Zeka ve Uygulamaları

Bölüm 5

Genetik Algoritma



Gaziantep Üniversitesi

Yönetim Bilişim Sistemleri, Tezsiz Yüksek Lisans Programı

`http://www1.gantep.edu.tr/~bingul/ai`

Mayıs 2021

İçerik

1. Genetik Algoritma Nedir?
2. Optimizasyon Örneği
3. ga() Fonksiyonu
4. MATLAB’da Genetik Algoritma Araç Kutusu
5. Uygulamalar

1. Kısım

Genetik Algoritma Nedir?

Genetik Algoritma (Genetic Algorithm)

- GA Darwin'in evrim teorisinden esinlenerek geliştirilmiş ardışık ihtimalli yapıya sahip bir arama yöntemidir.
- GA bir gruptaki en uygun bireylerin doğal seçilim (natural selection) yoluyla seçilip sonraki nesile aktarılmasını temel alır.
- GA ile bir problemin çözümü, problemin sanal olarak bir evrimden geçirilmesi ile gerçekleştirilir.
- GA bir veri grubundan özel veriyi bulmak için kullanılır.
- GA daha çok optimizasyon problemlerin çözümüne yardımcıdır.

Genetik Algoritma

- Genetik algoritmalar, bir çözüm uzayındaki her noktayı, kromozom adı verilen ikili bit dizisi ile kodlar.

- Her bir bit gen olarak adlandırılır.

- Her kromozomun bir uygunluk (fitness) değeri vardır.

- GA, tek bir kromozom yerine, bir kromozom kümesini (populasyon) saklar.

Kim Geliştirdi?

Genetik algoritmalar 1960'larda John Henry Holland tarafından ortaya atıldı.

Elektrik Mühendisi ve Bilgisayar Bilimci olan Holland, 1975'te yayınladığı Doğal ve Yapay Sistemlerde Adaptasyon (*Adaptation in Natural and Artifical Systems, MIT Press*) isimli kitabında genetik algoritmayı dünyaya tanıttı.

GA

Genetik algoritma 5 evrede gerçekleştirilir.

- Başlangıç Popülasyonu (Initial population)
- Uygunluk fonksiyonu (Fitness function)
- Şeçim (Selection)
- Çaprazlama (Crossover)
- Mutasyon (Mutation)

Başlangıç Populasyonu

GA algoritma ile bir problemin çözümü popülasyon olarak adlandırılan, bir veri kümesinin belirlenmesi ile başlar.

Kromozom kodlaması genellikle ikili (binary) olarak yapılır.

MATLAB'da, 10 taban ←→ 2 taban dönüşümleri:

```
>> b = dec2bin(25)
b = '11001'
```

```
>> d = bin2dec('11001')
d = 25
```

Uygunluk Fonksiyonu

Uygunluk fonksiyonu, bir kromozomun problem için ne kadar uygun olduğunu belirler. Her kromozom için bir sayı (skor) üretir.

Uygunluk fonksiyonu bazı kaynaklarda **amaç fonksiyonu** olarak da adlandırılır.

Seçim

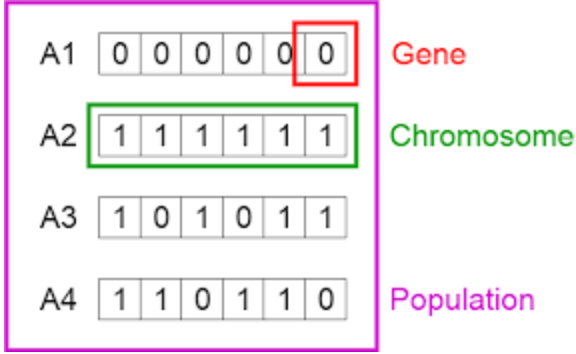
Şeçim evresinin amacı, her bir kromozomdaki en uygun genleri seçmek ve onları bir sonraki nesile (offspring) aktarmaktır.

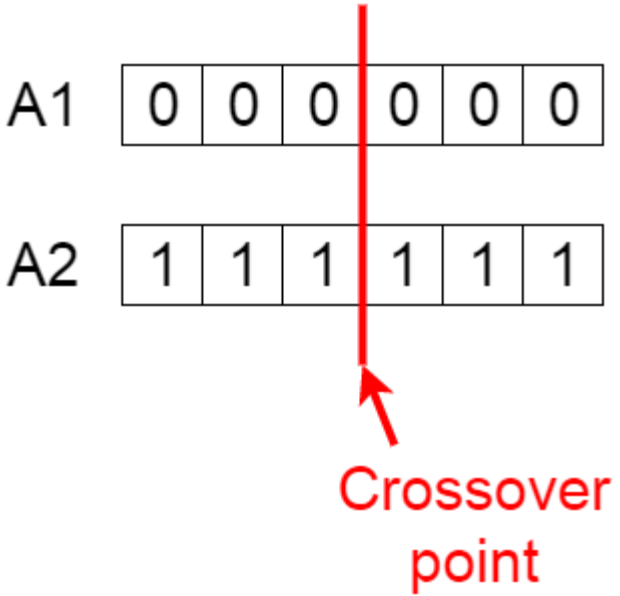
Buradaki nesiller, kodlamada iterasyonlardır.

