KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLER ENSTİTÜSÜ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANA BİLİM DALI

ADAPTİF BULANIK MANTIK DENETLEYİCİLER İÇİN YUMUŞAK HESAPLAMA TABANLI BİR OPTİMİZASYON ALGORİTMASI

YÜKSEK LİSANS TEZİ

Elektrik-Elektronik Müh. Reza Mirzapour

HAZİRAN 2014
TRABZON

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLER ENSTİTÜSÜ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

ADAPTİF BULANIK MANTIK DENETLEYİCİLER İÇİN YUMUŞAK HESAPLAMA TABANLI BİR OPTİMİZASYON ALGORİTMASI

Elektrik-Elektronik Müh. Reza Mirzapour

Karadeniz Teknik Üniversitesi Fen Bilimler Enstitüsünce
"ELEKTRONİK YÜKSEK MÜHENDİSİ"
Unvan Verilmesi için Kabul Edilen Tezdir.

Tezin Enstitüye Verildiği Tarih : 09.05.2014
Tezin Savunma Tarihi : 10.06.2014

Tez Danişmanı : Prof. Dr. İsmail Hakkı ALTAŞ

Trabzon 2014

Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü
Elektrik-Elektronik Mühendisliği Anabilim Dalında
Reza MIRZAPOUR tarafından hazırlanan

ADAPTİF BULANIK MANTIK DENETLEYİCİLER İÇİN YUMUŞAK HESAPLAMA TABANLI BİR OPTİMİZASYON ALGORİTMASI

başlıklı bu çalışma, Enstitü Yönetim kurulunun 20/05/2014 gün ve 1554 sayılı kararıyla oluşturulan jüri tarafından yapılan sınavda
YÜKSEK LİSANS TEZİ
olarak kabul edilmiştir.

Jüri Üyeleri

Başkan : Prof. Dr. İsmail Hakkı ALTAŞ .....

Üye : Yrd. Doç. Dr. Fatih M. NUROĞLU .....

Üye : Yrd. Doç. Dr. Tuğrul ÇAVDAR .....

Prof. Dr. Sadettin KORKMAZ
Enstitü Müdürü

ÖNSÖZ

Bu tez, Karadeniz Teknik Üniversitesinde Fen bilimler Enstitüsü Elektrik-Elektronik Mühendislik Anabilim Dalı, Elektrik mühendisliği Yüksek Lisans Programından hazırlanmıştır. Bu tezde Genetik Algortitması optimizeli Yapay sinir Ağlarını kulanarak Bulanık Mantık Denetleyici üzerinde üyelik fonksiyonlarının parametrelerinin ayarlanmasıyla elde edilen denetleyici ile Doğru Akım motorunun kontrolü Matlab/SIMULINK ortamında çalışılmıştır.

Öncelikle eğitimim süresince benden maddi ve manevi desteğini esirgemeyen annem Shamsi SAHRAYİ'ye teşekkürü borç bilirim.

Tezim süresince bana yalnızca iyi bir öğrencinin nasıl olacağını değil aynı zamanda iyi bir insanın nasıl olacağını öğretmeye çalışan ve tezim konusunda adım adım yol gösterip SEUL (Sürdürülebilir Enerji Uygulamaları Laboratuarı) labotatuarını açıp çalışmalarımı yapabilmek için gerekli ortamı sağlayan, tecrübeleriyle her zaman bana ve bu laboratuardaki POWENCON (Power, Energy, Control Research Group) elemanlarına her türlü yardımda bulunan danışmanım Prof. Dr. İsmail Hakkı ALTAŞ'a, sorunlarıma çözüm bulmak için benden yardımlarını esirgemeyen değerli hocam Doç. Dr. Halil İbrahim OKUMUŞ'a ve tezimde fikirleri ve yardımlarıyla yanımda olan kıymetli arkadaşım Yeliz YILDIRIM'a şükranlarımı sunarım.

Tezimin, sonraki çalışmalara faydalı olması dileklerimle.

Reza MIRZAPOUR

Trabzon 2014

TEZ BAYANNAMESİ

Yüksek Lisans tezi olarak sunduğum "ADAPTİF BULANIK MANTIK DENETLEYİCİLER İÇİN YUMUŞAK HESAPLAMA TABANLI BİR OPTİMİZASYON ALGORİTMASI" başlıklı bu çalışmayı baştan sona kadar danışmanım Prof. Dr. İsmail Hakkı ALTAŞ'ın sorumluluğunda tamamladığımı, örnekleri kendim topladığımı, başka kaynaklardan aldığım bilgileri metinde ve kaynakçada eksiz olarak gösterdiğimi, çalışma sürecinde bilimsel araştırma ve etik kurallara uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu yasal sonucu kabul ettiğimi beyan ederim. 09.05.2014

Reza MIRZAPOUR

İÇİNDEKİLER

Sayfa No

Yüksek Lisans Tezi

ÖZET

ADAPTİF BULANIK MANTIK DENETLEYİCİLER İÇİN YUMUŞAK HESAPLAMA TABANLI BİR OPTİMİZASYON ALGORİTMASI

Reza MIRZAPOUR

Karadeniz Teknik Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik-Elektronik Mühendisliği Anabilim Dalı
Danışman: Prof. Dr. İsmail Hakkı ALTAŞ
2014, 63 Sayfa

Günümüzde pekçok alanda hız kontrolünün gerektiği uygulamalar yapılmaktadır. Doğru akım (DA) motorları kontrolünün kolay olması, ucuz olması, küçük olması ve çok çeşitli olup gereken özelliğe göre pek çok ihtiyaca yanıt vermesi gibi özelliklerinden dolayı bu tür uygulamalarda kullanım açısından başı çekmektedir. Uygulamaya göre ihtiyaç duyulan motoru belirlemenin yanı sıra hangi kontrolörle kontrol edilmesi gerektiğini belirlemek de önemlidir. Bunu belirlerken geleneksel yöntemlerin bir çok uygulama için yeterli olmadığı ve yeni yöntemlerin gereksinim duyulduğu gerçeği göz önüne alınmalıdır. Çünkü geleneksel kontrol yöntemleri sadece lineer kullanımlara yanıt verebilmekte ancak lineer olmayan uygulamalar için yetersiz olmaktadırlar ayrıca çevredeki bozucu etkilerden çabuk etkilenmekte ve parametre değişikliklerinde takip etmekte güçlük çekmektedirler. Bu nedenler de uygulayıcıları Bulanık Mantık, Yapay Sinir Ağları gibi akıllı kontrolerlere yöneltmiştir.

