T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GENETİK ALGORİTMA YARDIMIYLA GÜÇ AKIŞI

YÜKSEK LİSANS TEZİ

Elektrik Elektronik Müh. Nur SARMA

Enstitü Anabilim Dalı : ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ
Enstitü Bilim Dalı : ELEKTRİK
Tez Danışmanı : Prof. Dr. Uğur ARİFOĞLU

Nisan 2012

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GENETİK ALGORİTMA YARDIMIYLA GÜÇ AKIŞI

YÜKSEK LİSANS TEZİ

Elektrik Elektronik Müh. Nur SARMA

Enstitü Anabilim Dalı : ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ

Enstitü Bilim Dalı : ELEKTRİK

Bu tez 16/04/2012 tarihinde aşağıdaki jüri tarafından Oybirliği ile kabul edilmiştir.

Prof. Dr. Uğur ARİFOĞLU	**Yrd. Doç. Dr. Yılmaz UYAROĞLU**	**Yrd. Doç. Dr. Kürşat AYAN**
Jüri Başkanı	**Üye**	**Üye**

ÖNSÖZ

Yazmış olunan bu yüksek lisans tezinde günümüzde adından sıklıkla söz ettiren sezgisel algoritma yöntemlerinden biri olan genetik algoritmayı ve uygulama alanlarını incelemektir. Aynı zamanda genetik algoritmanın uygulandığı güç akışı sistemlerinden de bahsedilerek yine çok sıklıkla kullanılan diğer bir güç akışı metodu olan Newton-Raphson algoritmasına da değinilmiştir. Genetik algoritmanın çalışma adımları ve algoritma yapısı açık bir dille ifade edilerek incelenmiştir. Bu inceleme yapılırken genetik algoritma güç akışı üzerine oturtulmaya çalışılmıştır.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR LİSTESİ

IEEE	: Elektrik Elektronik Mühendisleri Enstitüsü
B(b)	: Birim değer
UF	: Uygunluk fonksiyonu
AF	: Amaç fonksiyonu
MW	: Mega-watt
KV	: Kilo-volt
K	: Sabit bir sayı değeri
P	: Penaltı fonksiyonu
KF	: Kısıt fonksiyonu
MO	: Mutasyon oranı
PS	: Populasyon sayısı
l	: Toplam bit sayısı
GA	: Genetik algoritma
ÇN	: Çaprazlama noktası
ÇO	: Çaprazlama oranı
R	: Omik direnç değeri
X	: Endüktif reaktans değeri
Y	: Admitans değeri
Z	: Empedans değeri
J	: Jacobian matris
V	: gerilim genlik
δ	: gerilim faz açısı

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar kelimeler: Genetik Algoritma, Newton-Raphson, Güç Akışı

Çalışmanın amacı genetik algoritmayı ve uygulama alanlarını incelemek ve güç akışı hakkında bilgi vermektir. Çalışmada ilk olarak genetik algoritma kavramı ve temel teoremi hakkında bilgi verilmiştir. Daha sonra, basit genetik algoritmanın çalışma adımları anlatılmıştır. Ayrıca güç akışı ve çeşitleri, algoritma yapısı hakkında bilgi verilmiştir. Genetik algoritmaların çeşitli alanlardaki uygulamaları için birçok kaynak ve makale taranıp incelenmiştir, araştırması yapılmıştır.

POWER FLOW METHOD USING GENETIC ALGORİTHM

SUMMARY

Key Words: Genetic Algorithm, Newton-Raphson

The aim of this study is to investigate the genetic algorithm and its application areas and to give the basic knowledge of the power flow. Initially, the concept of the genetic algorithm and its fundamental theorem are defined. The operating principle of the genetic algorithm is then documented. Furthermore, the power flow and its types including the structure of algorithm are presented. Significant literature review on the several application areas is carried out.

BÖLÜM 1. GİRİŞ

1997 yılında İskoçya Edinburgh'da yapılan Dolly adlı bir koyunun kopyalanması deneyi ile genetik bilmi farklı bir boyut alma sürecine girdi. Bu süreç genetik algoritmanın keşfi ve bu algoritmanın hemen hemen tüm bilimsel alanlarda kullanılması ile hızlandı.

Sayısal optimizasyon yöntemlerinden olan genetik algoritmalar, evrimsel hesaplama tekniklerinin bir parçasını oluşturur ve bu gün daha önceden kullandığımız genetik algoritma yöntemlerinin çözümünde zorlandığı bir çok problemin çözümünde, gerek endüstride gerekse birçok bilimsel deneylerde verdiği verimlilik sebebiyle tercih sebebi olmuştur.

Genetik algoritma genel olarak Darwin tarafından ortaya atılan evrim teorisine dayanır. Bu teorisi güçlü olan bireylerin yaşamını sürdürürken, güçsüz olanların ise eleneceği bir sistemi öne sürer.

Genetik algoritmanın dayandığı evrimsel hesaplama mantığı ilk olarak 1973 yılında I.Rechenberg tarafından ortaya atılmış lakin bu algoritma mantığının bilgisayara taşınma olayı John Holland tarafından gerçekleştirilmiştir.

Genetik algoritmanın elektrik enerji sistemlerinde kullanımına değinecek olur isek, günümüzde artan nüfus yoğunluğu ve de mevcut kaynakların bu nüfusun gerektirdiği enerjiyi karşılayamaması sebebiyle santrallerin en verimli ve en uygun maliyetli halde çalışması ihtiyacı doğmuştur. Biz bu çalışma koşullarına optimum çalışma koşulları diyoruz. Ayrıca günümüzde enerji sektöründe yapılan birçok özelleştirme sayesinde enerji sektörü tarif edilemez şekilde büyümüş, bu büyüme maliyet açısından daha ekonomik enerji sistemlerinin işletilmesi ve halkın talep ettiği ucuz elektriğin üretilmesi ihtiyacını doğurmuştur. İşte bu ekonomik dağıtım

problemlerinin çözümünde güvenilir, hızlı ve en optimum sonucu veren algoritma yapıları aranmış ve halen de aranmaktadır. Son yıllarda bunu karşılamak için sezgisel metotlar yani yapay zeka algoritmaları; genetik, yapay arı kolonisi, parçacık sürü optimizasyonu, diferansiyel gelişim, karınca kolonisi ısıl işlem gibi metotlar kullanılmaktadır.

Evrimsel yaklaşımlardan olan genetik algoritmaların sürekli geliştirilebilir olması onu birçok alanda kullanılmasının başlıca nedeni olmuştur. Genetik algoritmalar günümüzde yumuşak hesaplama (soft computing) yöntemleri ile iç içe geçirilerek hibrid (hybrid) çözümler elde edilmiştir [1-7].

BÖLÜM 2 GENETİK ALGORİTMA

2.1. Genetik Algoritmanın Temeli

Genetik algoritma evrimsel hesaplama mantığına (evalutionary computing) dayalı bir yapay zeka algoritma türüdür ve Darwin'in evrim teorisine dayanır. Evrimsel hesaplamalar genel manada 3 kısıma ayrılmıştır; genetik algoritmalar, evrimleşme stratejileri (evolution strategies) ve de son olarak genetik programlama (genetic programming). Bu teknikler arasında birçok benzerlikler mevcuttur lakin aralarında önemli farklarda vardır. Bunlar arasındaki farklardan bazıları; evrimleşme stratejisi yalnız mutasyon işlemini kullanırken genetik algoritma çaprazlama ve mutasyon işlemini kullanır. Ayrıca bunlar temsil (represenration) amacı ile farklı veri yapılarını ve bundan dolayı da farklı genetik operatörleri kullanabilirler. Ayrıca Günümüzde bunların çeşitli özelliklerini bir arada bulunduran birçok melez algoritmalarda vardır. Genetik algoritma esnasında algoritmada yer alan bireyler doğadaki gibi eşleşirler ve çoğalarak neslin devamlılığını sağlarlar. Neslin devamlılığını sağlarken de evrim teorisinde de açıklandığı gibi iyinin yaşaması zayıf olanında ortadan kalkması olayı söz konusudur. Bu yüzden algoritma incelendiğinde başlangıç noktasından algoritmanın sonuna kadar sürekli iyiye giden sonuçlar ile karşılaşılacaktır. Her yeni nesil maruz kaldıkları özel genetik operasyonlarla (seçim, çaprazlama ve mutasyon) daha uygun bireyler olarak şekillenir [8].

2.2. Genetik Algoritmanın Doğuşu

Evrimsel hesaplama mantığı Rechenberg'in 'Evrim Stratejileri 1973' adlı eserinde ortaya atılmış, fakat bilgisayar kullanarak genetik algoritma oluşturulması ilk olarak Michigan Üniversitesi psikolog ve bilgisayar bilimleri profesörü John Holland tarafından yapılmış ve ''Adaptation in Natural Systems'' kitabını yayınlamıştır (1975). Holland canlılarda meydana gelen genetik değişimleri bilgisayarda gerçekleştirerek genetik algoritmanın temellerini atmıştır. Holland'ın öğrencisi olan

Illinois Üniversitesi profesörü David Golldberg genetik algoritmayı çeşitli uygulamalarda kullanarak yaygınlaştırmış ve 1989 yılında ‘’Genetic Algorithms in Search, Optimisation and Machine Learning’’ adlı kitabı yayınlamıştır. Genetik algoritmalar kullanarak genetik programlamaların geliştirilmesi ise 1992 yılında John Koza sayesinde olmuştur [9-11].

2.3. Neden Genetik Algoritma?

Genetik algoritmalar bugün birçok yerde kullanılmaktadır. Bilhassa deneysel çalışmalarda, mühendislik uygulamalarında çoğunlukla optimizasyon amacıyla ve de endüstrinin birçok alanında karşımıza çıkmaktadır. Günümüzde genetik algoritma; uygulanacağı alan geniş, kompleks, çözüme yönelik mevcut bilgi az ya da mevcut bilgi ile matematiksel çözümleme yapılamıyor ise yahut geleneksel metotlar ile iyi sonuçlar elde edilememiş ise kullanılır. Genetik algoritmaların yanında bugün ısıl işlem, tabu araştırma, karınca kolonisi, yapay bağışıklık, diferansiyel gelişim, parçacık sürü, yapay arı kolonisi algoritmaları gibi birçok yapay zeka algoritmaları mevcuttur. Geleneksel algoritma yöntemleri ile çözümlenmesi zor olan çok değişkenli problemlerde genetik algoritmadan sıklıkla faydalanılır çünkü daha kolay ve hızlı çözüm sunmaktadır[26]. Genetik algoritmalar bize karmaşık yapıları basit bit dizileri şeklinde kodlayarak problemlerin çözümünde kolaylık sağlar. Karmaşık problemlerin çözümünde birçok geleneksel yöntem; gradient, lineer olmayan programlama, quadratic programlama, lineer programlama, interior point, Newton-Raphson algoritma yöntemleri gibi, kullanılırsa zorluklar (local optimal solution) ortaya çıkmıştır. Eğer problemimiz non-diferansiyel, non-convex veya non-lineer ise genetik algoritma ile çözümleme yapıldığında daha optimum (global optimum solution) sonuç elde edilir Ayrıca genetik algoritma başlangıç popülasyonunu her jenerasyonda geliştirdiği için programın sonunda bize en uygun çözümü verir [9-11].

2.4. Genetik Algoritmanın Yapısı ve İşleyişi

Bilindiği üzere genetik algoritma, doğal seçim ve genetiğe dayalı algoritmadır (sezgisel). Eğer genetik algoritma optimizasyon problemlerinde kullanılıyor ise diğer optimizasyon metotları en optimal çözümü bulmada başarısız olurken, genetik

algoritma en kullanışlı ve güvenilir optimizasyon metotlarından biridir. Fakat bir dezavantajı olarak en son sonucu bulmak için yaptığı çok fazla sayıdaki hesaplama sayılabilir lakin günümüzde bu hızlı bilgisayarlar sayesinde önemsiz kalmıştır.

Genetik algoritma popülasyon denilen genelde ikili (binary) bit dizilerinin kodlanmasıyla oluşan vektörlerin birleşmesiyle oluşur. İkili kodlama yerine değer, genetik, ağaç, permütasyon veya alellik kodlama türleri de kullanılabilinir bu seçim problemimizin ne olduğuyla alakalıdır. Kodlama türüne göre genetik algoritmanın cinsi ve de hızı değiştirilmiş olur. Her bir vektöre birey (genotip) adı verilir ve bu bireyler popülasyonda mevcut olan değişkenlerin yan yana gelmesi ile oluşmuştur. Popülasyonda kaç adet değişken var ise popülasyondaki kromozom sayısı da o kadardır. Bu birey sayısını yani popülasyonun kaç bireyden oluşacağını aşağıda verilecek formülasyondan yararlanarak bulunur (n boyutlu). Ve bu bit dizilerinin 3 farklı operasyon ile 1-seçim (selection) 2-çaprazlama (crossover) 3-mutasyon (mutation) ile değişikliğe uğratılarak yeni nesiller (jenerasyonlar) üretilir ve en uygun çözümü bulması sağlanır. Algoritma başlangıçta rastgele bit dizileri ile üretilmiş başlangıç popülasyonu ile temsil edilir. Daha sonra yukarıda verilen 3 adım uygulanarak yeni popülasyon üyeleri bulunur ve en uygun cevap bulununcaya kadar bu böyle devam eder. Sonuçta popülasyondaki bireylerden iyi olanları seçilmiş kötü olanları ise elenmiş olur [8-14].

değişkenler	
X1	X2
11110000	01010101
1111000001010101(bir birey)	

Şekil 2.1. İki Farklı Değişkenin Bir Birey Oluşturması

Bu yöntem de bazı çözümler başlangıçta tahmini olarak biliniyor ise ve bunlarda en başta kullanılır ise optimizasyonun işlerken zaman açısından tasarruf sağlayacak ve böylece algoritmanın daha kısa sürede yakınsamasını sağlayacaktır.

Başlangıç popülasyonunun oluşturulmasına örnek verilir ise; 2 (x,y) değişkenli bir problemin genetik algoritmada incelenmesini yapmak istersek;

$0 \leq x \leq 15$

$0 \leq y \leq 20$

olsun ve de değişkenlerin 1 adım aralığıyla değiştiğini kabul edilirse;

x=0,1,2,…,15

y=0,1,2,…,20

o halde bit (gen) sayısı yani değişkenin kaç adet ikili (binary) koddan oluşacağını;

$$2^{ln} \geq \frac{X_{n\max} - X_{n\min}}{e} + 1 \qquad (2.1)$$

ln: n. değişkenin bit sayısı

X_{nmax} = n. değişkenin üst sınır değeri

X_{nmin} = n. değişkenin alt sınır değeri

e =artım aralık değeri

O halde 2.1 formülünden yararlanarak,

x için;

$$2^{l_x} \geq \frac{15-0}{1} + 1 \Rightarrow 2^{l_x} \geq 16 \Rightarrow 2^{l_x} \geq 2^4 \Rightarrow l_x = 4$$

x değişkeni 4 bit uzunluğunda olacak,

y için;

$$2^{l_y} \geq \frac{20-0}{1} + 1 \Rightarrow 2^{l_y} \geq 21 \Rightarrow 2^4 = 16, 2^5 = 32 \Rightarrow l_y = 5$$

y değişkeni 5 bit uzunluğundadır.

x ve y değişkenlerinin oluşturacağı bir bireye şöyle örnek verilebilinir;

x=0000, y=11111 olur ise

birey=000011111 olup 9 bit (4+5)uzunluğunda olacaktır.

Popülasyon sayısını yani popülasyonu oluşturacak bireylerin sayısını ise [15];

$$pop\_say \geq 1.65 * 2^{0.21*l} \tag{2.2}$$

formülü ile bulabiliriz. Formüldeki,

pop_say =popülasyonda yer alacak toplam birey sayısı

l=bir bireyin içerdiği toplam bit sayısı

O halde 2.2 formülüne göre yukarıda verdiğimiz x ve y değişkenlerinin oluşturacağı popülasyonunun birey sayısı;

$$pop\_say \geq 1.65 * 2^{0.21*l} \Rightarrow 1.65 * 2^{0.21*9} \Rightarrow 6.115$$

$pop\_say \geq 6.115 \Rightarrow pop\_say = 7$ tam sayısı seçilir.

x ve y değişkenlerinin oluşturduğu satır sayısı 7 sütun sayısı 9 olan bir matris başlangıç popülasyonu olarak herhangi bir bilgisayar programı yardımı ile oluşturulmalıdır. MATLAB programı kullanarak 9 satır 7 sütunluk, sadece ‘0’ ve ‘1’ den oluşan rastgele bir matris oluşturulmak istenir ise;

```
>>A=randsrc(7,9,[ 0 1])
```

A =

1	1	0	1	1	1	1	1	1
1	0	1	0	1	1	0	0	1
1	0	1	1	0	0	1	1	1
0	1	1	0	1	0	1	0	1
1	1	1	0	0	0	0	1	0
1	1	1	0	0	1	0	0	0
0	1	0	0	0	1	0	1	0

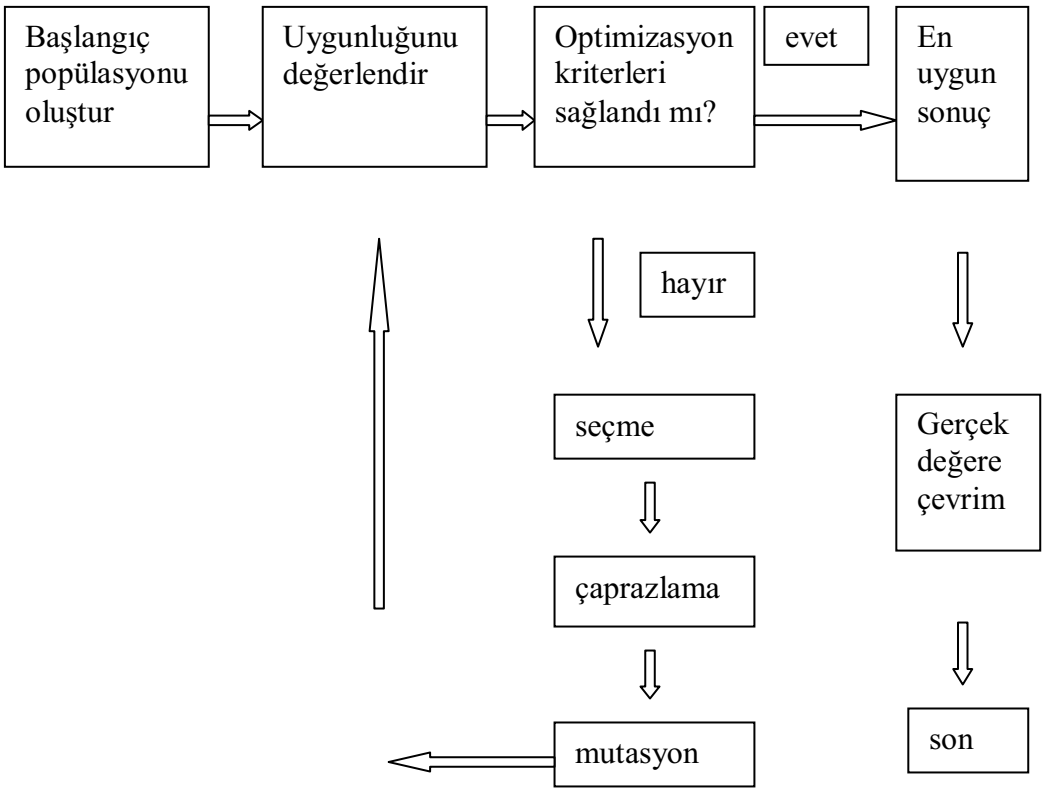
sonucunu elde edilir. Başlangıç populasyonu;

Birey numarası	x	y	
			⇒ 9 bit
1. birey	1101	11111	
2. birey	1010	11001	
3. birey	1011	00111	
4. birey	0110	10101	⇒ 7 birey
5. birey	1110	00010	
6. birey	1110	10000	
7. birey	0100	01010	

Şekil 2.2. Dört ve Beş Değişkenli İki Farklı Bireyin Oluşturduğu Popülasyon

şeklindedir.