Her nesilde, kromozomlar çiftler halinde uygunluk skoruna göre seçilir ve skoru büyük olanın yaşama şansı yüksek olur.

Çaprazlama

En önemli evredir. Her çift (ana) kromozom rastgele bir noktadan kesilir ve genler çaprazlama olarak eşlenir.





A5 1 1 1 0 0 0

A6 0 0 0 1 1 1

Bu yeni nesil kromozomlar popülasyona eklenir.

Mutasyon

Oluşan yeni nesil kromozomlarda, bazı genler (düşük olasılıkla) mutasyona uğratılır. Bu kromozom içindeki birkaç genin değerinin terslenmesi ile olur.

Before Mutation

A5	1	1	1	0	0	0

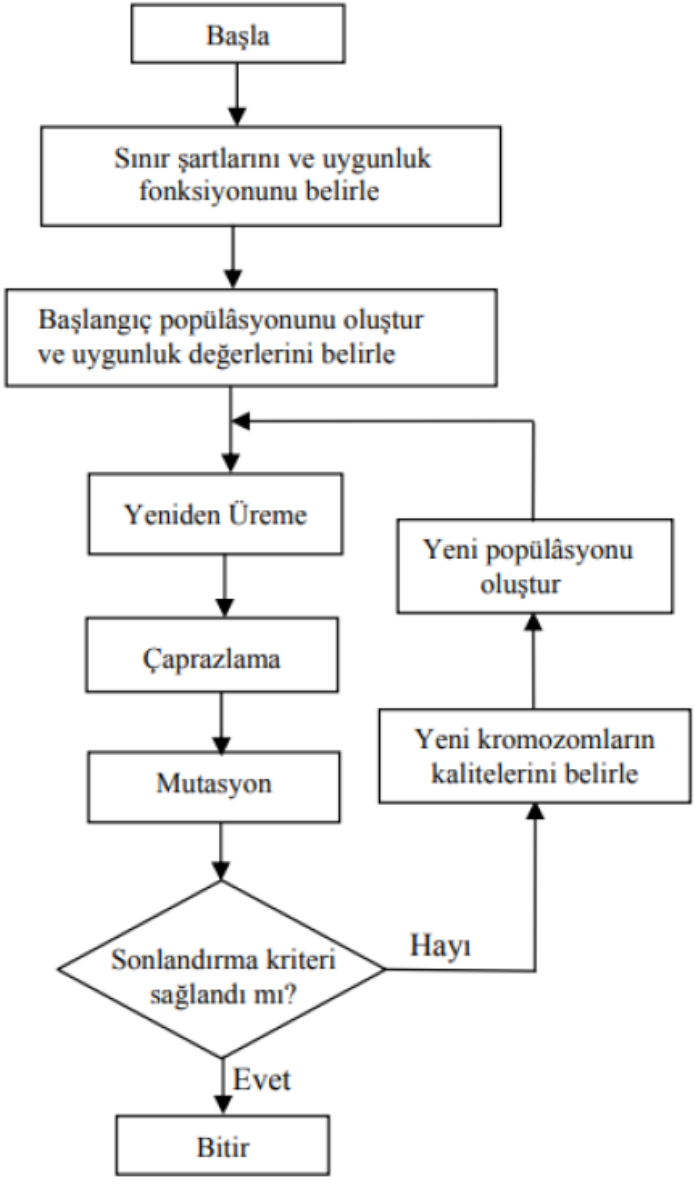
After Mutation

A5	1	1	0	1	1	0

Bu işlem her nesilde uygulanmaz. Birkaç nesilde bir uygulanır.

Sonlandırma

Popülasyonda üretilen yeni nesil genler, önceki nesillerden çok farklı olamıyorsa algoritma (iterasyonlar) sonlandırılır.