Bu tezde de bunlar göz önüne alınarak bir kaç akıllı sistemin birarada kullanıldığı bir denetleyici tasarlanmıştır ve DA motoru, üyelik fonksiyon aralıkları Yapay Sinir Ağlarıyla ayarlanan bir Bulanık Mantıkla denetlenmiştir. Bu denetleyicideki Yapay Sinir Ağının ağırlıkları da Genetik Algoritma ile optimize edilmiştir. Tez üç aşamadan oluşmaktadır. İlk aşamada tezde bilinmesi gereken genel bilgiler giriş başlığı altında verilmiştir. İkinci kısımda önerilen bu denetleyicinin işleyişi hakkında ön bilgi verilmiştir. Son kısım ise Matlab/SIMULINK uygulamalarına ayrılmış ve ilk olarak akıllı denetleyicilerle geleneksel denetleyicilerin kıyaslaması için DA motoru PID ile kontrol edilmiş, daha sonra Bulanık Mantık, Yapay Sinir Ağları gibi akıllı sistemlerle kontrolü yapılmış, son olarak da önerilen denetleyici ile denetlenmiştir. Tezin sonuç kısmında ise bu çıktıların ve dolayısıyla denetleyicierin kıyaslaması yapılmıştır.

Anahtar Kelimeler: DA Motoru, Bulanık Mantık, Yapay Sinir Ağları, Genetik Algotirma

Master Thesis

SUMMARY

A SOFT COMPUTING BASED OPTIMIZATION ALGORITHM FOR AN ADAPTIVE FUZZY LOGIC CONTROLLER

Reza MIRZAPOUR

Karadeniz Technical University
The Graduate School of Natural and Applied Science
Electrical and Electronics Graduate Program
Supervisor: Prof. Dr. İsmail Hakkı ALTAŞ
2014, 63 Pages

Nowadays, speed control is essential for many applications. DC motor has priority for many speed control technics due to its features like it is easy to control, generally it has small structure and it has lots of different types to find proper one for the purpose of implementation. Not only choosing appropriate motor for the application, but also determining with which controller it should be controlled, is important. To determe this, it should be take into consideration the fact that conventional methods are not adequate for many applications and new technics are needed. Because conventional controllers can only respond to linear applications, but it is not sufficient for non-linear applications and it can get affected easily with the environmental distrubances and it can have difficulties to track for for different parameters. Those reasons lead many users to intelligent controllers like Fuzzy Logic and Artificial Neural Network.

In this study, DC motor is controlled with a Fuzzy Logic controller whose membership functions are tuned by Artificial Neural Network. The weights of Artificial Neural Network in this study is optimised with Genetic Algorithm. This thesis is fomed in three steps. In first steps, general knowledge which should be known are given with the title of introduction. In second part, general function of the controller which is suggested is described. Last part is for the Matlab/SIMULINK applications. In this part, firstly DC motor is controlled with PID in order to make comparision between conventional controllers and intelligent controllers, then it is controlled with some intelligent controllers such as Fuzzy Logic and Artificial Neural Network and lastly it is controlled with proposed controller. Related outputs are compared in the conclusion of the thesis.

Key Words: Direct current motor, Fuzzy Logic, Artificial Neural Network, Genetic Algorithm

ŞEKİLLER DİZİNİ

Sayfa No

TABLOLAR DİZİNİ

Sayfa No

SEMBOLLER VE KISALTMALAR DİZİNİ

Semboller

B : Manyetik alan

F : Manyetik kuvvet

ω_m : Endüvi açısal hızı

n : Devir sayısı

V_a : Endüviye uygulanan gerilim

R_a : Endüviye sargı direnci

E_a : Elektromotor kuvvet

V_S : Kaynak gerilimi

V_f : Uyarma gerilimi

R_f : Uyarma direnci

ϕ_f : Alan akısı

R_{fs} : Seri uyarma direnci

R_{fp} : Paralel uyarma direnci

e : Hata

de : Hata değişimi

du : Çıkış değişimi

Kısaltmalar

DA : Doğru Akım

PID : Proportional- Integral- Derivative

YSA : Yapay Sinir Ağları

BMD : Bulanık Mantık Denetleyici

GA : Gentik Algoritma

TF : Transfer Fonksiyonu

1. GİRİŞ

Doğru Akım (DA) motorları mekanik olarak basit yapıya sahiptirler. Çok çeşitli olmalarından dolayı çok sayıda ihtiyaca karşılık verebilmektedirler. Boyutlarının küçük olması nedeniyle de gitgide küçülen teknolojide önem kazanmaktadırlar. Matematiksel modelinin basit olması nedeniyle kontrol uygulamalarının vageçilmez elemanlarından biridir. Her ne kadar doğrudan şebekeden beslenememesi gibi bir dezavantajı olsa da, Doğru Akım motorları, kontrol uygulamalarında önemli bir yere sahiptirler.