Programın başında Şekil 2.2'de gösterildiği gibi, programın sonucunda olması istenilen değeri programa bildirmek için bir uygunluk fonksiyonu (objective function) tanımlanır. Her birey bu uygunluk fonksiyonuna ulaşmayı amaçlar ve programın sonunda bireyin uygun olup olmadığı bu fonksiyona göre değerlendirilir. Eğer bireyler uygunluk fonksiyonu ile uyumlu ise bir sonraki nesle aktarılır. Uygunluk fonksiyonuna yakın olan bireylerin belirlenip bir sonraki nesle aktarılması olayına seçim denir. Seçilmiş bireylerin bir araya gelip üremesi olayına çaprazlama denir. Çaprazlama olayından sonra optimizasyonda yeni bireyler üremiş olur. Daha sonra bu yeni bireylerin bazı genlerinde değişiklik yaparak, yani 0 ise 1 veya 1 ise 0 yapılarak, bu değişiklik daha sonra verilecek mutasyon oranınca yapılacaktır, yeni bireylerin (children, offspring) eski bireylerin (ebebeyin, parents) kopyası olması önlenmiş olur. Yukarıda görüldüğü üzere başarılı bireyler yani uygunluk fonksiyonuna yakın bireyler sonraki nesle aktarılırken başarısız bireyler ortadan kalkarlar, her iterasyon adımında böyle devam ettikçe bizim popülasyonumuzda ki bireylerimizin kalitesi de artmış olacaktır. Programın en sonunda da tüm bireyler başarılı olacak ve istenilen uygunluk fonksiyonunu sağlamış olacaktır [15].



.

Şekil 2.3. Genetik Algoritma Yapısı

Genetik algoritmanın iterasyon sayısının ne zaman sona ereceğine birkaç farklı yöntemle karar verebilinir. Bunlar; programın çalışma süresi, elde ettiğimiz sonuçların belirli bir zaman boyunca aynı olması yada jenerasyon sayısının istenilen sayıya ulaşmasıdir [15].

2.4.1. Uygunluk fonksiyonu (UF) (fitness function)

Uygunluk fonksiyonu her birey için ayrı değerlendirilir ve her kromozomun ulaşmak istediği amaç değeridir. Yani uygunluk amaç fonksiyonu yardımı ile oluşturulur. Uygunluk fonksiyonu, amaç fonksiyonundan (goal function, AF) ceza faktörü (penalty function, CF) kadar farklıdır. Ceza faktörünü kısıt değerleri belirler ve kısıtları zorlayan bireyleri cezalandırmaya yarar. Amaç fonksiyonu ise mevcut olan değişkenlerin almasını istenilen durumlarının fonksiyonudur. Örneğin, amaç fonksiyonumuz güç sistemlerindeki toplam aktif güç kayıplarını minimum olması olabilir.

$$UF=AF\pm CF \tag{2.3}$$

Formül 2.3'den görüldüğü üzere uygunluk fonksiyonu, amaç fonksiyonu ile ceza fonksiyonun toplamı ya da farkıdır. Eğer maksimizasyon problemi yapıyor isek farkı, minimizasyon problemi yapıyor isek toplamı şeklinde yazılır. Ceza fonksiyonunun olup olmaması problemdeki kısıtlara bağlıdır örneğin bir jeneratörün üreteceği maksimum güç bir kısıt oluşturur.

Genetik algoritma kısıtları olmayan algoritma türüdür. Kısıtları olan problemlerde problemlerinde kısıtların ihmal edilmesi halinde, amaç fonksiyonunu ceza fonksiyonu ile cezalandırıp kısıtsız hale getirilir ve böylece amaç fonksiyonu da belirli sınırlar içinde tutulmuş olur [15].

$$UF=T+AF\pm CF \tag{2.4}$$

Burada eklenilen T değeri, büyük bir tam sayı değeri seçilerek uygunluk fonksiyonunun negatif bir değer alması önlenir. Programın sonunda bu T değeri çıkarılarak asıl değeri bulunur.

Ceza fonksiyonundan daha açık şekilde bahsedecek olur isek; ceza katsayıları ile çarpılmış kısıt fonksiyonların toplamından oluşur. Ceza fonksiyonu;

$$CF=\sum_{i=1}^{n} p_i * Kf_i(x_1,x_{2,}...)^2 \tag{2.5}$$

p_i = i. Kısıt fonksiyonu için ceza katsayısı

Kf_i = i. Kısıt fonksiyonu

Bireylerin uygunluk fonksiyonu ile uyumu araştırılmadan önce ikili kod sisteminde olan birey değerlerinin gerçek değerlerine yani onluk sisteme çevrilmesi gerekir. Onluk sisteme çevrilen her birey için ayrı ayrı uyumu aranır. Eğer uygun uyum sağlanamaz ise bu bireyler eski bireyler sınıfına sokularak seçim, çaprazlama ve mutasyon geçirerek yeni bireyleri oluşturur ve bunlarında uyumu aranır. Genetik algoritmamızı sonlandıracak parametrelere, örneğin olmasını istediğimiz maksimum jenerasyon, ulaşıncaya kadar bu uyumu arama işlemi devam eder. Uygunluk

fonksiyonu ne kadar verimli ve hassas ise genetik algoritma o kadar başarılı olacaktır [15].

2.4.2. Genetik algoritma operatörleri

Her birey için uygunluk fonksiyonu değerleri hesaplandıktan sonra içlerinde istenilene yakın bireyler seçilerek bir sonraki nesile (genleri) aktarılarak yeni bireyler oluşturmak istenecektir. İşte bu yeni nesil oluşturma aşamasında elitizim, seçim, çaprazlama ve mutasyon' dan oluşan genetik algoritma operatörleri kullanılır. Bu operatörlerin kullanarak başarılı bireyleri koruyup çeşitliliği arttırarak genetik algoritmanın performansını iyileştirmiş olunur [15].

2.4.2.1. Elitizim (Elitism, Seçkincilik)

Elitizim, popülasyon içerisinde yer alan bireylerden uygunluk fonksiyonu ile uyumu en iyi olanın yeni oluşturulacak popülasyon içerisinde yer almasını sağlamaktır. Bu, genellikle yeni oluşturulan popülasyonun ilk bir elemanının yerine bu elit birey ya da bireylerin değerleri yazılarak sağlanır. Böylece eski jenerasyonun en uyumlu bireyi yeni jenerasyonda da varlığını koruyacaktır. Bunu yaparak yeni jenerasyona iyi ebeveynler girecek ve bunların üreteceği bireylerden de daha iyi sonuçlar alınabilecektir [15].

2.4.2.2. Seçim (Tekrar üreme, Reproduction, Selection, Representation)

Popülasyon içinde yer alan bireylerin hangilerinin seçilerek yeni bireyleri üreteceğini bulma işlemine seçim denir. Seçme işleminde de yine uygunluk fonksiyonuna olan uyum ile yeni bireyleri (çocuk, offspring) üretecek eski bireyler (ata, ebeveyn) aşağıda verilecek farklı birtakım yöntemler ile belirlenir. Seçim işleminde uyumu fazla olan bireylerin genlerinin diğer nesile aktarılma olasılığı yüksek olur iken, uyumu düşük olan zayıf bireylerin ise seçilme olasılığı zayıftır. Seçim işlemi için bilinen metotlar; turnuva seçimi (tournament selection), rulet tekerleği seçimi (roulette wheel), Boltzman seçimi, sıralama seçimi (rank selection), sabit durum seçimi (steady state selection) gibi [15].

Rulet Tekerleği Seçimi: Bu seçme yönteminde uygunluk fonksiyonuna uyumu yüksek olan bireyin seçilme şansı yüksek, düşük olan bireyin ise azdır. Fakat uyumu ne kadar yüksek olsa da seçilmeme şansı bulunmaktadır. Popülasyonda yer alan bireyler uyumluluklarına göre bir çarkın (tekerleğin, dairenin) dilimlerini oluşturacak farz edersek bu tekerlek döndürüldüğünde uyumu fazla olan birey daha fazla alan kapladığı için gelme olasılığı yüksek olacaktır [15].

Rulet tekerleği yönteminde Şekil 2.4'den anlaşılacağı üzere ilk başta popülasyondaki tüm bireylerin ayrı ayrı uygunluk değerleri bir kenara yazılır ve daha sonra bunlar toplanarak toplam uyum yani bizim %100 bulunur.

$$\text{Top\_UF} = \sum_{i=1}^{n} \text{UF(i)} \tag{2.6}$$

Top_UF = popülasyondaki uygunluk fonksiyonları toplamı

UF(i) = i. Bireyin uygunluk fonksiyonu

o halde i. bireyin seçilme olasılığı;

$$\text{iht(i)} = \frac{\text{UF(i)}}{\text{Top\_UF}} \tag{2.7}$$

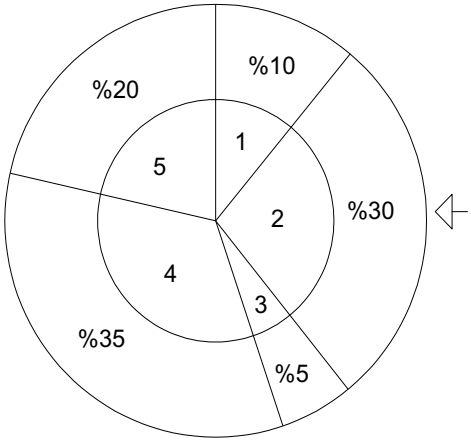
olacaktır. Örnek verecek olur isek;

		uyumu	
1.birey	00100	50	%10
2.birey	01001	150	%30
3.birey	10011	25	%5
4.birey	11001	175	%35
5.birey	01110	100	%20

Toplam	Toplam
500	%100

Şekil 2.4. Rulet Tekerleği Uygunluk Oranları

Şekil 2.5'de Şekil 2.4'de verilen rulet tekerleği uygunluk oranın dikkate alınarak rulet çemberine yerleştirilmesi gösterilmektedir. Beş farklı birey beş farklı oran da tekerleğe yerleştirilmiştir.



Şekil 2.5. Uygunluk Oranlarının Rulet Tekerleğin Yerleştirilmesi

Tekerlek oluşturulmadan önce yeni popülasyonu oluşturmak için elitizim yapıldı ise elitizimde seçilen birey sayısı kadar az, yapılmadı ise popülasyondaki birey sayısı kadar döndürülerek, her dönmede seçilen bireyler diğer genetik algoritma adımlarını uygulamak üzere eşleştirme havuzu (mating pool) denilen yerde toplanılmak koşulu ile ki bu havuzdan daha sonra ebeveyn çiftleri seçilecek yeni popülasyon oluşturulur. Tekerleğin döndürülmesi ile seçilen bireylere ebeveyn diyoruz ki bunlar eşleşerek daha sonra çocukları oluşturacaklar.

Bu yöntemin dezavantajı; eğer popülasyondaki uygunluk oranları çok farklı ise örneğin en iyi kromozom tekerleğin alanının %90 nını kaplıyor ise, diğer kromozomları seçilme şansı az olacak böylece diğerleri seçim sırasında yok olmuş olacaktır [16].

Bu seçilme en eski bayaslanmış rulet tekerleği (biased roulette Wheel) tekniği olarak kullanılmıştır.

Turnuva seçimi (Tournament): Bu seçim metodunda; ilk başta popülasyon içerisinde belirli sayıda bireyi içerisinde barındıran rastgele hazırlanmış popülasyon grupları seçilir. Daha sonra bu alt popülasyon içerisindeki bireyler uygunluk fonksiyonuna olan uyumları karşılaştırılmak koşulu ile yarıştırılırlar. Kalitesi yüksek olan bireyler alt popülasyondan çıkarak yeni popülasyonu oluşturmak için eşleştirme havuzuna

giderlerken, ebeveynleri oluşturmak için alt popülasyonda kalan bireyler diğerlerine dağılarak kendilerine seçilme şansı aramaya devam ederler. Bu işleme popülasyonun büyüklüğü kadar devam edilir. Bu yöntemin diğer yöntemlerden üstünlüğü; seçim esnasında uygunluk değeri düşük olan bireylerini yok olma ihtimallerinin düşük olmasıdır [16-19].

Sıralama seçimi: Bu yöntemde ilk önce popülasyonda yer alan bireyler, uygunluk değerleri iyiden kötüye doğru olacak şekilde sıralanır. Yani popülasyonda N adet birey var ise sıralama yapılırken en iyi uyumu gösteren birey birinci sırada yer alırken en sondaki birey ise en kalitesiz birey olacaktır. Daha sonra belirli bir kalite seviyesini geçen bireyler sıralamaya dikkat ederek ebeveyn çiftlerimiz oluşturulur.

Sıralama seçimi yönteminin avantajı; bütün kaliteli ya da kalitesiz ayrımı yapmaksızın, bütün bireylerin seçilme şansının olmasıdır. Bunu sayesinde uyumluluğu fazla bireylerin hüküm sürdüğü tekdüze bir popülasyonun oluşması engellenmiş olur. Dezavantajı ise; bizi çözüme götürme hızının düşük olmasıdır, çünkü en iyi bireyler diğerlerinden çok da farklı olmayacaktır [16-19].

Sabit durum seçimi: Bu yöntemde de yine bireylerimizin uygunluk değerlerine yani kalitelerine göre işlem yapılır lakin diğer seçim yöntemlerinden farklı olarak popülasyonun tamamını değiştiren bir seçim yöntemi değildir. İlk önce uygunlukları iyi olan birkaç adet birey popülasyon içerisinden yeni bireyler oluşturmak için seçilirler. Ardından uygunlukları kötü olan bireyler popülasyondan atılarak bunların yerine bir önceki adımda seçilen kaliteli bireyler yerleştirilir. Popülasyonun geri kalanıyla oynanmadan yeni nesile aktarılır [16-19].

2.4.2.3. Çaprazlama operatörü (gen takası)

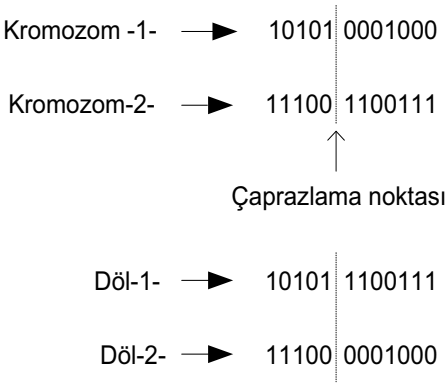
En basit anlatımı ile çaprazlama; iki ebeveyn kromozomlarının belirli parçalarının değiştirilerek (takas edilerek) yeni bireylerin (offspring) elde edilmesi olayıdır. Çaprazlama sayesinde çeşitlilik artar ve de ata bireylerin iyi özelliklerinin yeni nesile aktarılmasını sağlayarak daha iyi nesiller üretilir. Bu özelliği sayesinde genetik algoritma yapısının en önemli operatörü olarak görülür [26].

Seçim operatörünün işletimi sırasında, oluşturulan eşleştirme havuzlarından daha sonraki genetik algoritma operatör sistemlerinin işletimi sırasında kullanılmak üzere yeni ebeveyn çiftleri seçilir. Çaprazlama operatörünün işletimi sırasında ise, çaprazlama oranın gösterdiği kadar, bu oran yeni nesildeki bireylerin oluşturulması için çaprazlanması gereken ebeveyn çiftlerinin hangi oranda çaprazlanacağını gösteren bir orandır, havuzdan seçilen bireyler çaprazlanarak melez bir yeni nesil elde edilmiş olur. %0 çaprazlama oranına sahip bir popülasyonda çaprazlama olmaz iken %50 oranına sahip popülasyonda ise yeni neslin yarısı çaprazlama sonucunda oluşacaktır [16-19].

Çaprazlama işlemi dört farklı şekilde yapılabilmektedir;

Tek noktalı çaprazlama (single crossover): Öncelikle ebeveynler üzerinde rastgele bir çaprazlama noktası seçilir. Daha sonra bu noktaya kadar olan kısım birinci ebeveynden, geriye kalan kısım ise ikinci ebeveynden alınmak üzere çaprazlama yapılarak yeni kromozom dizileri elde edilmiş olur.