2. Kısım

Bir Optimizasyon Örneği

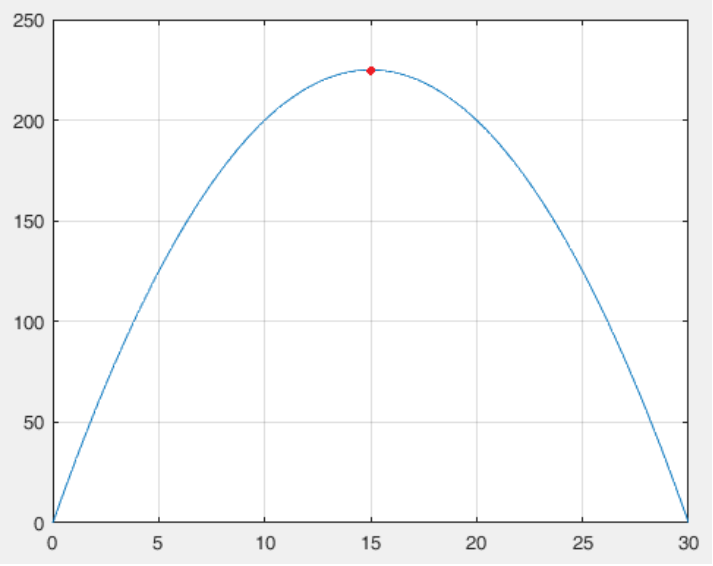
Örnek

Aşağıdaki fonksiyonunun [0, 30] aralığınaki en büyük değerini arayalım.

$$f(x) = 30x - x^2$$

Yani amaç fonk: **y = max[f(x)]**

```
>> x = 0:10;
>> y = 30*x-x.^2;
>> plot(x,y)
```



$$y = \max[\,30x - x^2\,]$$

Başlangıç populasyonu (keyfi olarak) 11011, 10101, 01100, 11110 olsun. Herbir kromozomun sayısal değeri, amaç fonksiyonundaki karşılığı ve seçilme olasılığı aşağıdaki gibidir.

#	K	x	y	p = y/sum(y)
1	11011	27	81	0.1667
2	10101	21	189	0.3889
3	01100	12	216	0.4444
4	11110	30	0	0

mean(y) = <y> = 121.5

1.Nesil: En yüksek olasılığa sahip üç ebeveyn kromozomu çaprazla.

Çaprazlama öncesi		Çaprazlama sonrası
1)	11011	11001
2)	10101	10111
2)	00101	10100
3)	01100	01101

Yeni popülasyon

#	K	x	y	p = y/sum(y)
1	11001	25	125	0.1768
2	10111	23	161	0.2277
3	01101	13	221	0.3126
4	10100	20	200	0.2829

mean(y) = <y> = 176.75

2.Nesil: En yüksek olasılığıa sahip üç ebeveyn kromozomu çaprazla.

Çaprazlama öncesi		Çaprazlama sonrası
2)	10111	01111
3)	01101	10101
3)	01101	10101
4)	10100	01100

Yeni popülasyon

#	K	x	y	p = y/sum(y)
1	01111	15	225	0.2747
2	10101	21	189	0.2308
3	10101	21	189	0.2308
4	01100	12	216	0.2637

mean(y) = <y> = 204.75

2.Nesil*: Rastgele bir kromozomu mutasyona uğrat.

Örneğin 4. kromozom, 3.gen.

Mutasyon öncesi	Mutasyon sonrası
4) 01100	01110

Yeni popülasyon

#	K	x	y	p = y/sum(y)
1	01111	15	225	0.2721
2	10101	21	189	0.2285
3	10101	21	189	0.2285
4	01110	14	216	0.2709

mean(y) = <y> = 206.75

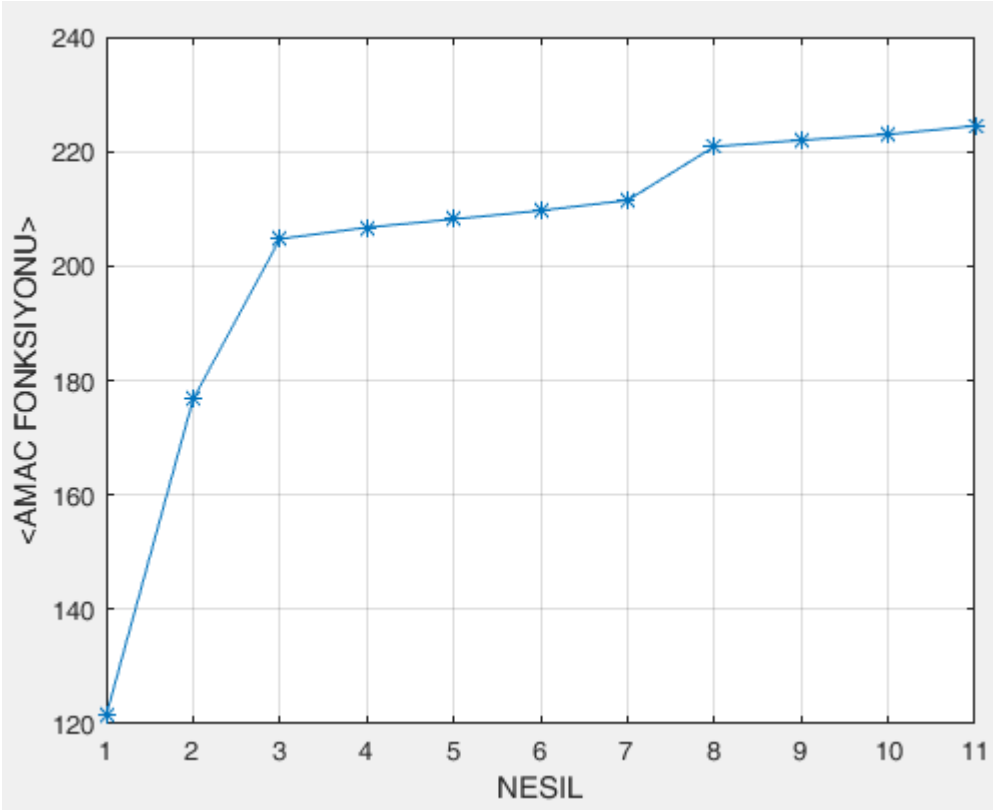
Bu şekilde, yeni nesillerde, işlemler devam ederek aşağıdaki tablo elde edilir.