DA motorlarının kontrolü kolay olsa da bazı uygulamalarda sistem karmaşıklaşabilmekte ve matematiksel modelinin çıkarımı hem zaman hem de enerji kaybına sebep olabilmektedir. Geleneksel yöntemlerde parametre ayarı yapmak da zaman almakta ve elde edilen uygun parametre değerleri bazı durumlarda gerçek hayata uygun olmayacak şekilde çok düşük ya da çok yüksek olabilmektedir. Ayrıca parametre değişikliklerinde geleneksel yöntemlerde yeniden ayar yapmak gerektiğinden ve bu yöntemler çevresel bozucu etkenler karşısında dayanıklı bir tavır sergileyememektedir. Bu sorunlarla başa çıkabilmek için akıllı sistemlere yönelim artmıştır. Akıllı denetleyiciler, matematiksel model çıkarma probleminin çözümü olmuştur. Çünkü akıllı denetleyicilerde tam bir matematikel modele gerek yoktur. Ayrıca parametre değişikliklerine ve çevresel bozucu etkilere karşı güçlüdürler. Bulanık Mantık, da akıllı bir sistem olduğu için matematiksel model gerektirmez, değişikliklere kararlı bir tutum sergiler ve takip yeteneği iyidir. Ancak Bulanık Mantıktaki üyelik fonksiyonlarının ayarlanması ve kural tablosunun belirlenmesi zorlu bir süreç olabilmektedir. Bu nedenle bir diğer akıllı sistem olan Yapay Sinir Ağı, Bulanık Mantık Denetleyici ile birleştirilerek etkisi arttırılabilir. Yapay Sinir Ağları (YSA) da matematiksel model gerektirmeyen akıllı bir sistemdir. Ancak Yapay Sinir Ağının sorunu, bazı durumlarda yeterli eğitim verisi (training data) nin toplanamamasıdır. Bu sebepler de bu iki akıllı sistemin beraber kullanılabilmesi için fikir vermiştir.

Bu tezin amacı Bulanık Mantık ve Yapay Sinir Ağlarını beraber kullanarak bu iki denetleyicinin olumsuz yönlerini olumlu hale getirmektir. Bu sebeple, DA motorunu kontrol etmek için kullanılan Bulanık Mantığın üyelik fonksiyon aralıkları, Yapay Sinir Ağlarıyla ayarlanmıştır. Böylece Bulanık Mantığın üyelik fonksiyonu belirleme sorunu bir nebze giderilmiştir. Bulanık Mantık üyelik fonksiyon aralıklarını ayarlayan Yapay Sinir

Ağının en iyi ağırlıkları da Genetik Algoritma ile ayarlanmış ve adaptif bir sistem elde edilmiştir.

Tezin yapısında, öncelikle DA Motoru ve PID (Proportional- Integral-Derivative), Bulanık Mantık, Yapay Sinir Ağları gibi bazı denetleyiciler ve optimizasyon yöntemi olan Genetik Algoritma hakkında genel bilgiler verilmiştir. Sonraki kısımda önerilen denetleyicinin yapısı açıklanmıştır. Son kısımda da Doğru Akım Motoru hız denetimi geleneksel yöntemlerle kıyaslama yapılabilmesi için PID denetleyici ile yapılmış, akıllı sistemlerin farkını ortaya koyabilmek için Bulanık Mantık ve Yapay Sinir Ağlarıyla ve son olarak da önerilen sistem ile Matlab/SIMULINK ortamında sabit yük ve değişken yük için benztimleri yapılmıştır.

Tezin sonucunda, Bulanık Mantıktaki parametre ayarlama sorununu ve Yapay Sinir Ağlarında gerekli eğitim verilerini elde etme sorununu çözebilen, hızlı yanıt verebilen ve değişiklikler karşısında daha dirençli bir denetleyici elde edilmiştir.

Doğru akım motoru kontrol uygulamalarında çok kullanılan bir elektrik motoru türü olduğu için bu konu geçmişten günümüze çok çalışılmıştır. Bununla ilgili sayısız makaleler bulunmaktadır. Bu çalışmaların herbirini inceleyebilmek mümkün olmasa da son zamanlarda yapılan ve daha çok akıllı sistemlerle olan Doğru Akım Motoru kontrol uygulamalarını incelemekte fayda olacaktır.

A.M. Sharaf ve İ.H.Altaş DA motor sürücüleri için yeni bir kontrolör önerdi. Bu kontrolör, değişken yapılı çok bölgeli PID türü olarak sınıflandırılıyordu. Bu denetleyicinin, değişken ve bulanık tür mekaniksel yükler durumunda da DC motor sürümü için uygun olduğunu savundular [1].

İsmail H. Altaş hareket eden nesnelerin takibi için bulanık kümeler ve Bulanık Mantık yaklaşımlarını kullanan bir kontrol algoritması geliştirdi. Denetleyiciyi sıfır hatayla, hareket eden nesneleri takip edecek şekilde tasarladı. Tasarımında kullandığı radar ve dişli sisteminde Doğru Akım motorunu kullandı [2].

B. Cakir ve arkadaşları serbest uyartımlı DA çekişli motor için yeni bir control yöntemi önerdiler. Çalışmada simüle simülasyon yaptılar ve çıktılarını geleneksel yöntem olan PID kontrolör ile karşılaştırdılar. Önerilen denetleyicinin, geçici ve kararlı durum performansının PID denetleyiciye nispeten daha üstün olduğunu ifade ettiler [3].

Ahmed Rubaai ve Raj Kotaru DC motor hız control sorununu geniş anlamda ele aldılar ve non-lineer haritalama yapabilmesi için ileri beslemeli Yapay Sinir Ağını kullandılar. Bunun için online olarak rastgele Yapay Sinir Ağı eğitimi yaptılar ve bununla

da Yapay Sinir Ağı, belirsizlikleri giderilmiş oldu. Çalışmalarında, öğrenme sürecinin sabitliğini oluşturması boyunca, öğrenme oranını mümkün olduğunca geniş tutan adaptif bir öğrenme algoritması önerdiler. Önerilen bu sistemin verimliliğini gösterdiler ve önerdikleri sistemin öğretme hızını arttırdığını, ayrıca gürültülü ortamlarda eğitilen Yapay Sinir Ağının da umut verici sonuçlar verdiğini iddia ettiler [4].