Şekil 2.6'da iki ebeveyn çiftinin tek bir yerden çaprazlama noktası seçilerek, bu noktadan sonra kalan kısımlarının yer değiştirilmesi ile oluşturulan yeni bireylerin oluşumu gösterilmiştir.

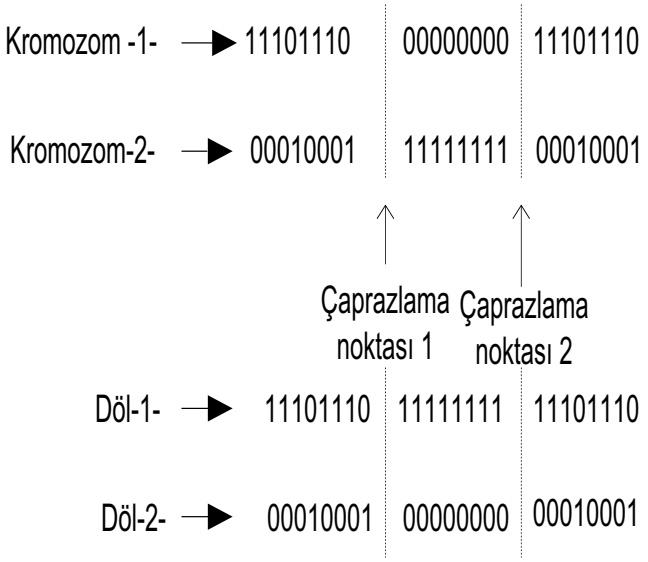


Şekil 2.6. Çaprazlama

İki noktalı çaprazlama (Two point): Bu çaprazlama tekniğinde ise eşleşme havuzundan seçilen ebeveynler üzerinden rastgele iki adet çaprazlama noktası seçilir. Seçilen bu iki çaprazlama noktası arasındaki genler ebeveynler arasında yer

değiştirilir. Yani bu yeni oluşturulan bireyler bu iki çaprazlama noktası arasında kalan bölüm dışında aynı kalır.

Örnek olarak;

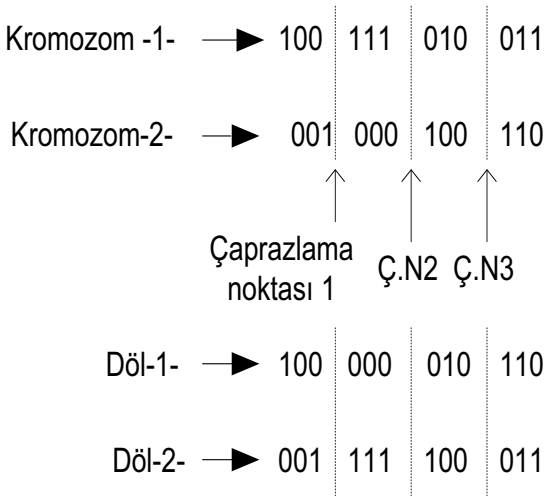


Şekil 2.7. İki Noktalı Çaprazlama

Şekil 2.7'de iki ebeveyn çiftinin iki farklı yerden çaprazlama noktası seçilerek, bu iki nokta arasının yer değiştirilmesi ile oluşturulan yeni bireylerin oluşumu gösterilmiştir.

Çok noktalı çaprazlama: İki noktalı çaprazlama yönteminin daha gelişmiş şeklidir. Bu yöntemde ikiden fazla çaprazlama noktası rastgele belirlenerek, eşleştirme havuzu içerisinden seçilen ebeveynler arasında noktalar arasını bir seçip bir atlamak koşulu ile gen takası yapılır.

Örnek olarak;



Şekil 2.8. Çok Noktalı Çaprazlama

Şekil 2.8'de iki ebeveyn çiftinin birden fazla yerden çaprazlama noktası seçilerek, bu noktalar arasında kalan kısımlarının yer değiştirilmesi ile oluşturulan yeni bireylerin oluşumu gösterilmiştir.

Tek biçimli çaprazlama (Uniform): Tek biçimli çaprazlama operatörü yardımı ile (mixing ratio) hangi ebeveynden hangi oranda kromozom yeni oluşturulacak bireylere geçeceği belirlenir.

Bu oranı 0.5 kabul eden bir çaprazlama ya örnek verilir ise;

$$\text{ebeveyn1} = 1101100100100010 10$$
$$\text{ebeveyn2} = 1001000101100111 00$$

0.5 olması yeni oluşacak bireylerin bitlerinin yarısını ebeveyn1 den yarısını da ebeveyn2 den alması anlamına gelir.

$$\text{döl1} = 1_1 0_2 0_2 1_1 1_1 0_2 0_2 1_2 0_1 0_1 1_2 0_2 0_1 0_1 1_1 1_2 0_2 0_1$$
$$\text{döl2} = 1_2 1_1 0_1 1_2 0_2 0_1 0_1 1_1 0_2 1_2 1_1 0_1 0_2 1_2 1_2 0_1 1_1 0_2$$

1 ve 0 ın altına yazılmış olan 1 ve 2 numaraları hangi bireyden alınmış olduğunu gösterir.

Yukarıdaki örneğe bakıldığı zaman 9 bitin ebeveyn 1 den 9 bitin ise ebeveyn 2 den alınmış olduğu görülür. Böylelikle 0.5 lik mixing oranını sağlamış olur.

Yukarıda verilen örnekte yeni oluşturulan bireylerdeki hangi bitlerin hangi ebeveynlerden geleceğini rastgele seçilmiştir. Bunu önlemek ve daha da pratiklik katmak için çaprazlama maskesi (yönetici) takımı oluşturulmuştur. Bu maske çaprazlamayı rastgele olarak yapmamıza yardımcı olur. Çaprazlama maskesi popülasyonda ki kromozom sayısına eşit sayıda biti bünyesinde bulundurur. Çaprazlama maskesi takımı her ebeveyn çifti için ayrı olmalıdır. 5 ebeveyn çifti yani 10 kromozom bulunan eşleştirme havuzu için biz 5 adet farklı çaprazlama maskesi hazırlanmalıdır. Çaprazlama maskesi kromozomlarında bit numarası 1 ise döl için birinci ebeveynden 0 ise ikinci ebeveynden o bit numarasındaki bit kopyalanır. İkinci döl için ise bu olayın tam tersi geçerlidir. Bunu bir örnekle açıklayalım;

Çaprazlama maskesi=	1	0	0	1	0	1
Ebeveyn_1_=	0	1	0	0	0	1
Ebeveyn_2_=	1	0	1	1	0	0

Döl_1_=	0	0	1	0	0	1
Döl_2_=	1	1	0	1	0	0

Şekil 2.9. Çaprazlama Sonucunda Meydana Gelen Dölleri

Şekil 2.9'da önceden oluşturulan çaprazlama maskesi dikkate alınarak, çaprazlama maskesi bit değeri 1 ise döl1 bit değerini birinci ebeveynden, 0 ise ikinci ebeveynden alarak (döl2 için de tam tersi geçerlidir), ebeveyn çiftlerinden yeni bireylerin oluşumu gösterilmektedir.

Aritmetik çaprazlama (Arithmetic): Aritmetik çaprazlama iki ebeveyni aşağıda verilen eşitliklere göre lineer şekilde eşleştirerek yeni döllerin bitlerini oluşmasını sağlar.

$$\begin{aligned} döl1 &= a*ebeveyn1+(1-a)*ebeveyn2 \\ döl2 &= (1-a)*ebeveyn1+a*ebeveyn2 \end{aligned} \tag{2.8}$$

a=çaprazlama sabitidir (weighting factor) ve her çaprazlama olayı için rastgele olacak şekilde farklı seçilir[26].

Örnek olarak;

Ebeveyn-1-	0.3	1.4	0.2
Ebeveyn-2	0.5	4.5	0.1
a=0.7			

Şekil 2.10. a. Ebeveyn Örnekleri

$döl1_1$=0.7*0.3+(1-0.7)*0.5
$döl2_1$=(1-0.7)*0.3+0.7*0.5

Döl-1-	0.36	2.33	0.17
Döl-2-	0.4	2.98	0.15

Şekil 2.10. b. Meydana Gelen Döller

Şekil 2.10'da a çaprazlama sabitinin ve ebeveyn bir ve iki değerlerinin formül 2.8'de yerleştirilmesi üzerine oluşan yeni bireylerin değerleri görülür.

2.4.2.4. Mutasyon operatörü (Mutation)

Genetik algoritmanın en önemli operatörlerinden biri olan mutasyon operatörünün amacı, ebeveynlerden üreyen yeni bireylerin farklılığını arttırarak uygunluk fonksiyonu ile olan uyumun bir önceki jenerasyonla aynı olmasını engellemek, uygunluğunu iyileştirdiği içinde çözüme daha genetik algoritmanın daha hızlı yakınsamasını sağlamaktır.

Mutasyonu kısaca anlatacak olur isek, çaprazlama işlemi sonucu oluşan yeni kromozom çiftlerinin rastgele seçilmiş bir ya da birden fazla bitinin kendi içerisinde değiştirilmesi işlemidir. Bu değiştirilme işleminde 0 olan bit 1 yapılırken 1 olan bit ise 0 yapılır. Mutasyon sonucunda oluşan yeni bireyler artık ebeveynlerin birer kopyası olmaktan uzaklaşacaktır.

Kromozom içerisindeki bitlerin ne kadarının mutasyona uğrayacağını belirleyen parametre mutasyon oranıdır (mutation rate). %0 mutasyon oranına sahip bir

algoritmada yeni bireylerin hiçbir kromozomu değiştirilmezken, %100 olan bir algoritmada bütün bitler değişime uğrayacaktır. Araştırmalar en sağlıklı oranın %1 ile %5 arasında olduğu sonucun ortaya koymuştur. Bu yeni bireyin bit sayısının %1 ile %5 nin değişeceği anlamına gelir.

Üreme ve çaprazlama olayları sırasında popülasyondaki bireyler değişmiyor ve üstüne üstelik bu algoritmada bir de mutasyon operatörü yok ise böyle bir algoritmanın yakınsaması yalnızca uygunluk fonksiyonuyla uyumlu bir başlangıç popülasyonuna sahip olmasıyla mümkündür. Mutasyon operatörü bulunmayan bir genetik algoritmanın verimli çalışması ancak başlangıç popülasyonunun büyük tutulmasıyla mümkün olabilir ki bu zaman kaybı olacaktır. Ayrıca mutasyon operatörünün diğer bir yararı da daha önceki jenerasyon işlemleri sırasında atılmış iyi kromozomların tekrar üreyebilmesine olanak sağlamasıdır [15-24].

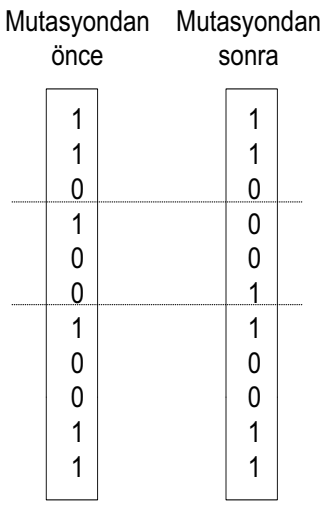
Bazı mutasyon operatörlerine örnek verilir ise,

Mutasyondan önceki	Birey 1	10110010111
	Birey 2	11011011001
Mutasyondan sonraki	Birey 1	10111010111
	Birey 2	11011010001

Şekil 2.11. Mutasyon Örnekleri

Şekil 2.11'de çaprazlama operatörü sonrası oluşturulan yeni bireylerin mutasyona uğrayarak birinci birey için beşinci bit değerinin, ikinci birey için ise sekizinci bir değerinin değişimi anlatılmaktadır.

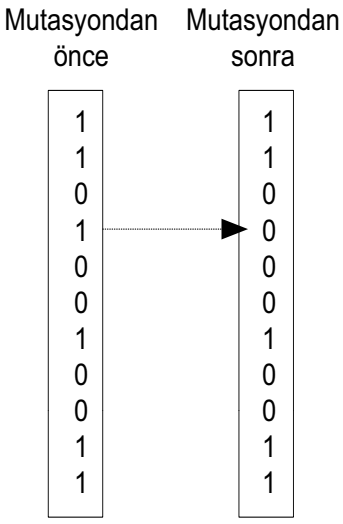
Ters çevirme: Bu mutasyon işleminde mutasyonun yapılacağı kromozom üzerinde rastgele gen grupları seçilir ve daha sonra seçilen bu gen gruplarındaki kromozom dizilimleri ters çevrilir.



Şekil 2.12. Mutasyon Önce ve Sonrası

Şekil 2.12'den görüleceği üzere mutasyonun yapılmasının istenildiği birey üzerinde rastgele bir kromozom parçası seçilir ve daha sonra bu kromozom parçası ters çevrilerek yeni bireyin oluşması sağlanır.

Ekleme: Ekleme mutasyon işleminde popülasyonun içine yerleştirilen genlerin rastgele yerleştirilmesidir. Ekleme yapılan yerdeki bit 0 ise 1, 1 ise 0 değerini alır.



Şekil 2.13. Ekleme İşlemi

Şekil 2.13'de seçilen bireyin ekleme mutasyon işlemi ile yeni birey haline getirilmesi işlemi anlatılmıştır. Bu işlem sırasında seçilen dördüncü bit değeri 1 iken 0 haline dönüştürülmüştür.

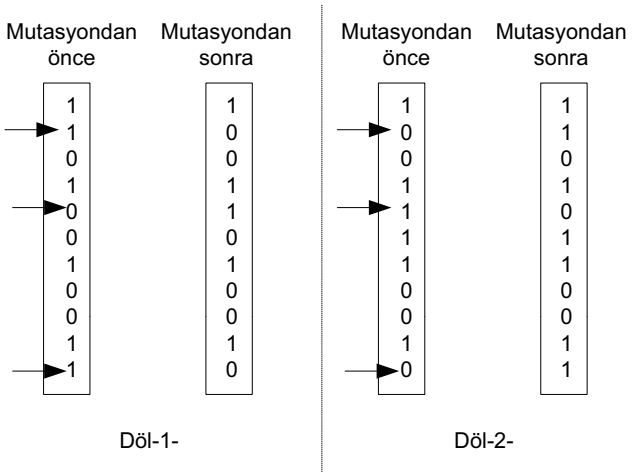
Yer değişikliği: Bu işlemde popülasyon içerisinde çaprazlamadan sonra oluşturulan bazı kromozom gruplarının (döller) içerisinden bit dizileri seçilir ve bunlar döller arasında karşılıklı değiştirilir.

Mutasyondan önceki	Birey 1	10110010111
	Birey 2	11011011001
Mutasyondan sonraki	Birey 1	10111010111
	Birey 2	11010010111

Şekil 2.14. Yer Değiştirme

Şekil 2.14'de seçilen birey çifti arasında kromozom değiş tokuşu ile mutasyon yapılması anlatılmaktadır. Seçilen çiftin dördüncü bitinden yedinci bitine kadar olan kısımları birbiri içerisinde yer değiştirilerek yeni bireyler elde edilmiştir.

Karşılıklı değişim: Bu işlemde popülasyon içerisinde çaprazlamadan sonra oluşturulan bazı kromozom gruplarının (döller) içerisinden bitler seçilir ve bunlar döller arasında karşılıklı değiştirilir.



Şekil 2.15. Karşılıklı Değişim

Şekil 2.15'de verilen örnekte mutasyondan önce seçilen iki,beş ve onbirinci bitlerin her iki ebeveyn arasında yer değiştirilmesi ile oluşturulan yeni bireyler gösterilmektedir.

BÖLÜM 3 GÜÇ AKIŞI

3.1. Giriş

Dünya popülasyonunun gün geçtikse artmakta ve de buna bağlı olarak enerjiye olan talep gün geçtikçe çoğalmaktadır. Enerji denilince aklımıza ilk olarak tabi ki elektrik enerjisi gelmektedir. Teknolojik gelişmeler de elektrik enerjisine olan taleple birlikte aynı hızda artmasına karşın hala enerji ihtiyacı tam olarak karşılanamamaktadır. İşte bu yüzden mevcut olunan enerji kaynaklarında ne iyi şekilde yararlanmak bir zorunluluk haline gelmiştir.

Güç akışı en basit tanımlaması şebekedeki gerilimleri ve güç akışını, sistemin kontrol değişkenleri yardımı ile, kontrol değişkenleri değerleri başlangıçta şebekenin yük tevzi merkezinden elde edilebilinir, bulunması demektir. Güç akışı problemlerinin çözümünde için doğrusal olmayan programlama, quadratik programlama, Newton tabanlı teknikler ve lineer programlama gibi farklı teknikler kullanılmıştır. Ancak bu metotlar yerel minimuma takılma ya da başlangıç noktası problemleri yaşadığından dolayı bugün daha çok bunlar yerine sezgisel metotlar kullanılmaktadır. Bu sezgisel metotlardan bazılarına; genetik algoritma (GA), karınca kolonisi algoritması, parçacık sürü optimizasyonu, evrimsel programlama ve diferansiyel evrim algoritmaları örnek gösterilebilinir. Bu yöntemler özellikle zaman tasarrufu sağladıklarından dolayı büyük avantaj sağlarlar [25-27].

Güç akışı algoritmaları sonucunda elde edilen bilgiler;

- her baradaki gerilim genliği (generatör baraları değerleri başlangıçta biliniyor),
- her baradaki faz açısı (slack bara değeri başlangıçta biliniyor),,
- her hatta akan aktif ve reaktif güçlerdir.

Farklı güç sistemlerinin birbirleri aralarında bağlanması sonucu oluşan enterkonnekte sistemler,

$$[A][x]=[b] \quad (3.1)$$

şeklinde ki nonlineer bir denklem takımıyla ifade edilirler. Bu denklem sistemlerinin çözüm süreci 1929'lar da analog hesap makinelerinin icadı ile 1956'lardan sonra ise bilgisayarların icadı ile farklı boyutlar kazanmıştır. Bilgisayarlar karmaşık şebekelerin incelenmesini daha da kolay ve hızlı hale getirmiştir. Sistemlerin bilgisayarlara tanıtılmasında bugüne kadar birçok farklı yöntem keşfedilmiş ve bu keşfedilme süreci bugün sezgisel metotların da gelişimiyle halen devam etmektedir [25].