Yeni popülasyon

#	K	x	y	p = y/sum(y)
1	01111	15	225	0.2500
2	01111	15	225	0.2500
3	01111	15	225	0.2500
4	01111	15	225	0.2500

mean(y) = <y> = 225.0

Nesil	Ort. Amaç Fonk.
0	121.50
1	176.75
2	204.75
2*	206.75
.	
.	
.	
10000	225.00



DİKKAT

Gerçel sayılarla çalışmak gerektiğinde aşağıdaki dönüşüm kullanılılır:

$$C = C_{\min} + \frac{b}{2^L - 1}(C_{\max} - C_{\min})$$

b = ikili kromozom değerinin 10 tabanındaki karşılığı

L = bit dizisinin uzunluğu

$[C_{\min}, C_{\max}]$ problemin doğasına bağlı, çözüme ait aralık

$C = b$ sayısının dönüşümü

GA Uygulama Özeti

```
START
Generate the initial population
Compute fitness
REPEAT
    Selection
    Crossover
    Mutation
    Compute fitness
UNTIL population has converged
STOP
```

3. Kısım

ga() Fonksiyonu

ga() fonksiyonu

MATLAB'da **`ga()`** fonksiyonu ile genetik algoritma tabanlı minimizasyon yapmak mümkündür. Genel Kullanım biçimleri (**help ga**)

```
x = ga(fun,nvars)
x = ga(fun,nvars,A,b)
x = ga(fun,nvars,A,b,Aeq,beq)
x = ga(fun,nvars,A,b,Aeq,beq,lb,ub)
x = ga(fun,nvars,A,b,Aeq,beq,lb,ub,nonlcon)
x = ga(fun,nvars,A,b,Aeq,beq,lb,ub,nonlcon,options)
x = ga(fun,nvars,A,b,[],[],lb,ub,nonlcon,IntCon)
x = ga(fun,nvars,A,b,[],[],lb,ub,nonlcon,IntCon,options)
x = ga(problem)
[x,fval] = ga(___)
[x,fval,exitflag,output] = ga(___)
[x,fval,exitflag,output,population,scores] = ga(___)
```

Örnek

En basit halde, tek değişkenli $y = f(x) = x^2$'inin minimumunu bulmak için:

```
>> f = @(x) x^2;
>> x = ga(f,1)
>> x = -0.0096
>>
```

yada

```
>> f = @(x) x^2;
>> [x fmin] = ga(f,1)
>> x = 0.0156
>> fmin = 2.4406e-04
>>
```

Yani, $x \approx 0$'da, fonksiyon bir minimuma sahiptir ve $f_{minimum} \approx 0$'dır.

Örnek

Maximizasyon problemlerinde, f(x) yerine, –f(x) fonksiyonunun minimumu aranır.

Örneğin, $y = f(x) = 30x – x^2$ maximumunu bulmak için:

```
>> f = @(x) -(30*x - x^2);
>> x = ga(f,1)
>> x = 14.9899
>>
```

yada

```
>> f = @(x) -(30*x - x^2);
>> [x fmin] = ga(f,1)
>> x = 14.9808
>> fmin = -224.9996
>>
```

Yani, x ≈ 15'de, –f(x) bir minimuma sahiptir ve $-f_{minimum} \approx -225$'dir.

Öyle ise, x ≈ 15'de, f(x) bir maximuma sahiptir ve $f_{maximum} \approx +225$'dir.

Örnek

İki veya daha fazla değişken için benzer bir yöntem uygulanır.