Ieroham S. Baruch ve arkadaşları yeniden eğitilebilir Yapay Sinir Ağlarını Zamanla geri yayılım (Backpropogation Throug-Time) öğrenme algoritmasıyla beraber, DA motor sürücüsünün gerçek zamanlı parametre kestirimi ve adaptif kontolü için uyguladılar. Elde etikleri sonuçlar, belirlenen kontrol yönteminin uygulanabilirliğini ve yeniden eğitilebilir Yapay Sinir Ağlarının kalitesini onayladığını belirttiler [5].

Phan Quoc Dzung ve Le Minh Phuong Yapay Sinir Ağlarının hız tahmini ve hız kontrolünde kullandılar. Bu YSA kontrolörü iki kısımdan oluşturdular. Biri, hız tahmini yapan hız tahmincisi diğeri de bir çevirici kontrol sinyali oluşturan YSA kontrolörü idi. Bu sinirler Levenberg-Marquardt geri yayılım yöntemiyle eğitiliyordu. Yapay Sinir Ağları standart gizli tabakalarında sigmoid içeren çıkış tabakasında purelin içeren ileri beslemeli YSA idi. Yapay Sinir Ağlarının geleneksel yöntemelere göre daha iyi sonuç verdiğini gördüklerini savundular [6].

M. R. Mosavi, A. Rahmati ve A. Khoshsaadat Bulanık Mantık ve Yapay Sinir Ağlarının hibrit bir şekilde kullanıldığı Adaptive Network-based Fuzzy Inference System (ANFIS) kullanarak DA motoru hız kontrolü yaptılar. Sürücü performansı için hata geri yayılımı(Error back propogation) öğrenme metodu yerine Bulanık Mantık temelli online denetimli öğrenme algortimasını kullandılar. Geleneksel Bulanık Mantık yönteminden daha az Bulanık kural gerektirdiğnden uygulanmasının daha kolay olduğunu savundular ve benzetim sonuçlarını geleneksel PID denetleyici veya diğer denetleyicilerle de elde edip karşılaştırdılar [7].

Kamel Sabahi DA motoru kontrolü için yeni bir tasarım öne attı. Bu tasarımda, Yapay Sinir Ağları ve PID denetleyicilerini aynı zamanda sistemi kontrol etmek için kullandı. PID'yi Yapay Sinir Ağlarını eğitmek için kullandı ve böylece DA motorunu kontrol etti. Uygulamanın önemini göstermek için de benzetimini yaptı [8].

Ahmad Faramarzi ve Kamel Sabahi doğrusal olmayan DA motorunun hız takibini tekrarlamalı Bulanık Yapay Sinir Ağlarıyla yaptılar. Hem Bulanık Mantığın hem de Yapay Sinir ağlarının olumlu yanlarına sahip ve yapısında geçmiş bilgiyi depolayabilen hafızaya sahip aynı zamanda doğrusal olmayan ve adaptif davranabilen bir yöntem oluşturmuş

oldular. Çıktılarıyla da yöntemlerinin önemini ve olumu yanlarını göstermeyi amaçladılar [9].

Aiping Shi ve arkadaşları otomatik kapılardaki DA motor kontrolünü Bulanık Mantık ve PID'yi birleştirerek yapmayı amaçladılar. Matematiksel modelini oluşturarak PID ve Bulanık PID yöntemlerini Matlabda simüle ettiler. Elde ettikleri benzetim sonuçlarına göre Bulanık PID'nin hızlı yanıt verdiğini, düşük aşma yaptığını, kararlı ve dayanıklı olduğunu ifade ettiler [10].

Bu tez çalışmasında literatürden farklı olarak bulanık kontroler tasarımında üyelik fonksiyonları YSA tarafından adaptif olarak ayarlanıyor. Diğer taraftan YSA' nın ağırlıkları GA ile optimize ediliyor. Elde edilen sonuçlar gösteriyor ki yapılan çalışma parametre değişikliklerine daha dayanıklıdır.

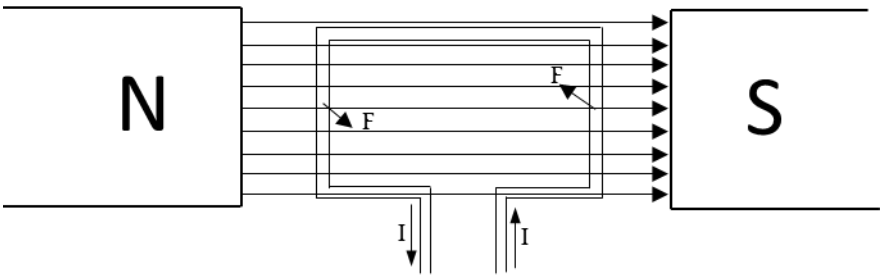
1.1. DA Motorlar

DA motorlar, doğru akımı kullanarak elektrik enerjisini mekanik enerjiye çeviren makinalardır. Doğru akım makinelerinin temel ilkeleri ilk defa Michael Faraday tarafından ortaya atılmıştır. Bu prensiplerden yola çıkan Belçikalı mühendis Grame, 1873 yılında ilk endüstriyel doğru akım dinamosunu yapmayı başarmıştır. Günümüzde ise her ne kadar alternatif akım motorlarının endüstride kullanımı artmışsa bile DA motorların ucuz ve küçük olması kaynağı olan doğru akımın pil, akü, batarya gibi kimyasal enerji elektrik enerjisine çeviren küçük, taşınabilir kaynaklardan elde edilebilir olması ve bu sayede de küçük elektrik aletlerde kolayca kullanılabiliyor olması DA akım motorunun kullanımında azalmaya sebep olmamış aksine kontrol edilebilmesinin kolay olması nedeniyle gelişen kontrol uygulamalarıyla birlikte kullanımı daha da yaygınlaşmıştır [11-14].