Güç akışı analizleri yapılırken hem şu anda sistemimizde bulunan üreticilerin en iyi şekilde işletilmesi hem de teknolojik gelişmelerle paralel şekilde elimizde bulunan sistemlerin geliştirilmesi hedeflenir. Sistemde mevcut olan her baraların geriliminin genliğinin, faz açısının, aktif ve reaktif gücünün takipleri buna yardımcıdır.

3.2. Güç Akışı Problemi

Güç akışı problemi, şebekedeki gerilim değerlerini bulmaktır. Bu değerleri bulmak için önceden bilinmesi gereken değerler şebekenin yük tevzi merkezinden alınır. Bu değerler; yük baralarında ki aktif ve reaktif güç değerleri, şebekenin aktif güç ihtiyacının santral arasında nasıl dağıldığı ve santralin gerilim genlikleridir. Bunlar veri olarak adlandırılır. Bilinmeyen değerlerimiz ise baralardaki karmaşık gerilim değerleri, santralin reaktif güç üretimleri ve çekilen gücün hatlara ne şekilde dağıtıldığıdır. Daha sonra bara ve hatlardaki bilinen ve bilinmeyen değerler düğüm gerilimler yöntemi kullanarak devre çözümlenir. Çözümleme sonunda sistemimizdeki yer alan tüm baraların gerilim genlik değerleri ve faz açıları, iletim hatlarındaki aktif ve reaktif güçler ve de sisteme ilişkin toplam aktif ve reaktif güç kayıpları bulunur [25-27].

3.3. Güç Akışı Kabulleri

Güç akışı dengeli, üç fazlı, sürekli hal koşulları altında yapılır. Gerçekte üç fazlı dağıtım sistemlerinde hatlarda yer altı kabloları kullanılması nedeniyle yüksek R/X oranı, hatlarda fazlar arasında çaprazlama yapılmaması nedeniyle dengesiz hat

empedansı, fazların ayrı yükleri beslemesi ve tek veya iki fazlı hatlarla elektrik enerjisinin dağıtılması nedeniyle dengesiz yüklenme gibi nedenler yüzünden farklı karakteristiklere sahip oluşları bu kabullerin yapılmasının nedenlerinden bazılarıdır [25].

Yapılan bu kabuller;

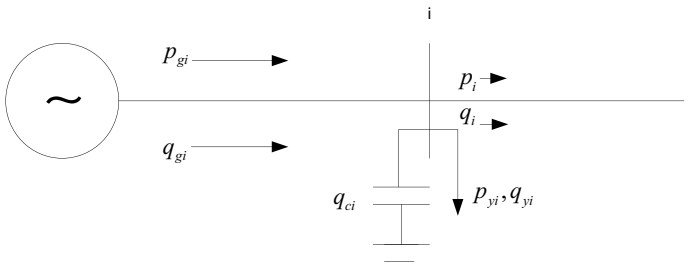
- Jeneratörler sistemdeki tüm yük taleplerini ve iletim hatlarında meydana gelen toplam güç kayıplarını karşılamaktadır.
- Bütün baraların gerilim genlikleri nominal gerilim sınırları civarındadır.
- Jeneratörlerin ürettikleri aktif ve reaktif güçler sınırları aşamaz.
- İletim hatları ve transformatörler aşırı yüklenemez [25].

3.4. Güç Akışı Metotları

Güç akışı eşitlikleri nonlineer denklemler olduğu için denklemlerin çözümünde lineer olmayan metotlar kullanılmalıdır. Gauss eliminasyon metodu, Gauss-Seidal veya Newton Raphson metodu bunlardan bazılarıdır.

3.5. Güç Akışı

Toplamda N baralı olan bir enterkonnekte sistemde i. baradan sisteme verilen güç;



Şekil 3.1. Bir Baranın Genel Amaçlı Gösterimi

p_{gi},q_{gi} = jeneratörün ürettiği aktif ve reaktif güçler

p_{yi},q_{yi} =i ninci baraya bağlı yükün çektiği aktif ve reaktif güç

p_i,q_i =i ninci baraya bağlı diğer hatlara verilen aktif güç

q_{ci}=i ninci baraya bağlı (şönt) reaktif güç üreteci. Bu değerin içine güç katsayısını düzeltmek için dışarıdan eklenecek kapasiteler veya senkron makineler v.s dahil edilebilinir.

baraya Kirchoff yasasını uygulanır ise;

$$p_{gi}=p_{yi}+p_i \quad (3.2)$$

$$q_{gi}+q_{ci}=p_{yi}+p_i \quad (3.3)$$

$$p_i-(p_{gi}-p_{yi})=p_i-p_{hati}=g_{pi}=0 \qquad (i=2,\ldots,n) \qquad (3.4)$$

$$q_i-(q_{ci}-q_{yi})=q_i-q_{hati}=g_{qi}=0 \qquad (i=ng+1,\ldots,n) \qquad (3.5)$$

ng= sistemdeki toplan jeneratör sayısı

Denklemleri ortaya çıkar. Gerçekte de güç akışı algoritması bu denklemleri sağlar.

Sistemde yer alan baralarda güç akışı boyunca dört adet değişken yer alır; $p_{hati}, q_{hati}, v_i, \delta_i$. Bu değişkenlerden her bara için iki tanesi bilinirken iki tanesinin değeri aranır. Ayrıca bu değişkenlerin hangilerinin bilinip hangilerinin aranmasına göre sistemdeki baralar çeşitlenir. Örneğin, salınım barası olarak tabir edilen referans ya da gevşek barada bu dört değişkenden v_i ve δ_i değerleri bilinir p_{hati} ve q_{hati} değerleri aranır. Salınım barasının p_{hati} ve q_{hati} değerlerinin bilinmeyişinin nedeni sistemdeki hat kayıplarının önceden bilinememesidir. Salınım barasının aktif gücü bilinmeyen seçilerek güç akışı sonunda sistemin aktif ve reaktif güç değerleri bulunduktan sonra bu baranın değerleri bulunur ve de sisteme ait güç dengesi böylelikle sağlanmış olur (üretilen=tüketilen+kayıp)[38]. Jeneratör barasında ya da diğer adıyla gerilim kontrollü barada (PV barası) p_{hati} ve v_i değerlerinin bilinir, δ_i ve q_{hati} değerlerini aranır. Bu barada yer alan jeneratörün p_{gi} si yani p_{hati} si türbin karakteristikleri, v_i ise makine uyarma sargılarını kontrol eden gerilim regülatörleri yardımı ile ayarlanarak daha önce bilinen değerinde sabit tutulur. Ve son olarak da yük barası diye adlandırılan PQ bara türünde ise p_{hati} ve q_{hati} değerlerinin bilinir ve v_i, δ_i aranır [25-27].

i'ninci baradan sisteme verilen kompleks güç ise;

$$s_i = p_i + jq_i = v_i i_i^* \tag{3.6}$$

i_i^* =i. bara akım fazörünün eşleniği

$$i_i^* = \sum_{j=1}^{n} y_{ij}^* v_j^* \tag{3.7}$$

$$\underset{\substack{\text{kompleks}\\ \text{değer}}}{v_i} = v_i(\cos\delta_i + j\sin\delta_i) \tag{3.8}$$

δ_i =salınım barasına göre (referans bara) i. baranın gerilim faz açısıdır.

y_{ij} =bara admitans matrisi $[y_{bara}]$

$$y_{ij} = g_{ij} + jb_{ij} \tag{3.9}$$

Bütün bu denklemler kompleks güç denkleminde yerine konulur ise;

$$s_i = v_i \sum_{j=1}^{n} v_j(g_{ij}\cos\delta_{ij} + b_{ij}\sin\delta_{ij}) + jv_i \sum_{j=1}^{n} v_j(g_{ij}\sin\delta_{ij} - b_{ij}\cos\delta_{ij}) \tag{3.10}$$

i=1,…,n

$$\delta_{ij} = \delta_i - \delta_j \tag{3.11}$$

δ_{ij} =i ve j ninci bara gerilim fazörleri arasındaki fark

Denklem 3.10'da gösterildiği gibi sanal ve reel kısımlar ayrılıp Denklem 3.6 ile eşleştirilirse ise bu durumda ortaya;

$$p_i = v_i \sum_{j=1}^{n} v_j(g_{ij}\cos\delta_{ij} + b_{ij}\sin\delta_{ij}) \tag{3.12}$$

$$q_i = v_i \sum_{j=1}^{n} v_j(g_{ij}\sin\delta_{ij} - b_{ij}\cos\delta_{ij}) \tag{3.13}$$

görüldüğü üzere güç eşitlikleri lineer olmayan denklemlerdir.

3.6. Kontrol ve Durum Değişkenleri

Kısaca özetleyecek olur isek güç akışı hesaplamaları boyunca iki çeşit değişken mevcuttur. Güç akışının temel mantığında kontrol değişkenleri güç akışı boyunca sabit bir değerde tutularak sistemin durum değişkenleri aranır. Sistemimizdeki kontrol değişkenlerimiz; jeneratör baralarının gerilim genlikleri (v_{gi}), ayarlanabilen reaktörlerin reaktif güç değerleri (q_{ci}), sürekli ayarın yapıldığı transformatörlerin kademe ayar değerleri (t_i), iken durum değişkenlerimiz ise; salınım barası hariç tüm baraların gerilim açı değerleri (δ_i), yük baralarının gerilim genlikleri (v_i) dir [25].

3.7. [Ybara]'nın Bulunuşu

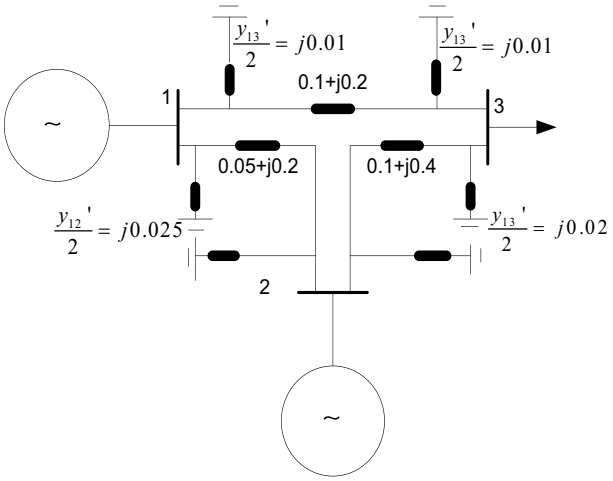
Denklem 3.7'den da görüldüğü üzere güç akışı boyunca bara admitans matrisi kullanılmaktadır[37].

$$y_{ij} = g_{ij} + jb_{ij} \Rightarrow [Y_{BARA}] = [Z_{BARA}]^* \qquad (3.14)$$

bara admitans matrisi

$[Z_{BARA}]$ matrisi kısa devre analizleri için daha uygun olduğundan dolayı güç akışında $[Y_{BARA}]$ matrisi kullanılır. Sistemin tek hat diyagramından hareketle ve iletim hatlarının seri empedansları ve şönt admitansları dikkate alınarak Y_{BARA} matrisi elde edilebilir.

Y_{BARA}' nın hesaplanmasına örnek verecek olur isek;



Şekil 3.2. Üç Baralı Bir Sistem Örneği

$$[Y_{BARA}]=\begin{bmatrix} y_{11} & y_{12} & y_{13} \\ y_{21} & y_{22} & y_{23} \\ y_{31} & y_{32} & y_{33} \end{bmatrix}$$

$$y_{11}=y_{12}+y_{13}+\frac{y_{12}'}{2}+\frac{y_{13}'}{2}=>$$

$$y_{11}=(0.05+j0.2)^{-1}+(0.1+j0.2)^{-1}+(j0.025)+(j0.01)=>$$

$$y_{11}=3.176-j8.68b$$

$$y_{12}=-z_{12}^{-1}=-(0.05+j0.2)^{-1}=-1.176+j4.71b$$

$$y_{13}=-z_{13}^{-1}=-(0.1+j0.2)^{-1}=-2+j4\,b$$

$$[Y_{BARA}]=\begin{bmatrix} 3.176-j8.67 & -1.176+j4.71 & -2+j4 \\ -1.176+j4.71 & 1.765-j7.014 & -0.588+j2.35 \\ -2+j4 & -0.588+j2.35 & 2.588-j6.323 \end{bmatrix} b$$

3.8. Metotlar

Hangi metodu seçeceğimizi belirlerken, iterasyon sayısı yani vereceği zaman tasarrufu göze alınır. Aşağıda en sık kullanılan iterasyon metotlarından iki tanesinden bahsedilecektir [25].

3.8.1. Gauss-Seidal Metodu

Güç akışı analizlerinde bir takım zorluklar ile karşılaşılır. Bu zorluklar değişik baralar için tarif edilen dataların farklılığından ileri gelir. Güç akış problemlerinin çözümleri bilinmeyen bara gerilimlerine tahmini değerler verip hesaplanan aktif ve reaktif güçler ve baralardaki tahmini değerlerden her bara için yeni bir gerilim değeri hesaplanmak ile yapılır.

Her bara için hesaplanan yeni gerilim değerleri diğer bara gerilimlerinin hesaplanmasında kullanılır. Bu olaya iterasyon adı verilir. İterasyon işlemi, bütün baralardaki değişimlerin tarif edilen minimum değerden küçük oluncaya kadar devam edilir [25].

Çözüme bara gerilim bağıntıları kurularak başlanır. Bara gerilimleri jeneratörlerden veya baralara bağlı yüklerden gelen aktif ve reaktif güçlerin fonksiyonudur. Yine bu bara gerilimleri tahmini olarak verilmiş veya daha önce diğer baralardan ve düğümlerin self ve karşılıklı admitanslarından hesaplanmıştır. Temel denklemlerin çıkarılmasına şebeke düğüm denklemleri ile başlanır. Önce dört baralı bir sistem için denklemler çıkarılacak, daha sonra denklemler genelleştirilecektir. Bir numaralı bara salınım barası olarak alınırsa, iki numaralı bara ile çözüme başlanır [25].

3.8.2. Newton-Raphson metodu

Diğer güç akışı metotlarına nazaran iterasyonu daha kısa sürede ve daha az iterasyonla tamamladığı için daha çok tercih edilmektedir. Newton-Raphson algoritmasında iterasyon sayısı ile sistemimizin boyutları farklıdır. Ayrıca bu algoritma türünde Taylor seri açılımlarından da faydalanılır [25].

Lineer olmayan fonksiyonlarımız;

$$p_i = v_i \sum_{j=1}^{n} v_j (g_{ij}\cos\delta_{ij} + b_{ij}\sin\delta_{ij}) \tag{3.12}$$

$$q_i = v_i \sum_{j=1}^{n} v_j (g_{ij}\sin\delta_{ij} - b_{ij}\cos\delta_{ij}) \tag{3.13}$$

(yukarıda nerden bulundukları verilmişti)

olmak üzere

$$[y] = \begin{bmatrix} f_1(x) \\ f_2(x) \\ \dots \\ f_n(x) \end{bmatrix} = \begin{bmatrix} P(x) \\ Q(x) \end{bmatrix} \tag{3.15}$$

eşitliği verilir. Bu eşitlikte ki fonksiyon birimlerini p_i ve q_i denklemleri doldurur. Bu eşitliklerdeki fonksiyonlardaki hataların sıfıra gitmesi için fonksiyonlar x_0 civarında

Taylor serisine açılırlar. En nihayetinde iki veya daha fazla değişkenli fonksiyonların Taylor serisine açılımı Newton-Raphson metodunun temelidir [25].

n değişkenli y=f(x) fonksiyonunun Taylor serisine açılımı;

$$y=f(x_0)+\left.\frac{\partial f}{\partial x}\right|_{x-x_0}.(x-x_0)+(BDBKT) \tag{3.16}$$

BDBKT=birinci dereceden büyük kısmi türevler. Bunlar ihmal edilir.

Buradan x'i eşitliğin sağına çekersek;

$$x=x_0+\left[\left.\frac{\partial f}{\partial x}\right|_{x=x_0}\right]^{-1}.(y-f(x_0)) \tag{3.17}$$

Bu denklemde x_0 yerine x(i) ve x yerine de x(i+1) yazılırsa;

$$x(i+1)=x(i)+\left[J(i)\right]^{-1}.(y-f\left[x(i)\right]) \tag{3.18}$$

$(y-f\left[x(i)\right])$ vektör matrisi daha önce bahsedilen;

$p_i-(p_{gi}-p_{yi})=p_i-p_{hati}=g_{pi}=0 \qquad (i=2,\ldots,n)$

$q_i-(q_{ci}-q_{yi})=q_i-q_{hati}=g_{qi}=0 \qquad (i=ng+1,\ldots,n)$

denklemleri yardımı ile hesaplanacaktır.

Her iterasyonda $x(i+1)-x(i)\le\varepsilon$ kontrolü yapılır ve bu eşitlik sağlanır ise algoritmamız sona erer. Eğer sağlanamaz ise son iterasyonda bulunan x değerleri yeni başlayacak olan iterasyonun başlangıç değerleri olarak atanırlar ve algoritma tekrar edilir [25].