Örneğin, $f(x, y) = 10x^2 + 3y^2 - 10xy + 2x$ fonk. minimumu için aşağıdaki program kullanılabilir. [Analitik sonuç: x = -0.6, y = -1.0 ve f_{min} = -0.6 dır]

```
% --- ga ile minimizazyon ---
% uygunluk (fitness) fonksiyonu
f = @(x) 10*x(1)^2 + 3*x(2)^2 -10*x(1)*x(2) + 2*x(1);

% degisken sayisi
desay = 2;

[x fmin] = ga(f,desay)
```

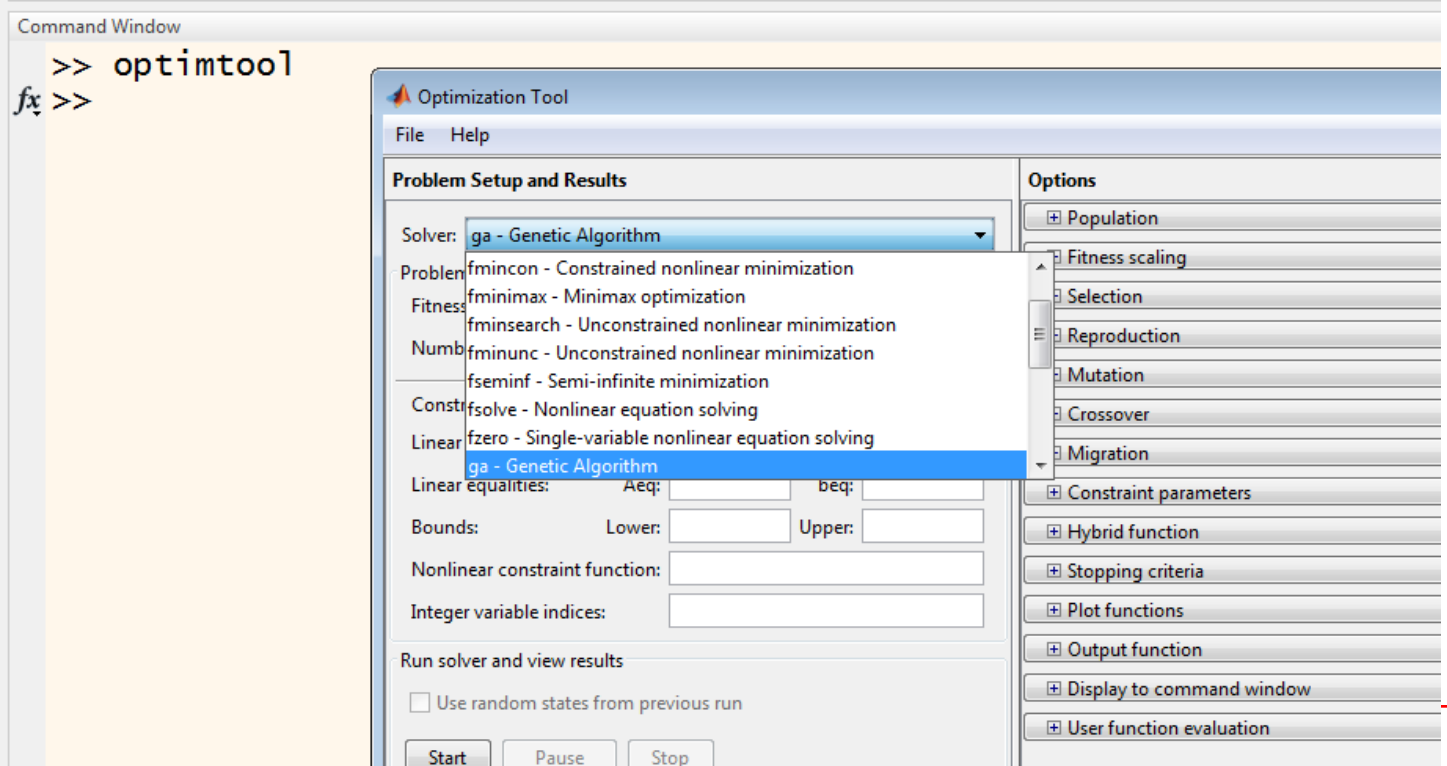
```
x    = -0.5724   -0.9408
fval = -0.5982
```