1.1.1. DA Motor Çalışma Şekli

İçinden doğru akım geçen bir iletken, N-S mıknatıs kutupları arasındaki manyetik alan içine yerleştirildiği zaman iletkenin hareket ettiği görülür. Alan kutupları tarafından iletkenin etrafında meydana gelen bu alan solda ana alana ters; sağda ise onu kuvvetlendirecek yöndedir. Bu da bir kuvvet meydana getirir ve çubuğu alanın dışına iter [13].

Aynı şey iletken çubuk yerine Şekil 1‘ deki bir bobine uygulanırsa, kenarlarına bir kuvvet uygulandığı görülür ve bu manyetik kuvvet (F) in etkisiyle döner. Ancak bu manyetik alan (B) içerisinde bulunan bobbin yatay konuma geldiğinde uygulanan zıt kuvvetlerin birbirlerine eşitliğinden durur. Bobinin durmasını önlemek ve sürekli aynı yönde dönmesini sağlamak için kenarların bir kutubun etkisi altından kurtulup diğerinin etkisi altına girerken içerisinden geçirilen akımın yönünün değiştirilmesi gerekmektedir, bunu yapabilmek için de bobinin uçları kollektöre bağlanır.

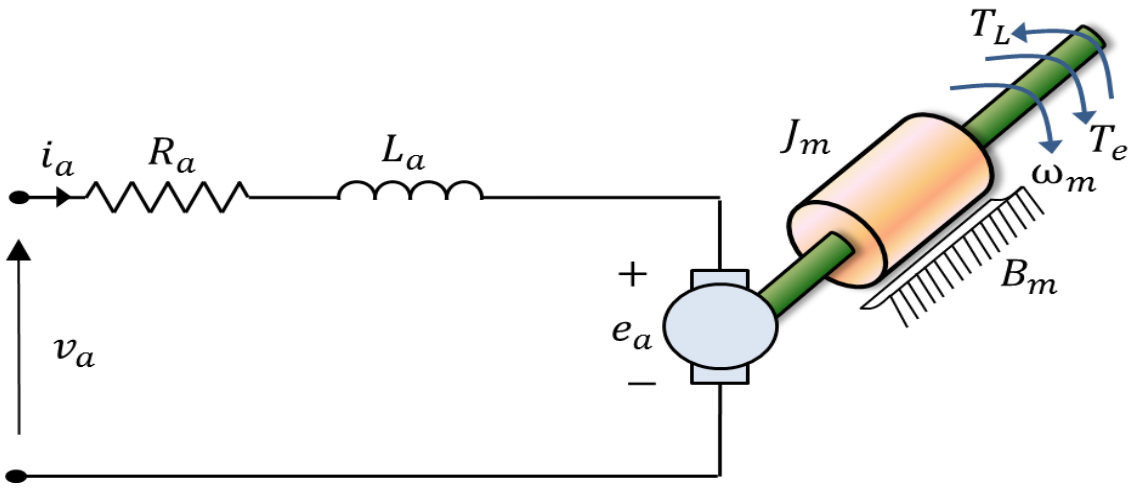


Şekil 1. DA motoru çalışma şekli

Bu kutuplar arasına tek bir bobin yerine silindirik bir demir nüve üzerine açılmış oluklara yerleştirilen birçok bobin kullanılır ve kollektör ve fırçalarla düzgün şekilde dönmesi sağlanırsa bir doğru akım motoru elde edilmiş olur.

1.1.2. DC Motorun Matematiksel Modeli

DA motorunun hız kontrolünün yapılabilmesi için öncelikle matematiksel modellemesini yapmak ve transfer fonksiyonunu elde etmek gerekir. Şekil 2 [14]’ de gösterilen DC motorun yapısına göre matematiksel denklemeleri çıkarılabilir.



Şekil 2. DC motor yapısı

Elektriksel denklemleri (1) - (3) denklemlerindeki gibi

$$e_a(t) = -R_a i_a - L_a \frac{di_a(t)}{dt} + v_a(t) \tag{1}$$

$$e_a(t) = K_f \omega_m(t) \tag{2}$$

$$K_f \omega_m(t) = -R_a i_a - L_a \frac{di_a(t)}{dt} + v_a(t) \tag{3}$$

Mekaniksel denklemleri (4) - (6)' daki gibi yazılabilir.

$$T = J \frac{d\omega_m(t)}{dt} + B\omega_m(t) + T_L \tag{4}$$

$$T = K_f i_a(\mathrm{t}) \tag{5}$$

$$K_f i_a(\mathrm{t}) = J \frac{d\omega_m(t)}{dt} + B\omega_m(t) + T_L \tag{6}$$

Bunların s domeninde yazılmasıyla (7) ve (8) elde edilir.

$$V_t(\mathrm{s}) - K_f \omega_m(\mathrm{s}) = R_a I_a(\mathrm{s}) + L_a s I_a(\mathrm{s}) \tag{7}$$

$$K_f I_a(\mathrm{s}) = Js\omega_m(\mathrm{s}) + B\omega_m(\mathrm{s}) \tag{8}$$

Bu denklemlerden hareketle transfer fonksiyonu denklem (9)' daki gibi olur.

$$\frac{\omega_m(\mathrm{s})}{V_a(\mathrm{s})} = \frac{K_f}{K_f{}^2 + (\mathrm{R_a} + \mathrm{L_a s})(\mathrm{Js} + \mathrm{B})} \tag{9}$$

Yukardaki eşitliklerde açısal hız ve elektrik akımı durum değişkenleri olara kabul edilirse; armatör gerilimi giriş ve açısal hız da çıkış olacağından durum denklemleri (10)-(13) gibi ifade edilebilir.

$$\begin{bmatrix} \dot{\omega}_m \\ \dot{i} \end{bmatrix} = \begin{bmatrix} -\frac{B}{J} & \frac{K_f}{J} \\ -\frac{K_f}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} \omega_m \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix} v_a \tag{10}$$

$$\dot{\omega}_m = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \omega_m \\ i \end{bmatrix} \tag{11}$$

Durum denklemleri standart formatta (12) ve (13)' da verilmiştir.