J(i) olarak bahsi geçen matrisi Jacobian matristir ve n*n boyutlu bir matristir. Jacobian matrisi aktif ve reaktif güçlerin gerilim açısı ve genliğinin değişimlerini veren matristir. Bu yüzden kendi içinde birçok türevli denklemi barındırır.

$$[J(i)] = \left[\left. \frac{\partial f}{\partial x} \right|_{x=x(i)} \right] = \left[\begin{array}{ccc} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \cdots & \cdots & \cdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{array} \right]_{x=x(i)} \tag{3.19}$$

3.18 denkelmini daha açık bir biçimde yazacak olur isek;

$$[x]^T = \left[\delta_2 \quad \ldots \quad \delta_n \quad v_{ng+1} \quad \ldots \quad v_n \right] \tag{3.20}$$

$$[y]^T = \left[p_{hat2} \quad \ldots \quad p_{hatn} \quad q_{hat(ng+1)} \quad \ldots \quad q_{hatn} \right] \tag{3.21}$$

$$[f(x)]^T = \left[p_2 \quad \ldots \quad p_n \quad q_{(ng+1)} \quad \ldots \quad q_n \right] \tag{3.22}$$

Jacobian matrisin içini doldurmak için ise ana köşegen ve ana köşegen dışı elemanlara bağlı olarak bulunmuş mevcut formüller kullanılır;

$$\frac{\partial p_i}{\partial \delta_i} = J1(i,i) = v_i \sum_{\substack{j=1 \\ j \neq i}}^{n} v_j (-g_{ij} \sin\delta_{ij} + b_{ij} \cos\delta_{ij}) \tag{3.23}$$

$$\frac{\partial p_i}{\partial \delta_j} = J1(i,j) = v_i v_j (g_{ij} \sin\delta_{ij} - b_{ij} \cos\delta_{ij}) \tag{3.24}$$

$$\frac{\partial p_i}{\partial v_i} = J2(i,i) = \sum_{\substack{j=1 \\ j \neq i}}^{n} v_j (g_{ij} \cos\delta_{ij} + b_{ij} \sin\delta_{ij}) + 2 v_i g_{ii} \tag{3.25}$$

$$\frac{\partial p_i}{\partial v_j} = J2(i,j) = v_i (g_{ij} \cos\delta_{ij} + b_{ij} \sin\delta_{ij}) \tag{3.26}$$

$$\frac{\partial q_i}{\partial \delta_i} = J3(i,i) = v_i \sum_{\substack{j=1 \\ j \neq i}}^{n} v_j (g_{ij} \cos\delta_{ij} + b_{ij} \sin\delta_{ij}) \tag{3.27}$$

$$\frac{\partial q_i}{\partial \delta_j} = J3(i,j) = -v_i v_j (g_{ij} \cos\delta_{ij} + b_{ij} \sin\delta_{ij}) \tag{3.28}$$

$$\frac{\partial q_i}{\partial v_i} = J4(i,i) = \sum_{\substack{j=1 \\ j\neq i}}^{n} v_j (g_{ij}\sin\delta_{ij} - b_{ij}\cos\delta_{ij}) - 2v_i b_{ii} \tag{3.26}$$

$$\frac{\partial q_i}{\partial v_j} = J4(i,j) = v_i (g_{ij}\sin\delta_{ij} - b_{ij}\cos\delta_{ij}) \tag{3.30}$$

$$[J] = \begin{bmatrix} \frac{\partial p_2}{\partial \delta_2} & \frac{\partial p_2}{\partial \delta_3} & \cdots & \frac{\partial p_2}{\partial \delta_n} & \frac{\partial p_2}{\partial v_{ng+1}} & \frac{\partial p_2}{\partial v_{ng+2}} & \cdots & \frac{\partial p_2}{\partial v_n} \\ \frac{\partial p_3}{\partial \delta_2} & \frac{\partial p_3}{\partial \delta_3} & \cdots & \frac{\partial p_n}{\partial \delta_2} & \frac{\partial p_3}{\partial v_{ng+1}} & \frac{\partial p_3}{\partial v_{ng+2}} & \cdots & \frac{\partial p_3}{\partial v_n} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \frac{\partial p_n}{\partial \delta_2} & \frac{\partial p_n}{\partial \delta_3} & \cdots & \frac{\partial p_n}{\partial \delta_n} & \frac{\partial p_n}{\partial v_{ng+1}} & \frac{\partial p_n}{\partial v_{ng+2}} & \cdots & \frac{\partial p_n}{\partial v_n} \\ \frac{\partial q_{ng+1}}{\partial \delta_2} & \frac{\partial q_{ng+1}}{\partial \delta_3} & \cdots & \frac{\partial q_{ng+1}}{\partial \delta_n} & \frac{\partial q_{ng+1}}{\partial v_{ng+1}} & \frac{\partial q_{ng+1}}{\partial v_{ng+2}} & \cdots & \frac{\partial q_{ng+1}}{\partial v_n} \\ \frac{\partial q_{ng+2}}{\partial \delta_2} & \frac{\partial q_{ng+2}}{\partial \delta_3} & \cdots & \frac{\partial q_{ng+2}}{\partial \delta_n} & \frac{\partial q_{ng+1}}{\partial v_{ng+1}} & \frac{\partial q_{ng+2}}{\partial v_{ng+2}} & \cdots & \frac{\partial q_{ng+2}}{\partial v_n} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \frac{\partial q_n}{\partial \delta_2} & \frac{\partial q_n}{\partial \delta_3} & \cdots & \frac{\partial q_n}{\partial \delta_n} & \frac{\partial q_n}{\partial v_{ng+1}} & \frac{\partial q_n}{\partial v_{ng+2}} & \cdots & \frac{\partial q_n}{\partial v_n} \end{bmatrix} \tag{3.31}$$

$$[J] = \begin{bmatrix} J1 & J2 \\ J3 & J4 \end{bmatrix} \tag{3.32}$$

$$[J(i)].\Delta x(i) = \Delta y(i) \tag{3.33}$$

$$x(i+1) = x(i) + \Delta x(i) \tag{3.34}$$

İterasyon $|\Delta y_k(i)| < \varepsilon, k=1,2,...,n$ denklemini sağlayıncaya kadar devam eder.

$$\Delta x(i) = \begin{bmatrix} \Delta\delta(i) \\ \Delta V(i) \end{bmatrix} = \begin{bmatrix} \delta(i+1) - \delta(i) \\ V(i+1) - V(i) \end{bmatrix} \tag{3.35}$$

$$\Delta y(i) = \begin{bmatrix} \Delta P(i) \\ \Delta Q(i) \end{bmatrix} = \begin{bmatrix} P - P[x(i)] \\ Q - Q[x(i)] \end{bmatrix} \tag{3.36}$$

P ve Q baralara basılan net aktif ve reaktif güç değerleridir.

O halde;

$$\begin{bmatrix} J1(i) & J2(i) \\ J3(i) & J4(i) \end{bmatrix} \begin{bmatrix} \Delta\delta(i) \\ \Delta V(i) \end{bmatrix} = \begin{bmatrix} \Delta P(i) \\ \Delta Q(i) \end{bmatrix} \tag{3.37}$$

$$x(i+1) = \begin{bmatrix} \delta(i+1) \\ V(i+1) \end{bmatrix} = \begin{bmatrix} \delta(i) \\ V(i) \end{bmatrix} + \begin{bmatrix} \Delta\delta(i) \\ \Delta V(i) \end{bmatrix} \tag{3.38}$$

Yukarıda anlattığımız Newton-Raphson algoritması sona erdiğinde program x durum değişkenleri vektörünü verir. Lakin x vektörü gerilim genlik ve açılarından oluşmaktadır. Lakin PV baralarının ve salınım barasının ürettiği reaktif güçleri ve salınım barasında üretilen aktif gücü kayıp güçleri bulunmak isteniyor ise, durum değişkenleri vektöründe bulunan değerler aşağıdaki denklemlerde yerine konularak bulunur.

Reaktif güçler için[37];

$$q_i = v_i \sum_{j=1}^{n} v_j (g_{ij} \sin\delta_{ij} - b_{ij} \cos\delta_{ij}) \tag{3.39}$$

Salınım barasının ürettiği aktif gücü bulmak için ise;

$$p_1 = v_1 \sum_{j=1}^{n} v_j (g_{1j} \cos\delta_{1j} + b_{1j} \sin\delta_{1j}) \tag{3.40}$$

formülleri kullanılır. Daha sonra hatlardaki kayıp gücü;

$$\sum_{i=1}^{ng} p_{gi} - \sum_{i=1}^{n} p_{yj} = \text{toplam aktif güç kaybı}$$

İki bara arasında akan aktif ve reaktif güç değerlerini ise;

$$s_{ij} = p_{ij} + q_{ij} = v_i (v_i^* - v_j^*) y_{ij}^* + v_i v_i^* \left(\frac{y_{ij}'}{2}\right)^* \tag{3.41}$$

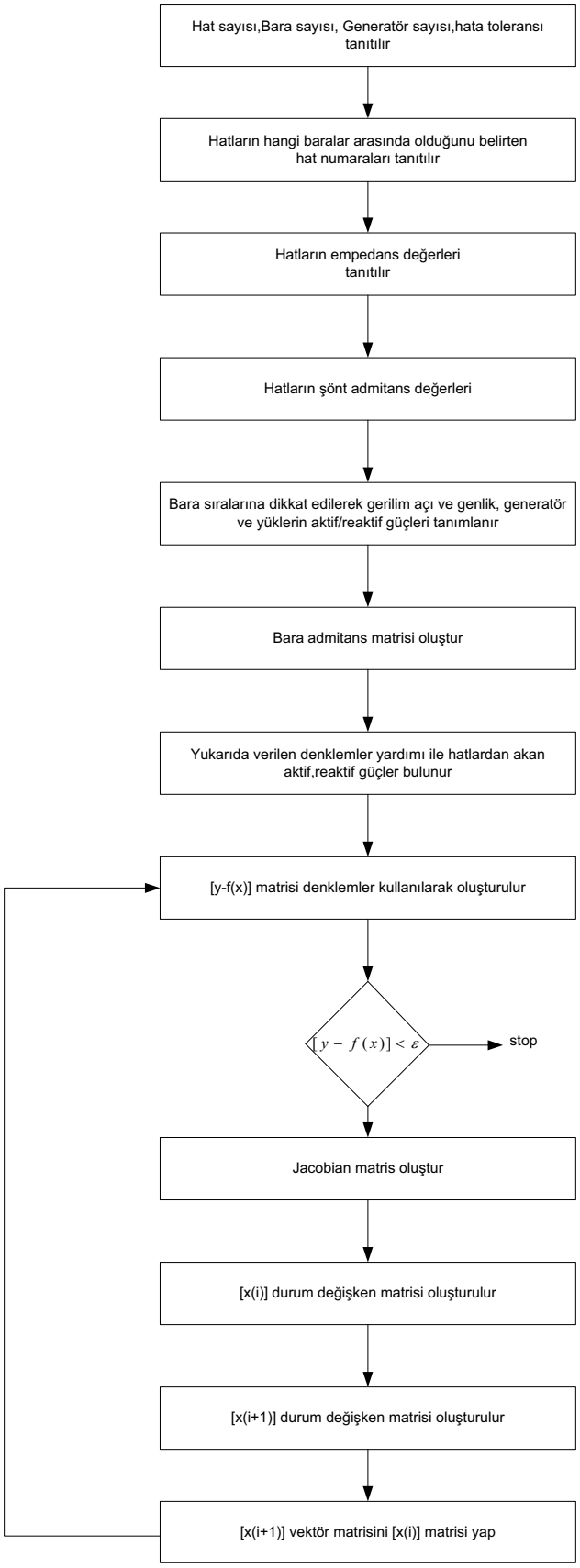
Buradan;

$$p_{ij} = \mathrm{Re}\left\{ v_i (v_i^* - v_j^*) y_{ij}^* + v_i v_i^* \left(\frac{y_{ij}'}{2}\right)^* \right\} \tag{3.42}$$

$$q_{ij} = \mathrm{Im}\left\{ v_i (v_i^* - v_j^*) y_{ij}^* + v_i v_i^* \left(\frac{y_{ij}'}{2}\right)^* \right\} \tag{3.43}$$

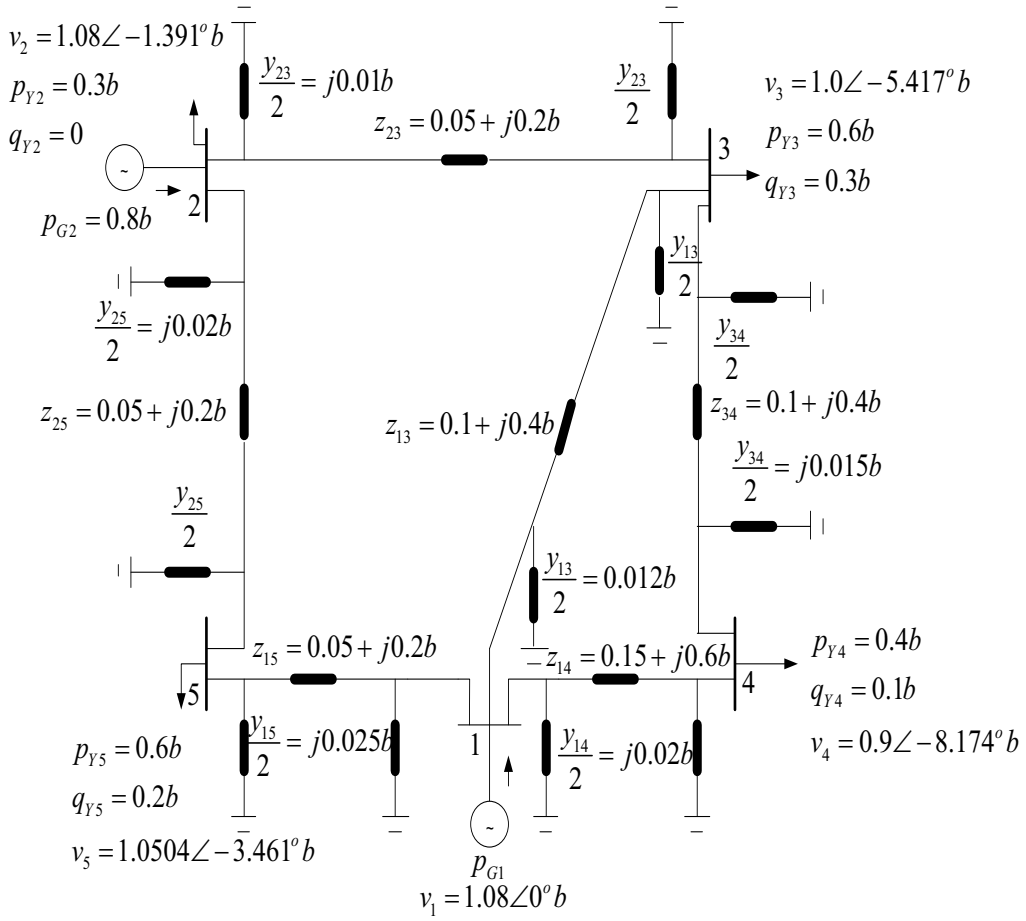
Güç akışına başlamadan önce mevcut olan bazı sınırlar vardı, örneğin jeneratörlerin üretebilecekleri maximum güçler gibi. Eğer bu sınırların altına ya da üstüne çıkılırsa algoritmada değişiklikler yapılması şarttır[25]. Bu değişiklikler;

- Eğer PQ baralarından birinde bara gerilim genlik değeri sınır değeri aşmış ya da altına düşmüş ise bu bara için v=vmin ya da v=vmax alınıp bu baradan çekilen reaktif güç değeri (qy) serbest bırakılır. Böylelikle PQ barası PV barası yapılmış olur ve yeni baranın pgen değeri py alınarak güç akışı algoritmasına devam edilir.
- Eğer PV baralarının birisinde reaktif güç değeri sınırdan uzaklaşmış ise q=qmin ya da q=qmax alınır ve v değeri serbest bırakılarak bara PQ barası haline getirilir. Bu baraya ilişkin py değeri de pgen alınır ve güç akışına devam edilir.
- Eğer salınım barasından sisteme aktarılan reaktif güç değeri sınırı geçer ise q1=q1min ya da q1=q1max alınarak salınım barasının geriliminin genlik değeri serbest bırakılır. Böylelikle salınım barası PQ barası haline gelmiş olur ve salınım barasının yerine PV baralarından biri seçilir.



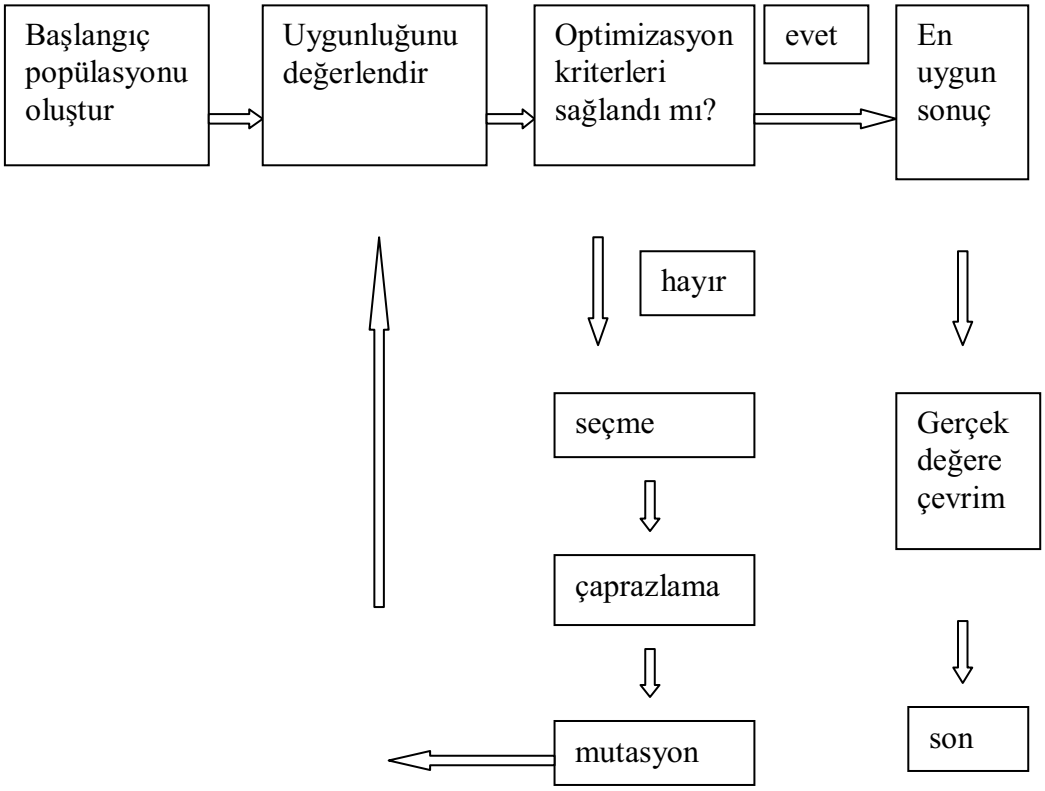
Şekil 3.2. Newton-Raphson Algoritmasının Temel Mantığı

BÖLÜM 4. BEŞ BARALI SİSTEM İÇİN GENETİK ALGORİTMA YARDIMI İLE GÜÇ AKIŞI



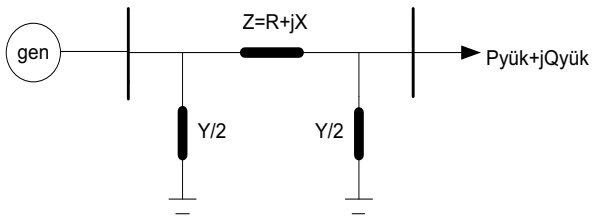
Şekil 4.1. Beş Baralı Sistem Gösterimi

Şekil 4.1'de güç akışı yapılan 5 baralı sistemin bir kutuplu gösterimi verilmiştir. Şekil 4.2'de ise genetik algoritma işaret akış şeması gösterilmiştir. Güç alışını yapmakta kullanılan genetik algoritma Şekil 4.2'de verilen akış diyagramı mantığına göre çalışır.