4. Kısım

MATLAB GA Araç Kutusu

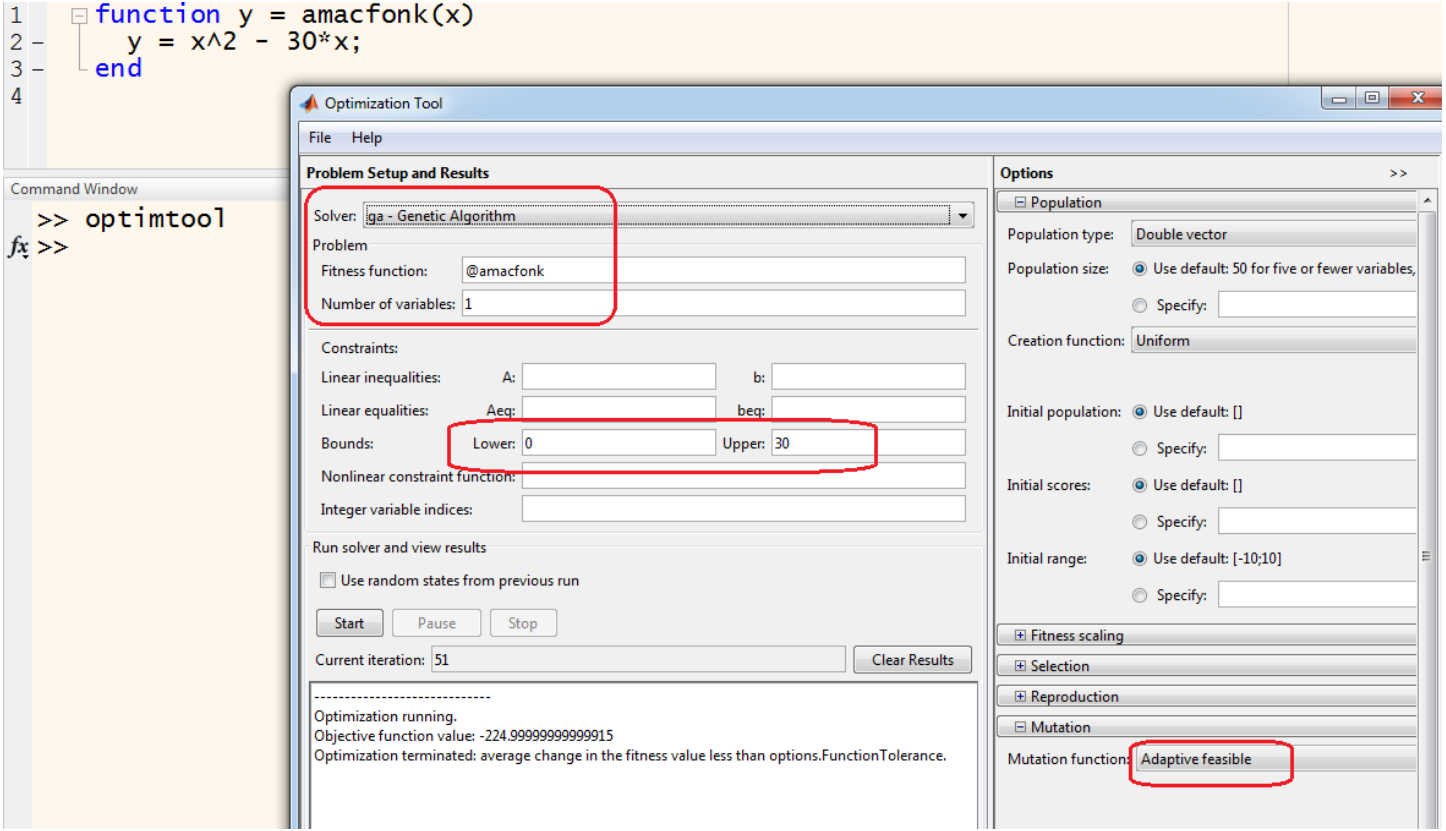
ga

Komut satırında **`optimtool`** yazarak “Optimization Tool Box” başlatılır. Genetik Algoritma ile optimizasyon yapabilmek için, **Solver** menüsünden **ga** seçilmelidir.