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{12}$$

Burada D=0 olup, durum vektörü

$$\dot{x} = \begin{bmatrix} \dot{\omega}_m & \dot{i} \end{bmatrix}^T \tag{13}$$

olarak, durum değişkenleri de $x = \begin{bmatrix} \omega_m & i \end{bmatrix}^T$ olarak alınmaktadır. Sistem çıkış ise $y = \omega_m$ dir.

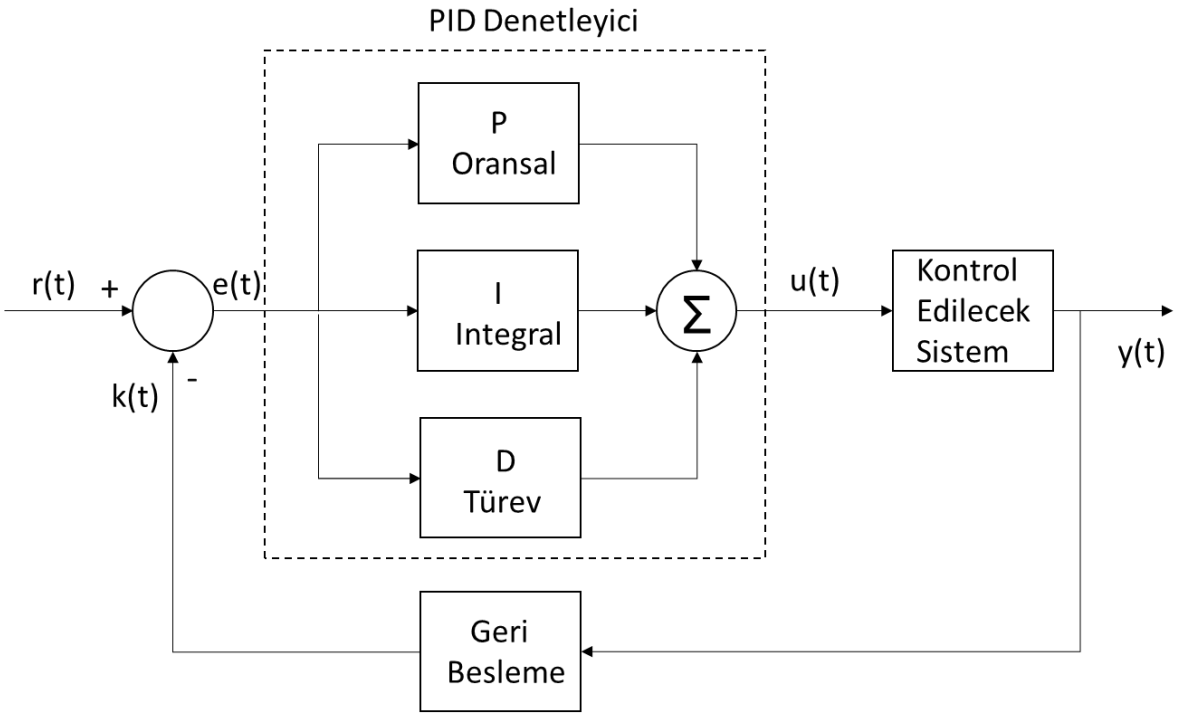
1.2. Denetleyiciler

Tezde kullanılan denetleyicilerle ilgili genel bilgiler aşağıda verilmiştir.

1.2.1. PID Denetleyiciler

PID denetleyiciler günümüzde endüstriyel uygulamalarda en çok kullanılan denetleyicilerden biridir. Basit yapısı nedeniyle, gelişmiş farklı kontrol uygulamalarına ragmen çok tercih edilen bir yöntemdir. PID denetleyici P oransal, I integral ve D türev denetleyicilerden oluşmaktadır. Bunların her birinin farklı şekillerde sisteme etkileri vardır (Tablo 1). Örneğin, integral denetleyici büyük yük değişimlerinde oluşan oransal ofseti azaltır, türev denetleyici ise osilasyon eğilimini azaltır ve hata sinyalini önceden sezer [15]. Bu denetleyicilerin hepsi toplanıp aynı sistemde kullanıldığıda ise herbirinin iyi yönlerini birleştirilmiş böylece de her birine oranla daha iyi bir denetleyici elde edilmiş olur. Bu nedenle P, PD, PI denetleyicilere oranla PID daha fazla tercih edilmektedir.

PID denetleyiciler Şekil 3'te gösterildiği gibi oluşturulup sisteme uygulanmaktadırlar.



Şekil 3. PID kapalı çevrim kontrolü

PID denetiminde referans değer ve çıkış değeri arasındaki fark alınarak elde edilen hata değerinin türevi, integrali alınmış ve oransal denetleyiciden geçmiş halleri toplanır. Hatanın türevini almak aşmayı azaltırken, integralini almak da kalıcı hatayı sıfırlamış olur.

Bunların denklemlerle de ifade etmek mümkündür. Sistemin hatası denklem (14)' deki gibidir.

$$e(t) = k(t) - \mathrm{r}(t) \tag{14}$$

Ideal bir PID kontroler çıkışı denklem (15) şeklinde olacaktır.

$$u(t) = K_P e(t) + K_I \int_0^t e(t)dt + K_D \frac{de(t)}{dt} \tag{15}$$

Başlangıç koşulları sıfır alınırsa s-uzayında

$$U(s) = K_P E(s) + \frac{K_I}{s} E(s) + sK_D E(s) \tag{16}$$

olur. Transfer fonksiyonu denklem (17)' deki gibi elde edilir.

$$\frac{U(\mathrm{s})}{E(\mathrm{s})} = \frac{s^2 K_D + sK_P + K_I}{s} \tag{17}$$

PID denetleyicinin K_P, K_D veK_I değerleri sistemin cevabını farklı şekillerde etkileyebilmektedir. Tip sıfır bir sisteme uygulanması halinde oluşabilecek etkiler Tablo 1'de gösterilmiştir.