.

Şekil 4.2. Genetik Algoritma Akış Diyagramı



Şekil 4.3. Güç Akışı Algoritmasında Kullanılan Hattın Bir Kutuplu Pi Eşdeğer Devresi

Tablo 4.1. Şekil 4.1'de Verilen Beş Baralı Sistemin Hatlarına Ait R, X ve Y Değerleri

Hat no	R(b)	X(b)	Y(b)
1-3	0.1	0.4	0.012
1-4	0.15	0.6	0.02
1-5	0.05	0.2	0.025
2-3	0.05	0.2	0.015
2-5	0.05	0.2	0.02
3-4	0.1	0.4	0.015

5 baralı test sisteminde salınım barası 1. bara kabul edilmiş ve hat değerlerinden kondüktans değeri ihmal edildiğinden admitans değeri süseptans değerine eşit alınmıştır.

Tablo 4.2. Baralara Ait Gerilim Genlik ve Açı Başlangıç Değerleri

Bara no	V(b)	δ (b)
1(PV)	1.08	0
2(PV)	1.08	-1.391
3(PQ)	1.0	-5.417
4(PQ)	0.9	-8.174
5(PQ)	1.0504	-3.461

Tablo 4.2 de verilen başlangıç değerleri güç akışı algoritmalarında sisteme tanıtılarak daha kısa zamanda doğru sonuca ulaşılmasını sağlayacaktır.

4.1 de verilen empedans ve admitans değerleri Şekil 4.3 de belirtilen hattın bir kutuplu pi eşdeğer devresinden hareketle, Matlab yardımı ile, Ybara yani bara admitans matrisini bulmada kullanır. Bunu;

Empedans vektörü oluşturulurken Şekil 4.3 den hareketle;

Empedans=[z13, z14, z15, z23, z25, z34]

Şönt admitans vektörünü oluştururken Şekil 4.3 den hareketle;

$$\text{şönt}=[\frac{y_{13}}{2}, \frac{y_{14}}{2}, \frac{y_{15}}{2}, \frac{y_{23}}{2}, \frac{y_{25}}{2}, \frac{y_{34}}{2}]$$

yazılabilinir.

Yukarıda verilen empedans ve şönt admitans matrisi oluştururken her zaman birim değerlerle işlem yapılır. Güç akışının birim değerlerle çalışması işlem kolaylığı açısından tercih edilir[27].

Matlab ortamında;

nb= bara admitans matrisi hesaplanacak sistemde yer alan bara sayısı,

hb=[1 1 1 2 2 3;

3 4 5 3 5 4]; bu matrisin aynı sütununda yer alan değerler hat numaralarını oluşturur,

Hatsayısı= bara admitans matrisi hesaplanacak sistemde yer alan hat sayısı,

Ybara'yı bulduracak MATLAB kodları;

```
empedans=[0.1+0.4i,0.15+0.6i,0.05+0.2i,0.05+0.2i,0.05+0.2i,0.1+0.4i]
sont=[0.012i,0.02i,0.025i,0.01i,0.02i,0.015i]
for i=1:nb
   for j=1:nb%burada bara matris icin yer actik
      y(i,j)=0;
   end
end
 for i=1:nb
   for j=1:nb
      if i~=j
         for m=1:hatsayisi
            if (i==hb(1,m))&(j==hb(2,m))% kosegen disi elemanlari olusturuldu
               y(i,j)=-1/empedans(m);
            end
            if (i==hb(2,m))&(j==hb(1,m))% kosegen disi elemanlari olusturuldu
               y(i,j)=-1/empedans(m);
```

```
            end
        end
    end
end
 for i=1:nb
    for j=1:hatsayisi
        if i==hb(1,j)
            y(i,i)=y(i,i)+(1/empedans(j))+sont(j);
        end
        if i==hb(2,j)
            y(i,i)=y(i,i)+(1/empedans(j))+sont(j);
        end
    end
end
for i=1:nb
    for j=1:nb
        g(i,j)=real(y(i,j));
        b(i,j)=imag(y(i,j));
    end
end
```

Matematiksel anlamda Ybara oluşturulurken ana köşegen ve ana köşegen dışı elemanların hesaplanması gerekmektedir. Ynn olarak ana köşegen elemanları

hesaplanırken, n. baraya temas eden hatların admitans ($Z' = Y$) değerlerinin toplamı ve yine bu baraya temas eden süseptans değerlerinin toplamı alınır[37]:

$$y_{11}=y_{14}+y_{15}+y_{13}+(\frac{y_{14}}{2})+(\frac{y_{15}}{2})+(\frac{y_{13}}{2})$$

Ynk değerleri olarak köşegen dışı elemanlar alınırsa, n ve k'ıncı baralar arasındaki empedansın tersinin negatif işaretli değeri alınır:

$$y_{13}=-z_{13}^{-1}$$

5 baralı sistem için empedans matrisi aşağıda verildiği gibi olacaktır. Ybara admitans matrisinin reel bileşenleri g, imajiner bileşenleri ise b ile gösterilirse;

[Ybara]= $(y_{ij}=g_{ij}+jb_{ij})$=>

2.1569 - 8.5705i	0	-0.5882 + 2.3529i	-0.3922 + 1.5686i	-1.1765 + 4.7059i
0	2.3529 - 9.3818i	-1.1765 + 4.7059i	0	-1.1765 + 4.7059i
-0.5882 + 2.3529i	-1.1765 + 4.7059i	2.3529 - 9.3748i	-0.5882 + 2.3529i	0
-0.3922 + 1.5686i	0	-0.5882 + 2.3529i	0.9804 - 3.8866i	0
-1.1765 + 4.7059i	-1.1765 + 4.7059i	0	0	2.3529 - 9.3668i

[g]=>

2.1569	0	-0.5882	-0.3922	-1.1765
0	2.3529	-1.1765	0	-1.1765
-0.5882	-1.1765	2.3529	-0.5882	0
-0.3922	0	-0.5882	0.9804	0
-1.1765	-1.1765	0	0	2.3529

[b]=>

-8.5705	0	2.3529	1.5686	4.7059
0	-9.3818	4.7059	0	4.7059
2.3529	4.7059	-9.3748	2.3529	0
1.5686	0	2.3529	-3.8866	0
4.7059	4.7059	0	0	-9.3668

elde edilir.

Genetik algoritma ve Newton-Raphson metodu kullanarak yapılacak olan güç akışı çalışmalarında aşağıdaki bilgiler kullanılmıştır [28-30].

1-Yapılan bu güç akışında bir fazlı güç akışı modeli kullanılmıştır, yani sadece bir faz dikkate alınarak işlem yapılmıştır. Tüm fazların dengeli olduğu kabul edilmiştir.

2-Güç akışı sonunda hesaplanan baraların aktif ve reaktif güç değerleri;

-sistemin gerilim genliği,

-sistemin gerilimlerinin açıları,

-sistemin içerisinde yer alan hattın parametrelerine, yani R ve X e bağlıdır.

3-Sonuçları yorumlama kolaylığı sağlaması ve bazı elemanların elektriksel modellerini basitleştirmesi nedeni ile güç akışı eşitliklerinde çoğunlukla birim değerler kullanılır.

4-Güç akışının bitmesi demek, her bir baradaki güç dengesinin sağlanması, yani giren ve çıkan güçlerin eşitlenmesi demektir.

5-Güç akışı yapılan bu sistemde P ve Q'nun tüm değişkenlerle ilişkisi olmasına rağmen $P \Rightarrow \delta$ ve $Q \Rightarrow |v|$ ile daha yakın bir ilişkisi söz konusudur. Uyarma akımı kontrol edilerek (dolayısı ile generatör barası gerilim genlik değeri), santrallerde endüktif veya kapasitif reaktif güç üretilebilir. Bu nedenle v ve Q birbirleri ile daha yakın ilişki halindedir. Generatör aktif güç değeri ise türbin tahrik gücü ile bağlantılı olduğundan, türbin ise generatör ile aynı hızda döndüğünden, generatör sargıları da türbin ile aynı devirde dönecektir. Bu nedenle faz sargısının pozisyonu ile türbinin ürettiği güç arasında yakın ilişki bulunur (P ile δ).

6-Güç akışı yapılırken merkezi bilgisayarlar sistemde yer alan kontrol değişken verilerini belirli zaman aralıklarında yenileyerek güç akışı tekrarlanır. Yani sistemde bir geri besleme söz konusudur.

7-Şekil 4.1'de verilen beş baralı sistem için yapılan güç akışında;

Kontrol değişkenleri;

-Slack bara dışındaki generatör barası aktif güçleri

-Generatör barası gerilim genlik değerleri

Durum değişkenleri ise:

-Slack bara dışındaki baraların gerilim açı değerleri,

-Yük barası gerilim genlik değerleri

kabul edilir

Kontrol değişkenleri güç akışı boyunca sabit değerde tutulurken, güç akışı sonunda durum değişkenlerinin bu kontrol değişkenlerine bağlı olarak değerler alır.

8-Güç sistemlerinde güç akışı ile ilgili kısıtlar;

-bara gerilim genlikleri(lineer)

-hatlardan akan güçlerin limitleri,

-generatörün üreteceği aktif-reaktif gücün üst değerleri,

-trafo kademe ayarlarının alt-üst değerleri (Şekil 4.1'deki sistem için mevcut değil)

olarak alınır.

4.1. Beş Baralı Sistemin Genetik Algoritma ve Newton-Raphson Güç Akışı Sonuçlarının Değerlendirilmesi

Temel görevi amaç fonksiyonunu minimize etmek olan genetik algoritmanın Matlab program kodları Bölüm 4.2'de verilmiştir. Kodların işleyişine geçmeden önce, Şekil 4.1'de verilen sistem için amaç fonksiyonu ve değişkenleri tanıtılacaktır. Formül 4.1 de belirtilen p_{gi} ; i. generatör barasının sistemin aktif güç ihtiyacını karşılamak için ürettiği aktif güç değerinin birim olarak karşılığıdır. Fakat, slack baranın sistemin ihtiyacını karşılamak için ne kadar aktif güç üreteceği başlangıçta bilinemediği için, slack bara generatör aktif güç değeri, güç akışı sonunda, formul 4.1 yardımı ile

hesaplanır. Sistemde var olan generatör baralarının üreteceği aktif güçler; ‘’sisteme giren güçler’’ olarak adlandırılır. p_{yi} ; değeri ise i. baraya bağlı yükün tüketeceği aktif güç bileşeninin birim değer karşılığıdır. Sistemde var olan yüklerin tükettiği aktif güçler ‘’sistemden çıkan güçler’’ olarak adlandırılır ise, hatlarda kaybolan aktif güç değeri giren güç-çıkan güç eşitliğinden bulunabilir. (formül 4.1’nin bulunuşu 3. bölümde ayrıntılı şekilde açıklanmıştır.)

ng =sistemde var olan generatör sayısı,

n =sistemdeki toplam bara sayısı,

p_{gi} =i. generatör barasının ürettiği aktif güç değeri (birim değer),

p_{yi} =i. baranın yükünün tükettiği aktif güç değeri (birim değer),

$$\sum_{i=1}^{ng} p_{gi} - \sum_{j=1}^{n} p_{yi} = \text{amaç fonksiyonumuz} \qquad (4.1)$$

ş4.2’deki 5 baralı sistemde, pgen2, vgen1,vgen2 değerleri kontrol değişkenleri olarak alınmış, slack bara hariç tüm bara gerilim açıları ve yük bara gerilim genlikleri ise durum değişkeni olarak kabul edilmiştir.

Genetik algoritma güç akışı sonuçları;

Tablo 4.3. Genetik Algoritma İle Güç Akışı Sonuçları

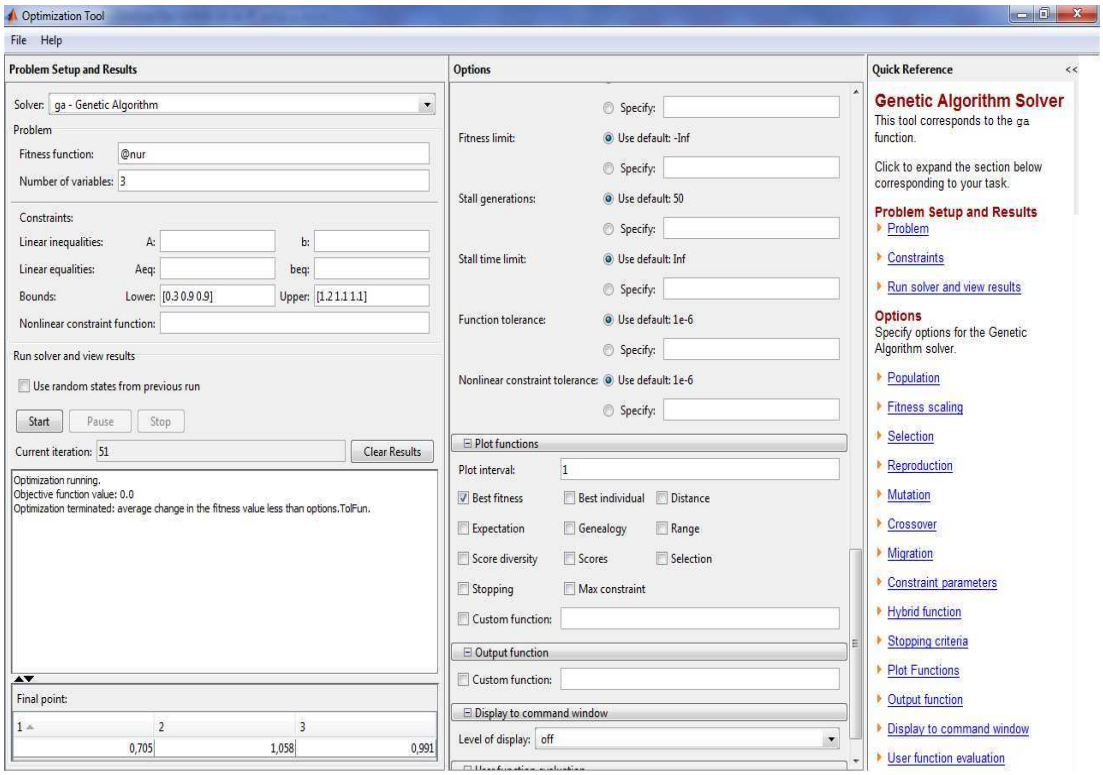
Bara no	Gerilim Genlik	Gerilim Açı(der)	Yük Bara Aktif G	Yük Bara Reaktif G	Gen. Bara Aktif G	Gen. Bara Reaktif G
1	1.0800	0	0	0	1.1470	0.2006
2	1.0800	-1.3910	0	0	0.5939	0.3635
3	1.0040	-5.4170	0.6	0.3	0	0
4	0.9805	-8.1740	0.4	0.1	0	0
5	1.0368	-3.4610	0.6	0.2	0	0

Newton-Raphson güç akışı sonuçları;

Tablo 4.4. Newton-Raphson İle Güç Akışı Sonuçları

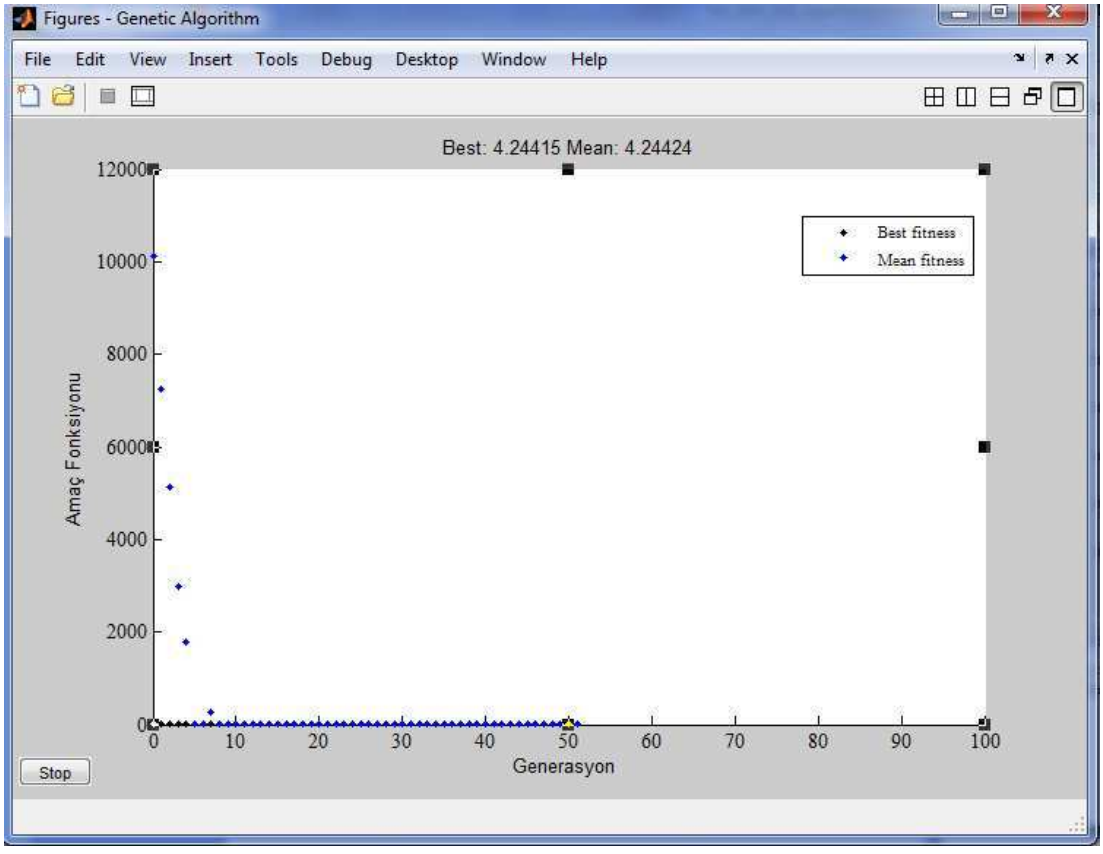
Bara no	Gerilim Genlik	Gerilim Açı(der)	Yük Bara Aktif G	Yük Bara Reaktif G	Gen. Bara Aktif G	Gen. Bara Reaktif G
1	1.0800	0	0	0	0.9097	0.1141
2	1.0800	-1.4152	0	0	0.7338	0.4411
3	1.0104	-5.5919	0.6	0.3	0	0
4	0.9846	-8.4997	0.4	0.1	0	0
5	1.0430	-3.5945	0.6	0.2	0	0

Tablo 4.3 ve Tablo 4.4'den de görüldüğü üzere, genetik algoritma ile bulunan sonuçlar, Newton-Raphson ile yapılan güç akışı sonuçlarından çok da farklı değildir. Bunun nedeni; genetik algoritmanın amaç fonksiyonunu minimize etmek için en uygun durum değişken değerlerini aramasıdır. Genetik algoritma, geleneksel metotlarla çözümü zor veya çözüm zamanı, problemin büyüklüğü ile doğru orantılı değişen problemlerde, zaman tasarrufu sağlayan sonuçlar verir.