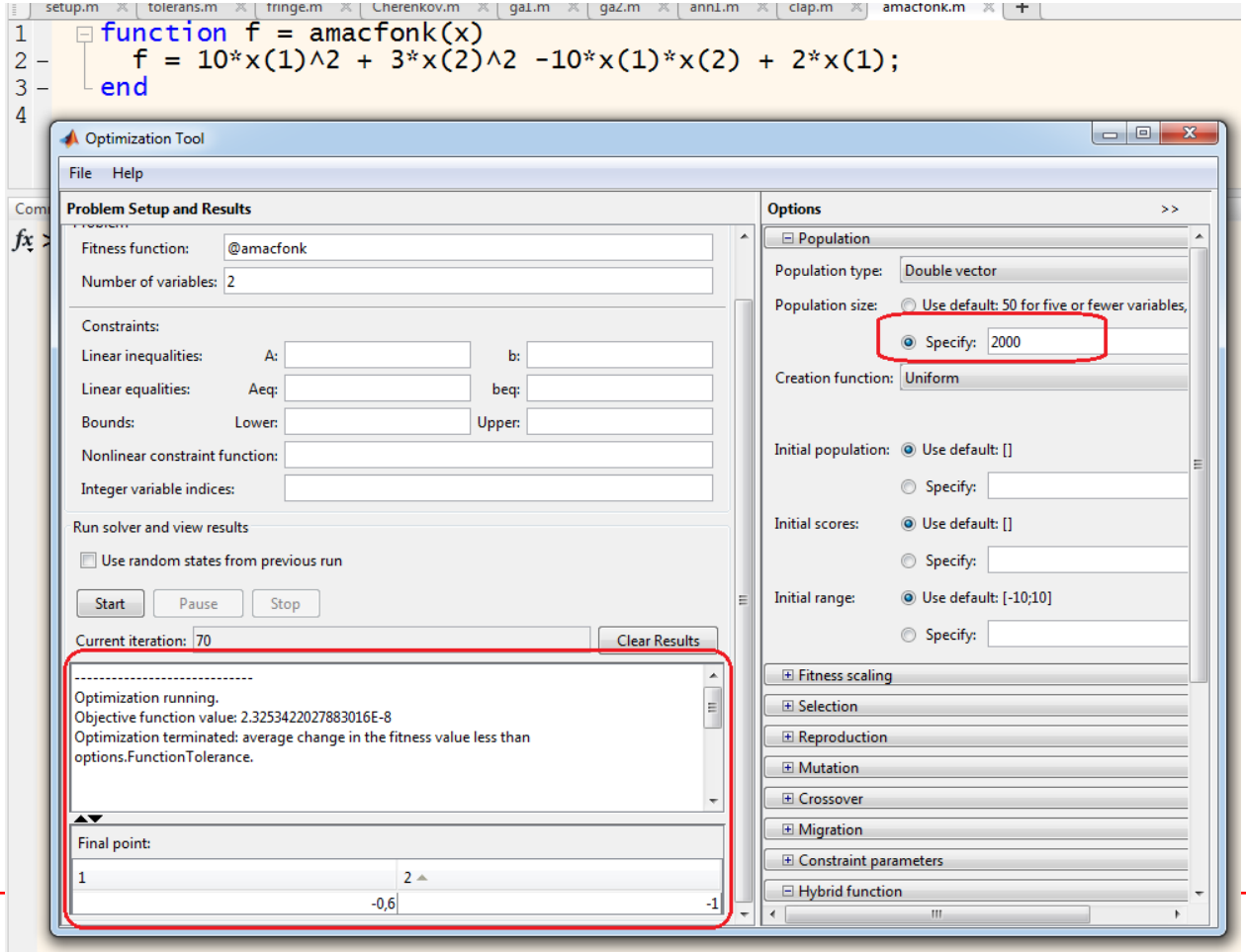
Örnek

$y = f(x) = x^2 - 30x$ minimumu [0,30] aralığındaki minimumu:



Örnek

$f(x, y) = 10x^2 + 3y^2 - 10xy + 2x$ fonksiyonunun minimumu



5. Kısım

Uygulamalar

Örnek – Doğrusal Programlama

Aşağıdaki kısıtları dikkate alarak Z değerini max. yapan x_1 ve x_2 değerlerini bulun.