Tablo 1. Tip sıfır sisteme uygulanması halinde K_P, K_I ve K_D parametrelerinin sistemin yükselme zamanı, aşması, oturma zamanı ve kalıcı durum hatasına etkisi

P,I,D Denetleyicilerin Sisteme Etkileri				
	Yükselme Zamanı	**Aşma**	**Oturma Zamanı**	**Kalıcı Durum Hatası**
K_P	Azaltır	Arttırır	Az Arttırır	Arttırır
K_I	Azaltır	Arttırır	Arttırır	Yok Eder
K_D	Az Değiştirir	Azaltır	Azaltır	Az Değiştirir

Yukarıda belirtilen PID eşitlikleri yeniden K_P, K_D ve K_I değerleriyle denklem (18) ’le formülize edilebilir ve denklem (19)’ la Transfer Fonksiyonu (TF) elde edilebilir.

$$u(t) = K_P e(t) + K_I \int_0^t e(t)dt + K_D \frac{de(t)}{dt} \tag{18}$$

$$TF = \frac{sK_P + K_I + s^2 K_D}{s} \tag{19}$$

PID denetleyiciler günümüzde pek çok alanda kullanılmaktadırlar. Sıvı seviye kontrolünde, motor kontrolünde, sıcaklık denetleme sistemleri gibi pek çok alanda ve özellikle sıcaklığın önemli olduğu nükleer santraller, tıbbi elektronik alanlarında çokça tercih edilmektedirler. Son zamanlarda daha çok gelişen akıllı kontrolerlerle birlikte kullanılıp olumsuz yanları elenerek daha hassas kontrol yapabilmekte ve daha iyi sonuç verebilmektedirler.

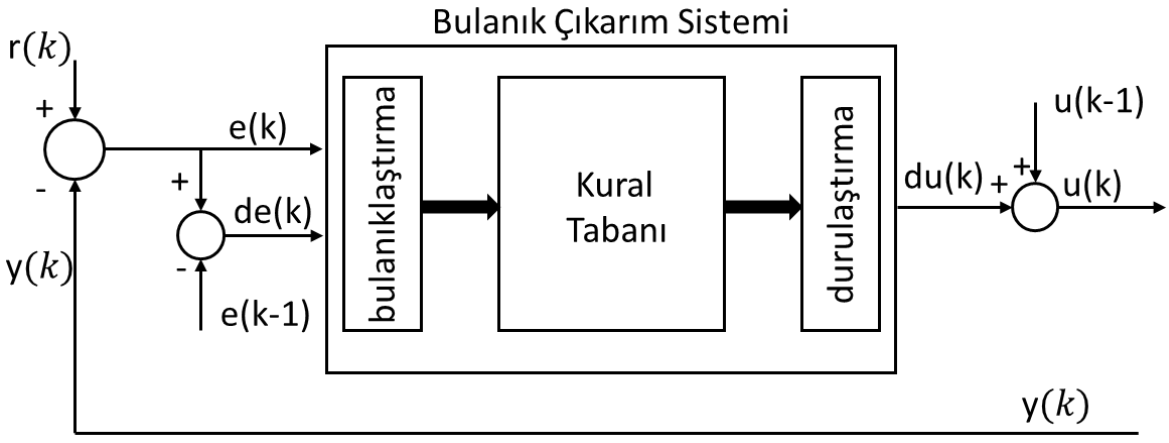
1.2.2. Bulanık Mantık Denetleyici

Günümüzdeki modern kontrol tekniklerinde kararsızlık ve belirsizliklerle başa çıkabilmek çok büyük önem taşımaktadır. Bulanık mantıkta bulunan belirsizliklere dayanarak oluşturulan üyelik fonksiyonları kontrol sistemlerindeki kararsızlık ve belirsizliklerin giderilebilmesine hız kazandırmıştır. Bulanık kümelere dayanan bulanık

mantık denetleyici, insanların bilgilerini ve tecrübelerini bulanık mantık kuralları adı altında harfsel ifaderle belirten bir denetleyicidir. İstenen çıkışı elde edebilmek için gereken girişlerin ayarlanması sistem dinamiği ve parametre çeşitliliği bilgisi gerektirmeden sadece gözlemlemeye göre ayarlandığı için de sistem kesin bir matematiksel modeli gerektirmez [16-21]. Bu nedenle de Bulanık Mantık Denetleyici(BMD), matematiksel model ve parametre tahmini gerektirmeden karmaşık, belirsiz ve lineer olmayan sistemlerde istenen kontrolü sağlayabilme yeteneği sayesinde parametre değişikliğinden etkilenmeyerek adaptif ve liner olmayan bir yapı sergiler [22]. Bulanık mantık denetleyici, ona uyumlu bilgisayar ve çip geliştirilmesiyle birlikte özellikle liner olmayan sistemlerin kontrolünde çok önemli ve kullanışlı bir yapı haline gelmiştir [23-27].

Bulanık Mantık Denetleyici çalışmaları Lotfi Zadeh [23] tarafından ortaya atılmasından itibaren sürekli hız kazanmıştır. Mamdani ve arkadaşları tarafından yapılan Bulanık uygulamasıyla sistem kontrolü Bulanık Mantığın gerçek hayatta uygulamalarını tetiklemiştir [28].