Şekil 4.4. Genetik Algoritmanın Matlab Toolbox Görüntüsü

Şekil 4.4'de verilen Matlab toolbox görüntüsünü elde edebilmek için program öncelikle çalıştırılmalıdır. Daha sonra command window ortamına gatool yazılıp enter'a tık'lanıldığında, Şekil 4.4'de verilen ekran görüntüsü ile karşılaşılır. Çıkan 'Optimization Tool' penceresinde 'Fitness Function' boşluğuna genetik algoritma içine gömülen Newton-Raphson dosyasının adı başına @ konulur. Gerekli sınır aralıkları 'bounds' kısmına girilir. Sağ kısımda yer alan 'options' bölmesinden genetik algoritmasının yapısı (seçim stratejisi, çaprazlama stratejisi gibi) ve kontrol parametreleri (mutasyon oranı, çaprazlama oranı gibi) istenildiği gibi ayarlanabilinir. İstenir ise 'plot function' kısmından 'best function' kısmı işaretlenerek amaç fonksiyonunun durumu çizdirilebilir. Şekil 4.5'de Şekil 4.1'de verilen beş baralı güç sistemine ait güç akışı amaç fonksiyonun, güç akışı süresince aldığı değerler çizdirilmiştir.



Şekil 4.5. Şekil 4.1'de Verilen Beş Baralı Sisteme Ait Genetik Algoritma Yardımı İle Güç Akışı Sonucu Elde Edilen Amaç Fonksiyonu Görüntüsü

Şekil 4.5'den görüldüğü üzere, genetik algoritma yardımı ile (yapılan güç akışı sonunda alınan amaç fonksiyonun ekran görüntüsünden de anlaşılacağı gibi) genetik algoritma ile hat kayıpları minimize edilmiştir.

4.2. Program Kodları

4.2.1. Newton-Raphson güç akışı program kodları

```
%ADIM 1,sistem başlangıç değerlerinin girilmesi
hatsayisi=6;
nb=5;
ng=2;
n=5;
epsilon=0.0001;
hb=[1 1 1 2 2 3; 3 4 5 3 5 4];
empedans=[0.1+0.4i,0.15+0.6i,0.05+0.2i,0.05+0.2i,0.05+0.2i,0.1+0.4i];
sont=[0.012i,0.02i,0.025i,0.01i,0.02i,0.015i];
vb=[1.08,1.08,1,0.9,1.0504];
aci=[0,-1.391,-5.417,-8.174,-3.461];
pgen=[0,0.8,0,0,0];
qgen=[0,0,0,0,0];
pyuk=[0,0.3,0.6,0.4,0.6];
qyuk=[0,0,0.3,0.1,0.2];
A=[0.5 0.5];
B=[0.351 0.389];
C=[0.444 0.460];
wsifir=17;
```

```
wz=1;

iterp=0;

vsinir=[1+1.1i,1+1.1i,0.9+1.05i,0.9+1.05i,0.9+1.05i];

psinir=[0.1+1i,0.1+1i];

c=0.92;

eopt=0.1;

iter=0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%ADIM 2,bara admitans matrisinin oluşturulması

%

for i=1:nb

   for j=1:nb

      y(i,j)=0;

   end

end


for i=1:nb

   for j=1:nb

      if i~=j

         for m=1:hatsayisi

            if (i==hb(1,m))&(j==hb(2,m))% kosegen disi elemanlari
olusturuldu

               y(i,j)=-1/empedans(m);
```

```
                end
                if (i==hb(2,m))&(j==hb(1,m))% kosegen disi elemanlari
olusturuldu
                    y(i,j)=-1/empedans(m);
                end
            end
        end
    end
    for i=1:nb
        for j=1:hatsayisi
            if i==hb(1,j)
                y(i,i)=y(i,i)+(1/empedans(j))+sont(j);
            end
            if i==hb(2,j)
                y(i,i)=y(i,i)+(1/empedans(j))+sont(j);
            end
        end
    end
    for i=1:nb
        for j=1:nb
            g(i,j)=real(y(i,j));
            b(i,j)=imag(y(i,j));
```

```
    end
end
%%%%%%%%%%%%%%%%%%
for i=1:nb
    acir(i)=aci(i)*pi/180;%aci radyana cevrildi
end
for i=1:nb
    for j=1:nb
        rad(i,j)=acir(i)-acir(j);
    end
end
%%%%%%%%%%%%%%%%%%
%
iterp=0;
x1=[0 0 0 0 0 0 0];
x2=[0 0 0 0 0 0 0];
iter=0;
j=zeros(7,7);
dfp=1;
%ADIM 4, iterasyon döngü başlangıcı
while max(abs(dfp))>eopt
    iterp=iterp+1;
    iter=0;
```

```
%ADIM 5,durum değişken vektörünün oluşturulması
  for i=1:(nb-1)%slack bara acisi alinmiyor
     x1(i)=acir(i+1);
  end
  for i=nb:7
     x1(i)=vb(i-2);
  end
  %
  %baralara ait aktif reaktif guc dengesi arastirilir
  vv=zeros(1,nb);
  vu=zeros(1,nb);
  p=zeros(1,nb);
  for i=1:nb
     for j=1:nb
        vu(j)=vb(j)*((g(i,j)*cos(rad(i,j)))+(b(i,j)*sin(rad(i,j))));
        vv(i)=vv(i)+vu(j);
     end
     p(i)=vb(i)*vv(i);
  end
  vv=zeros(1,nb);
  vu=zeros(1,nb);
  q=zeros(1,nb);
  for i=1:nb
```

```
    for j=1:nb
        vu(j)=vb(j)*((g(i,j)*sin(rad(i,j)))-(b(i,j)*cos(rad(i,j))));
        vv(i)=vv(i)+vu(j);
    end
    q(i)=vb(i)*vv(i);
end
for i=2:nb
    farkp(i)=-pgen(i)+pyuk(i)+p(i);
end
for i=3:nb
    farkq(i)=qyuk(i)+q(i);
end
for i=1:(nb-1)
    xx1(i)=farkp(i+1);
end
for i=1:(nb-ng)
    xx2(i)=farkq(i+ng);
end
xx=[xx1,xx2];
 while max(abs(xx))>epsilon
    iter=iter+1;
     for i=2:nb
        acir(i)=x1(i-1);
```

```
end
for i=3:nb
   vb(i)=x1(i+2);
end
for i=1:nb
   for j=1:nb
      rad(i,j)=acir(i)-acir(j);
   end
end
 for i=2:nb
   pilk=0;
   for j=1:nb
      pilk=pilk+vb(j)*(g(i,j)*cos(rad(i,j))+b(i,j)*sin(rad(i,j)));
   end
   p(i)=vb(i)*pilk;
end
 for i=3:nb
   qilk=0;
   for j=1:nb
      qilk=qilk+vb(j)*(g(i,j)*sin(rad(i,j))-b(i,j)*cos(rad(i,j)));
   end
   q(i)=vb(i)*qilk;
end
```

```
for i=2:nb
   farkp(i)=-pgen(i)+pyuk(i)+p(i);
end
for i=3:nb
   farkq(i)=qyuk(i)+q(i);
end
kon(1:4)=farkp(2:5);%kontrol vektoru
kon(5:7)=farkq(3:5);%kontrol vektoru
for i=1:(nb-1)
   xx1(i)=farkp(i+1);
end
for i=1:(nb-ng)
   xx2(i)=farkq(i+ng);
end
xx=[xx1,xx2];
%jacobian icin
j1=zeros(4,4);
j2=zeros(4,3);
j3=zeros(3,4);
j4=zeros(3,3);
jac1=zeros(5,5);
jac3=zeros(5,5);
for i=1:nb
```

```
    for j=1:nb
        if (i~=j)
            jac1(i,i)=jac1(i,i)+vb(j)*(-g(i,j)*sin(rad(i,j))+b(i,j)*cos(rad(i,j)));
            jac3(i,i)=jac3(i,i)+vb(j)*(g(i,j)*cos(rad(i,j))+b(i,j)*sin(rad(i,j)));
        end
    end
end
for k=1:nb
    jac1(k,k)=vb(k)*jac1(k,k);
    jac3(k,k)=vb(k)*jac3(k,k);
end
for k=1:nb
    for n=1:nb
        if (k~=n)
            jac1(k,n)=vb(k)*vb(n)*(g(k,n)*sin(rad(k,n))-
b(k,n)*cos(rad(k,n)));
            jac3(k,n)=-
vb(k)*vb(n)*(g(k,n)*cos(rad(k,n))+b(k,n)*sin(rad(k,n)));
        end
    end
end
j1=-jac1(2:nb,2:nb);
j3=-jac3(ng+1:nb,2:nb);
 jac2=zeros(nb,nb);
```

```
jac4=zeros(nb,nb);
for k=1:nb
   for n=1:nb
      if (k~=n)

jac2(k,k)=jac2(k,k)+vb(n)*(g(k,n)*cos(rad(k,n))+b(k,n)*sin(rad(k,n)));
         jac4(k,k)=jac4(k,k)+vb(n)*(g(k,n)*sin(rad(k,n))-
b(k,n)*cos(rad(k,n)));
      end
   end
end
for k=1:nb
   jac2(k,k)=jac2(k,k)+2*vb(k)*g(k,k);
   jac4(k,k)=jac4(k,k)-2*vb(k)*b(k,k);
end
for k=1:nb
   for n=1:nb
      if (k~=n)
         jac2(k,n)=vb(k)*(g(k,n)*cos(rad(k,n))+b(k,n)*sin(rad(k,n)));
         jac4(k,n)=vb(k)*(g(k,n)*sin(rad(k,n))-b(k,n)*cos(rad(k,n)));
       end
   end
end
j2=-jac2(2:5,3:5);
```

```
    j4=-jac4(3:5,3:5);
     jac=[j1 j2;j3 j4];
     jacters=inv(jac);
    x2=x1'+jacters*kon';
    x1=x2';
    if iter>20    %Hata durdurma komutu
       break
    else
    end
end
 for i=1:ng
    pgen(i)=p(i)+pyuk(i);
end
for i=1:ng
    qgen(i)=q(i)+qyuk(i);
end
%%%%%%%%%%
%df1/dx  hesaplayalim=>
 f1pg1=(2*A(1)*pgen(1))+B(1);%dfp1/dpg1
 for i=2:nb
    p1aci(1,(i-1))=vb(1)*vb(i)*((g(1,i)*sin(rad(1,i)))-(b(1,i)*cos(rad(1,i))));
end
for i=(ng+1):nb
```

```
    p1v(1,(i-ng))=vb(1)*((g(1,i)*cos(rad(1,i)))+(b(1,i)*sin(rad(1,i))));
end
p1x1=[p1aci p1v];
p1x=p1x1';%dp1/dx i bulduk
 dfdx=f1pg1*p1x;%dfp/dx=(dfp/dpg1)*(dp1/dx)
%%%%%%%%%%%%%
%%%%%%%%%%%
%dw/dx i bulalim=>
for i=2:nb
    wx1(i-1)=0;%acinin kisiti olmadigi icin sifir
end
toi=wsifir*(wz^iterp);
vr=real(vsinir);
vi=imag(vsinir);
for i=1:(nb-ng)
    if vb(i+ng)<vr(i+ng)
        wx2(i)=2*toi*(vb(i+ng)-vr(i+ng));
    elseif vb(i+ng)>vi(i+ng)
        wx2(i)=2*toi*(vb(i+ng)-vi(i+ng));
    else
        wx2(i)=0;
    end
end
```

```
wx=[wx1 wx2];
dwdx=wx';%dw/dx bulundu
%%%%%%%%%%%%%
 %%%%%%%%%%%%%%%
%[(dgp/dx) (dgq/dx)]  i bulalim=>
dgx=j';%jacobian in transpozudur
%%%%%%%%%%%%%
%%%%%%%%%%%%%
%o zaman lamdax=>
kk=-(dfdx+dwdx);
lamx=(inv(dgx))*kk;
%%%%%%%%%%
%%%%%%%%%%%%%
%[(dgp/du) (dgq/du)] i bulalim=>
for i=2:nb
   for j=2:ng
      if i==j
         gpu((j-1),(i-1))=-1;
      else
         gpu((j-1),(i-1))=0;
      end
   end
end
```

```
for i=(ng+1):nb
   for j=2:ng
      gqu((j-1),(i-ng))=0;
   end
end
dgdu=[gpu gqu];%dg/du matrisi bulundu
%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%
%uyeni yi bulalim
pr=real(psinir);
pi=imag(psinir);
if iterp==1
   fk1=0;
   for i=2:ng
      rk1(i-1)=-dfp((i-1),1);
      rk(i-1)=rk1(i-1);
      fk1=fk1+(dfp((i-1),1)^2);
   end
   fk1=sqrt(fk1);
   for i=2:ng
      pgen(i)=pgen(i)+(c*rk(i-1));
   end
else
```

```
fk=0;
 for i=2:ng
    fk=fk+(dfp((i-1)^2));
end
 fk=sqrt(fk);
beta=(fk/fk1)^2;
fk1=fk;
for i=2:ng
    rk(i-1)=-dfp((i-1),1)+(beta*rk1(i-1));
    rk1(i-1)=rk(i-1);
end
 for i=2:ng
    pgen(i)=pgen(i)+(c*rk(i-1));
    if pgen(i)<pr(i)
        pgen(i)=pr(i);
    elseif pgen(i)>pi(i)
        pgen(i)=pi(i);
    else
        pgen(i)=pgen(i);
    end
 end
end%if in end i
%%%%%%%%%%%%%%%
```

```
    %toplam hatlardaki kayiplarının bulunması=>
    hk=0;
    for i=1:(n-1)
       for j=(i+1):n
          hk=hk+(-g(i,j)*((vb(i)^2+vb(j)^2)-2*vb(i)*vb(j)*cos(rad(i,j))));
       end
    end
    %%%%%%%%%%%%%%
    if iterp>25    %Hata durdurma komutu
       break
    else
    end
end%p opt un sonu
 disp('*****************************')
disp('*5 barali sistem icin sonuclar*')
disp('       ')
disp('bara adm matrisi')
disp('==========')
y
disp('           ')
disp('gerilim  genlikleri')
disp('========')
vb
```

```
disp('          ')
disp('iterasyon sayisi')
disp('========')
iter
disp('     ')
disp('jaco matrisi')
disp('======')
jac
disp('    ')
disp('durum deg vektoru')
disp('======')
x2
disp('    ')
disp('son guc durumlari')
disp('======')
pgen(1)
pgen(2)
qgen(1)
qgen(2)
disp('          ')
disp('          ')
disp('p opt sonrasi hat kayiplari')
disp('========')
```