$$x_1 + 2x_2 \leq 4$$

$$4x_1 + 2x_2 \leq 12$$

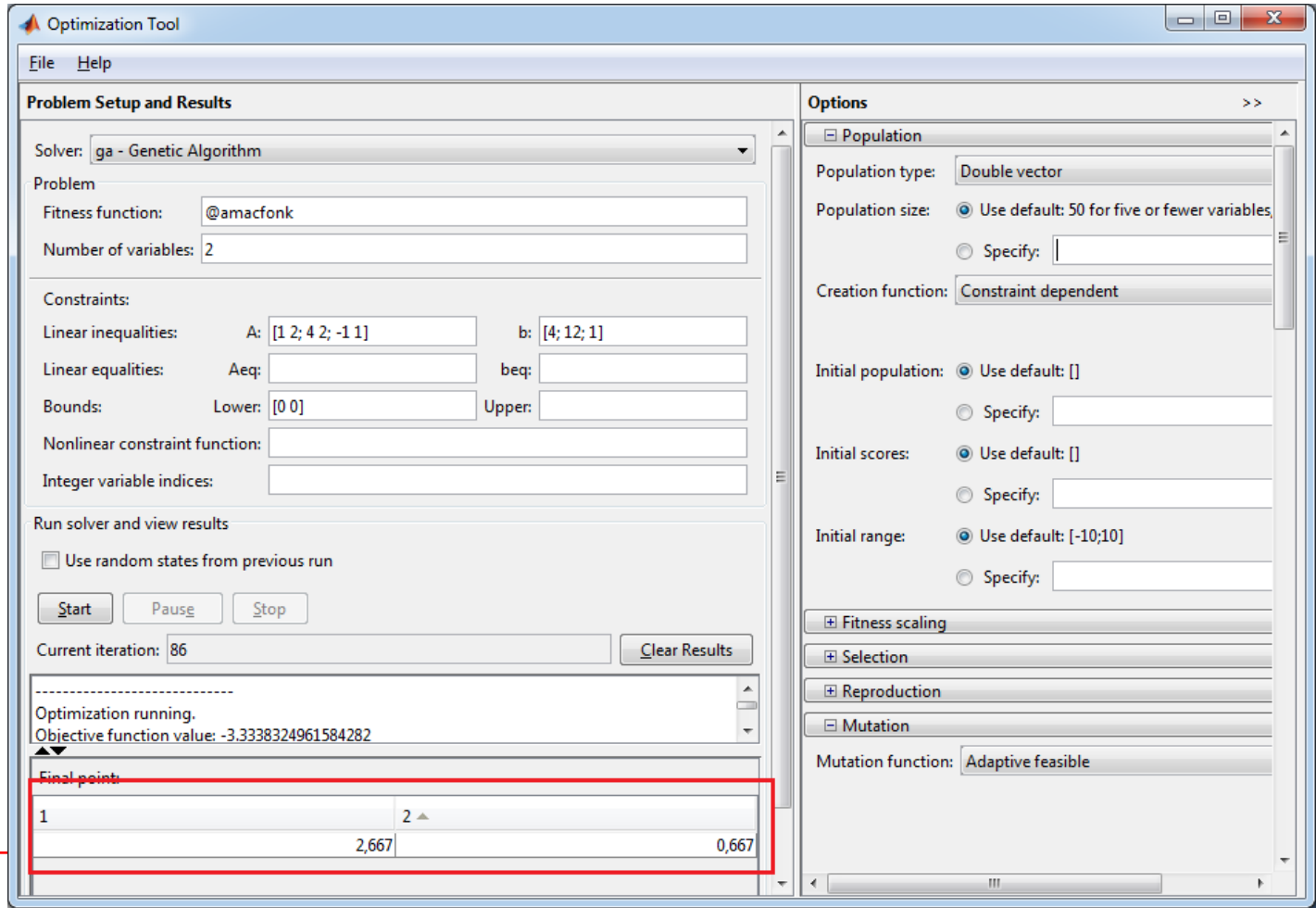
$$-x_1 + x_2 \leq 1$$

$$x_1 \geq 0$$

$$x_2 \geq 0$$

$$Z_{max} = x_1 + x_2$$

Fitness: $f = -x_1 - x_2$;



Örnek – Doğrusal Programlama

Bir oyuncak firması, pilli ve pilsiz oyuncak arabalar üretmektedir. Firma oyuncak arabaları üretirken M1, M2 ve M3 makinelerini kullanmaktadır. Pilsiz oyuncak araba üretilirken bir günde, M1' in 2 saat, M2'nin 1 saat ve M3'ün 1 saat çalışması gerekmektedir. Pilli oyuncak araba üretilirken ise bir günde, M1 makinesinin 1 saat, M2'nin 2 saat ve M3'ün 1 saat çalışması gerekmektedir. M1, M2 ve M3'lerin aylık çalışma saatleri en fazla 180, 160 ve 100 dür. Firma üretilen bir adet pilsiz oyuncak arabadan 40 TL, bir adet pilli oyuncak arabadan ise 60 TL kar etmektedir. Oyuncak firması günlük karını en büyük yapmak için her bir oyuncak arabadan kaç tane üretmelidir? Buna göre problemi, d.p.p. biçiminde modelleyiniz.

Çözüm:

	Makineler			
Çalışma süreleri	**M1**	**M2**	**M3**	**Birim kar**
Pilsiz oyuncak için günlük süre (sa)	2	1	1	40
Pilli oyuncak için günlük süre (sa)	1	2	1	60
Aylık toplam çalışma süresi (sa)	180	160	100	

X_1 : Bir günde üretilen pilsiz oyuncak araba sayısı (adet)

X_2 : Bir günde üretilen pilli oyuncak araba sayısı (adet)

$$
\begin{aligned}
P: \quad \max \; & Z = 40X_1 + 60X_2 \\
& 2X_1 + X_2 \le (180/30) \\
& X_1 + 2X_2 \le (160/30) \\
& X_1 + X_2 \le (100/30) \\
& X_1, X_2 \ge 0
\end{aligned}
$$

Örnek – Gezgin Satıcı Problemi

Gezgin satıcı problemi şu şekilde tanımlanabilir:

* Bir seyyar satıcı var;

* Bu satıcı, mallarını *n* şehirde satmak istiyor;

* Öte yandan, mantıklı bir şekilde, bu satıcı bu şehirleri mümkün olan en kısa şekilde ve her bir şehre maksimum bir kere uğrayarak turlamak istiyor.

Problemin amacı, satıcıya bu en kısa yolu sunabilmektir.