hk

4.2.2 Genetik algoritma yardımı ile güç akışı program kodları

```
function p_u

clear all;

clc;

FitnessFunction = @ulas; %

numberOfVariables = 3;

%Değişkenler:

lb = [0.3 0.9 0.9]; % alt limit

ub = [1.2 1.1 1.1]; % ust limit

A = []; b = []; % lineer olmayan eşitsizlik kısıtları

Aeq = []; beq = []; % lineer olmayan eşitlik kıstları

[x,Fval,exitFlag,Output] = ga(FitnessFunction,numberOfVariables,A, ...

   b,Aeq,beq,lb,ub);

x

Fval

bs=5;

gs=2;

bara=[1.0000   1.00      0      0      0      0      0

      2.0000   1.00      0      0      0      0.8    0

      3.0000   1.00      0      0.6    0.3    0      0

      3.0000   1.00      0      0.4    0.1    0      0

      3.0000   1.00      0      0.6    0.2    0      0];
```

```
 for d=2:gs
   bara(d,6)=x(1,d-1);
end
for d=1:gs
   bara(d,2)=x(1,d+(gs-1));
end
 Pload=sum(bara(:,4));
 b=[   2.1569        0      -0.5882      -0.3922      -1.1765
       0         2.3529     -1.1765        0          -1.1765
     -0.5882     -1.1765     2.3529      -0.5882        0
     -0.3922        0       -0.5882       0.9804        0
     -1.1765     -1.1765      0            0          2.3529];
g=[   -8.5705      0          2.3529       1.5686      4.7059
        0         -9.3818     4.7059       0           4.7059
      2.3529      4.7059     -9.3748       2.3529      0
      1.5686      0          2.3529      -3.8866      0
      4.7059      4.7059      0            0          -9.3668];
Psnr=[ 0.3  0.3  ; 1.2 1.2 ];%generatörler sinir degerleri
Qsnr=[-0.2 -0.1 ; 0.6 0.6];
hata=10^(-6);
 snc=2;
ite3=0;
% döngülerin başlangıcı
```

```
while snc>1
   ite3=ite3+1;
   %%
   P=zeros(bs,1);
   for l=1:bs
      aratop=0;
      if bara(l,1)>1
         for j=1:bs
            aratop=aratop+bara(j,2)*((g(l,j)*cos(bara(l,3)-
bara(j,3)))+b(l,j)*sin(bara(l,3)-bara(j,3)));
         end
         P(l,1)=bara(l,2)*aratop;
      end
   end
      %%%  Generatör baraları dışında kalan Yük baralarının alternatif akım
hatlarından çektiği Reaktif GÜÇ %%%
   Q=zeros(bs,1);
   for l=1:bs
      aratop2=0;
      if bara(l,1)==3
         for j=1:bs
            aratop2=aratop2+bara(j,2)*((g(l,j)*sin(bara(l,3)-bara(j,3)))-
(b(l,j)*cos(bara(l,3)-bara(j,3))));
         end
```

```
        Q(l,1)=bara(l,2)*aratop2;
    end
end
%%% Aktif  ve Reaktif Güç Dengesi  %%%
rt=0;
ty=0;
yt=0;
for l=1:bs
    if bara(l,1)>1
        ty=ty+1;;
        phat(ty,1)=bara(l,6)-bara(l,4)-P(l,1);
        rt=rt+1;
    end
    if bara(l,1)==3
        yt=yt+1;
        qhat(yt,1)=bara(l,7)-bara(l,5)-Q(l,1);
        rt=rt+1;
    end
end
    for l=1:ty
    error(l,1)=phat(l,1);
end
for l=1:yt
```

```
    error(l+ty,1)=qhat(l,1);
end
%% Durum değişkenleri
ss=1;
ff=0;
for l=1:bs
    if bara(l,1)==2
        ddegisken(ss,1)=bara(l,3);
        ss=ss+1;
        ff=ff+1;
    end
    if bara(l,1)==3
        ddegisken(ss,1)=bara(l,3);
        ss=ss+1;
        ff=ff+1;
    end
end
% ddeğişken matrisinin diğer elemanları yük baralarının gerilim değeri olacaktır..
for l=1:bs
    if bara(l,1)==3
        ddegisken(ss,1)=bara(l,2);
        ss=ss+1;
    end
```

```
end
[bb,aa]=size(ddegisken);
%jacobian matrisin olusturulmasi
j1=zeros(bs,bs);
for l=1:bs
   tp3=0;
   if bara(l,1)>1
      for z=1:bs
         if l==z
            for j=1:bs
               if ((l>j) | (j>l))
                  tp3=tp3+bara(j,2)*(((-1)*g(l,j)*sin(bara(l,3)-
bara(j,3)))+(b(l,j)*cos(bara(l,3)-bara(j,3))));
               end
               j1(l,l)=bara(l,2)*tp3;
            end
         end
      end
   end
end
for l=1:bs
   if bara(l,1)>1
      for j=1:bs
```

```
            if ((l>j) | (j>l))
                j1(l,j)=bara(l,2)*bara(j,2)*((g(l,j)*sin(bara(l,3)-bara(j,3)))-
(b(l,j)*cos(bara(l,3)-bara(j,3))));
            end
        end
    end
end
j2=zeros(bs,bs);
for l=1:bs
    tp4=0;
    if bara(l,1)==3
        for z=1:bs
            if l==z
                for j=1:bs
                    if ((l>j) | (j>l))
                        tp4=tp4+bara(j,2)*((g(l,j)*cos(bara(l,3)-
bara(j,3)))+(b(l,j)*sin(bara(l,3)-bara(j,3))));
                    end
                    j2(l,l)=tp4+(2*bara(l,2)*g(l,l));
                end
            end
        end
    end
end
```

```
for l=1:bs
    if bara(l,1)>1
        for j=1:bs
            if ((l>j) | (j>l))
                j2(l,j)=bara(l,2)*((g(l,j)*cos(bara(l,3)-bara(j,3)))+(b(l,j)*sin(bara(l,3)-
bara(j,3))));
            end
        end
    end
end
j3=zeros(bs,bs);
for l=1:bs
    tp5=0;
    if bara(l,1)==3
        for z=1:bs
            if l==z
                for j=1:bs
                    if ((l>j) | (j>l))
                        tp5=tp5+bara(j,2)*((g(l,j)*cos(bara(l,3)-
bara(j,3)))+(b(l,j)*sin(bara(l,3)-bara(j,3))));
                    end
                    j3(l,l)=bara(l,2)*tp5;
                end
            end
```

```
        end
      end
    end
    for l=1:bs
      if bara(l,1)==3
        for j=1:bs
          if ((l>j) | (j>l))
            j3(l,j)=(-1)*bara(l,2)*bara(j,2)*((g(l,j)*cos(bara(l,3)-
bara(j,3)))+(b(l,j)*sin(bara(l,3)-bara(j,3))));
          end
        end
      end
    end
    j4=zeros(bs,bs);
    for l=1:bs
      tp6=0;
      if bara(l,1)==3
        for z=1:bs
          if l==z
            for j=1:bs
              if ((l>j) | (j>l))
                tp6=tp6+bara(j,2)*((g(l,j)*sin(bara(l,3)-bara(j,3)))-
(b(l,j)*cos(bara(l,3)-bara(j,3))));
              end
```

```
                j4(l,l)=tp6-(2*bara(l,2)*b(l,l));
            end
        end
    end
end
for l=1:bs
    if bara(l,1)==3
        for j=1:bs
            if ((l>j) | (j>l))
                j4(l,j)=bara(l,2)*((g(l,j)*sin(bara(l,3)-bara(j,3)))-(b(l,j)*cos(bara(l,3)-
bara(j,3))));
            end
        end
    end
end
bir=0;
iki=0;
uc=0;
for l=1:bs
    if bara(l,1)==1
        bir=bir+1;
    end
```

```
    if bara(l,1)==2
        iki=iki+1;
    end
    if bara(l,1)==3
        uc=uc+1;
    end
end
jacob=zeros(bb,bb);
for l=(bir+1):bs
    for j=(bir+1):bs
        jacob(l-1,j-1)=j1(l,j);
    end
end
for l=(bir+1):bs
    for j=(bir+iki+1):bs
        jacob(l-1,bs-bir-iki-1+j)=j2(l,j);
    end
end
for l=(bir+iki+1):bs
    for j=(bir+1):bs
        jacob(bs-bir-iki-1+l,j-1)=j3(l,j);
    end
end
```

```
for l=(bir+iki+1):bs
    for j=(bir+iki+1):bs
        jacob(bs-bir-iki-1+l,bs-bir-iki-1+j)=j4(l,j);
    end
end
jjacob=inv(jacob);
yddegisken=ddegisken+(jjacob*error);
error2=yddegisken-ddegisken;
merror2=abs(error2); % karşılaştırma için kullanılacak..
snc=test(merror2,hata);
ddegisken=yddegisken;
asd=1;
for l=1:bs
    if bara(l,1)==2
        bara(l,3)=ddegisken(asd,1);
        asd=asd+1;
    end
    if bara(l,1)==3
        bara(l,3)=ddegisken(asd,1);
        asd=asd+1;
    end
end
for l=1:bs
```

```
    if bara(l,1)==3
        bara(l,2)=ddegisken(asd,1);
        asd=asd+1;
    end
end
ibara=bara;
ss=1;
ff=0;
for l=1:bs
    if bara(l,1)==2
        ibara(l,3)=ddegisken(ss,1);
        ss=ss+1;
        ff=ff+1;
    end
    if bara(l,1)==3
        ibara(l,3)=ddegisken(ss,1);
        ss=ss+1;
        ff=ff+1;
    end
end
for l=1:bs
    if bara(l,1)==3
        ibara(l,2)=ddegisken(ss,1);
```

```
            ss=ss+1;
        end
    end
    for d1=1:bs
        ibirey2(ite3,d1)=ibara(d1,2);
        ibirey2(ite3,bs+d1)=ibara(d1,3);
    end
end
for d=1:bs
    bara(d,2)=ibirey2(ite3,d);
    bara(d,3)=ibirey2(ite3,d+bs);
end
 %Slack Baradan Hatlara Aktarılan aktif Güç Değeri.
for l=1:bs
    tp7=0;
        for j=1:bs
            tp7=tp7+bara(j,2)*((g(l,j)*cos(bara(l,3)-bara(j,3)))+b(l,j)*sin(bara(l,3)-
bara(j,3)));
        end
       Phat(l,1)=bara(l,2)*tp7;
end
 %Generatör baralarından Hatlara Aktarılan Reaktif Güç değerleri.
for l=1:bs
```

```
    tp8=0;
        for j=1:bs
            tp8=tp8+bara(j,2)*((g(l,j)*sin(bara(l,3)-bara(j,3)))-(b(l,j)*cos(bara(l,3)-
bara(j,3))));
        end
        Qhat(l,1)=bara(l,2)*tp8;
end
bara(:,6)=bara(:,4)+Phat(:,1);
bara(:,7)=bara(:,5)+Qhat(:,1);
bara(:,3)=bara(:,3)*180/pi;
bara
for d=1:gs
    Pgen(d,1)=Psnr(1,d)*100;
    Pgen(d,2)=bara(d,6)*100;
    Pgen(d,3)=Psnr(2,d)*100;
    Qgen(d,1)=Qsnr(1,d)*100;
    Qgen(d,2)=bara(d,7)*100;
    Qgen(d,3)=Qsnr(2,d)*100;
end
sum(Pgen(:,2))
Pload
disp('NR iterasyon sayısı')
disp(ite3)
```

```
disp('Toplam Kayıp (MW)')

disp('==============')

disp((sum(Pgen(:,2))-Pload*100))
```

Genetik algoritma yardımı ile güç akışı kodlarını farklı büyüklükteki sistemlere uygulanmak istenirse,

- 'Fitness function=@' satırında genetik algoritma içine gömülmek istenen Newton-Raphson güç akışının adı yazılmalıdır [31].
- 'NumberofVariable' satırında sistemde yer alan kontrol değişkeni sayısı yazılmalıdır [31].
- lb ve ub vektörlerine kontrol değişkenlerinin alt ve üst sınırları yazılmalıdır [31].
- bs (bara sayısı) ve gs (generatör sayısı) değerleri sisteme uyumlu olarak değiştirilmelidir [31].
- [bara] matrisinin satır sayısı, sistemde yer alan bara sayısına bağlı olarak değişir. Birinci satıra slack bara değerleri birinci sütunda 1.000 olarak tanımlanmıştır. Birinci satırın altına gelecek satırlara sırasıyla generatör baraları daha sonra ise yük baraları değerleri yazılmalıdır. Generatör bara değerleri 2.000, yük bara değerleri ise 3.000 olarak alınmıştır. Ayrıca bu matris içerisin de yer alan değerler sistemden sisteme değişir [31].
- Psnr ve Qsnr değerleri sistemde yer alan generatörün aktif ve reaktif güç sınırlarına göre yeniden şekillenir [31].
- Bulunan değişken değerlerinde sınırları aşma gibi bir durum söz konusu ise Newton-Raphson algoritması buna program içerisinde müdahale etmez, algoritma bittikten sonra sınır değerleri kontrolü yapılır. Fakat genetik algoritmada sınır değerler programın en başında girildiği için, bu değerler uygun sonuçlar program sonucunda muhakkak verilecektir. İşte bu nedenden ötürü güç akışı hesaplamalarında genetik algoritma Newton-Raphson yönteminden daha üstün olabilir [31].

BÖLÜM 5. SONUÇLAR

Genetik algoritmalar ile çözümü zor veya imkansıza yakın problemler kolaylıkla çözülebilmektedir. Mühendislikte daha çok optimizasyon amaçlı kullanılır ve bunun sonucunda da geleneksel yöntemlere nazaran oldukça iyi sonuçlar vermektedir. Bu çalışmamızda genetik algoritma ve güç akışı algoritması genel olarak incelenmiş, amaç fonksiyonun minimize edilmesinde, genetik algoritmanın üstünlüğü vurgulanmıştır. Son bölümde örnek bir sistem üzerinde genetik algoritma uygulanmıştır.

KAYNAKLAR

[1] KARABOĞA,D., Yapay zeka optimizasyon algoritmaları, Nobel yayınevi, s: 73-106, Ocak 2011

[2] HOLLAND,J., Adaptation in natural and artificial systems, University of Michigan Press, Ann Arbor,1975

[3] GOLDBERG, D.E., Genetic algorithms in search, optimization and machine learning, Addison-Wesley, 1989

[4] BORIE, J.A., Modern control systems, Prentice-Hall International, 1986

[5] ZHANG, W., LIU, Y., Reactive power optimization based on PSO in a Partical power system. Power Eng. Society General Meeting, Vol.1,239-234, 6-10 June 2004

[6] RECHENBERG, I., Evolution strategie , Frommann-Holzboog, Stuttgart, 1973

[7] KOZA,J.R., Genetic programming: on the programming of computers by means of natural selection, MIT Press, 1992

[8] ÖZTÜRK, A., Güç sistemlerinde gerilim kararlılığının genetic algoritma ile incelenmesi, doktora tezi, Sakarya Üniversitesi, Fen Bilimleri Enstitüsü, 2007

[9] NABİYEV, V.V., Yapay zeka problemler, yöntemler, algoritma, vol.1

[10] IBA, K., Reactive power optimization by genetic algorithm, IEEE Trans. on Power Systems, vol.9,No.2, May 1994

[11] MIRANDA, V., RANİTO, J.V., PROENCA, L.M, Genetic algorithm in optimal multistage distribution network planning, IEEE PRE. Winter meeting

[12] WU, Q.H., and MA, J.T., power system optimal reactive power dispatch using evolutionary programming, IEEE trans. on power systems, vol.10,no.3, August 1995

[13] LEI,X., LERCH,E., POVH,D., Genetic algorithm solution to economic dispatch problem, ETEP,vol9,no.6,pp 347-352, Nov/Dec 1999

[14] LEE, K.Y., PARK, Y.M.,ORTIZ, J.L., A united approach to optimal and real and reactive power dispatch. IEEE Trans. on power apparatus and systems, May, 1985, pp.1147-1153

[15] ÖZTÜRK A., DUMAN S., Genetik algoritma kullanarak güç sistemlerinde optimal çalışma şartlarının belirlenmesi, Gazi Üniv. Mühendislik ve mimarlık dergisi, vol.24, no.3, s:539-548, 2009

[16] ELMAS, Ç., Yapay zeka uygulamaları, Seçkin yayıncılık, s:388-401, 2007

[17] ERCAN, Ö., ÖZTÜRK, A., genetic algorithm application for hydrometer tests, asian journal of chemistry, vol.21, no. 9, pp:6857-6868,2009

[18] Y.J.CAO, Q.H.WU, Teaching genetic algorithm using matlab, Int. J, Elec. Eng. Educ., vol.36, pp:139-153, ManchesterU.P, 1999

[19] BOLAT,B.,EROL,K.,O.,İMRAK,C.,E., Genetic algorithms in engineering applications and the function of operators, Journal of engineering and Natural Sciences, Sigma, 2004/4

[20] KWANG Y.L, XİAOMİN B., YOUNG-MOON P., optimization method for reactive power planning by using a medified simple genetic algorithm, IEEE Trans. on power systems, vol. 10, no.4, Nov 1995

[21] DEMİRÖREN, A., ZEYNELGİL, H.,L., Çevresel/ekonomik yük dağılımında genetic algoritmanın kullanılması, İ.T.Ü. Elektrik-elektronik fakültesi, elektrik müh. Bölümü, Maslak

[22] DURAİRAJ, S., KANNAN, P.S., DEVARAJ, D., Application of genetic, algorithm to optimal reactive power dispatch including voltage stability constraint, journal of energy&environment 4, 2005

[23] SUBBURAJ, P., SUDHA, N., RAJESWARI, K., RAMAR, K., GANESAN, L.,Optimum reactive power dispatch using genetic algorithm, academic open int. journal, ISSN 1311-4360

[24] DEJONG,K.A.,An Analysis of the Behaviour of a class of Genetic Adaptive Systems. PhD thesis, University of Michigan, Ann Arbour. Department of Computer and Communication Sciences.1975

[25] ARİFOĞLU, U., Güç sistemlerinin bilgisayar destekli analizi, alfa yayınevi, 2002

[26] ABACI, K., YALÇIN, M., UYAROĞLU, Y., Güç sistemlerinde farkli salinim barasi seçiminin gerilim kararliliği açisindan incelenmesi, Elektrik-Elektronik Mühendisliği Bölümü Sakarya Üniversitesi, Esentepe Kampüsü, Sakarya

[27] YAŞAR, C., FADIL, S., TAŞ, M.,A., YILDIZ, T., Görsel bir program : yük akışı analizi ve aktif güç optimizasyonu, elektrik-elektronik müh. 10. ulusal kongresi

[28] ARİFOĞLU, U., MATLAB simulink ve mühendislik uygulamaları, Alfa yayınevi, 2005

[29] DUMAN, C., Genetik Algoritma İle Tesis Yerleşimi,Tasarımı ve Bir Uygulama, Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü , 2007, İstanbul

[30] KILIÇ, U., AYAN, K., Optimal Güç Akışı Probleminin Çözümü İçin GA, MA ve YAK Algoritmalarının Karşılaştırılması, 6th International Advanced Technologies Symposium (IATS'11), 16-18 May 2011, Elazığ, Turkey

[31] KILIÇ, U., Elektrik Enerjisi İletim ve Dağıtımı dersi notları, Mehmet Akif Ersoy Üniversitesi, 2012

ÖZGEÇMİŞ

Nur SARMA, 12.12.1987'de İstanbul' da doğdu. İlköğretimini Eskişehir, orta öğretimini İstanbul ve lise eğitimini Sakarya'da tamamladı. 2005 yılında başladığı Sakarya Üniversitesi Elektrik-Elektronik mühendisliği bölümünü 2009 yılında bitirdi. 2009 yılında Sakarya Üniversitesi, aynı bölümün elektrik anabilim dalında yüksek lisansa başladı.