Bulanık mantık, bulanıklaştırma, kural tabanı ve durulaştırma olmak üzere üç aşamadan oluşur. Bulanıklaştırmada adımında, giriş sinyallerinin yani asıl sinyal ve her bir örneklemedeki asıl sinyal değişikliklerinin krisp değerleri bulanık verilere çevrilir. Daha sonra çıkış sinyaline denk gelen bulanık mantık sayılarının belirlenmesi için bu bulanık veriler kural tabanında kullanılır. Son olarak da çıkış sinyalinin karşılığı olarak oluşturulmuş bu birleşik bulanık altkümeler, krisp verilere dönüştürülerek durulaştırılır.



Şekil 4. Bulanık mantık kontrolörün temel yapısı

Şekil 4’ de görüldüğü gibi Bulanık Çıkarım Sistemi biri hatadan (20) diğeri hata değişiminden (21) oluşan iki giriş içerir.

$$e(k) = y(k) - r(k) \tag{20}$$

$$de(k) = e(k) - e(k-1) \tag{21}$$

Bu hata ve hata değişimi Bulanık Bloğun girişleri olarak kullanılacakları için önce bulanıklaştırılır, sonra kural tablosu uygulanır ve daha sonra durulaştırılır.

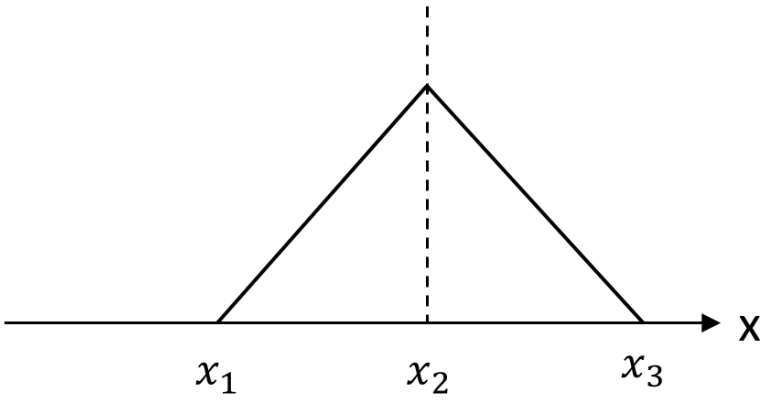
1.2.2.1. Bulanıklaştırma

Bulanık Mantığın ilk adımı olan bulanıklaştırmada krisp değerde olan bu iki hata e ve hata değişimi de girişi, Bulanık Mantıkta kullanılabilir bulanık kümelere çevrilmesi gerekir. Krisp sayıları bulanık değerlere dönüştürmek için üçgen, gauss, yamuk, çan gibi pek çok farklı şekillerde üyelik fonksiyonları kullanılabilir[29]. Ancak bunlardan üçgen şeklinde olanında üyelik derecelerini yakalamak daha kolay olduğu için üçgen yapılı bulanık kümelerin kullanımı daha yaygındır.

Üçgen üyelik fonksiyonunu oluşturmak için (22)’deki fonksiyon kullanılır.

$$\mu(x) = \max\left[\min\left(\frac{x - x_1}{x_2 - x_1}, \frac{x_3 - x}{x_3 - x_2}\right), 0\right] \tag{22}$$

Burada x, tanımlı uzaydaki değişkeni ; x_1, x_2 ve x_3 ise Şekil 5’ te verildiği gibi üçgen üyelik fonksiyonuna ait parametreleri ifade etmektedir.

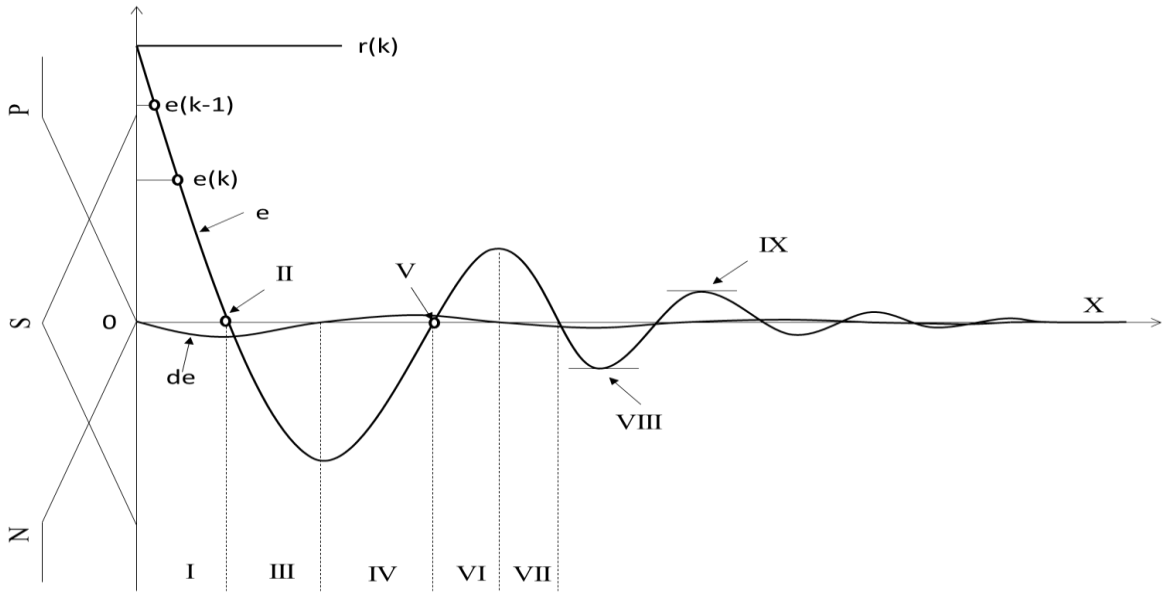


Şekil 5.Üçgen üyelik fonksiyonu

1.2.2.2. Kural Tabanı

Bulanık kuralları belirleyebilmek için çıkış, hata ve hata değişimi gözlenmeli ve hatayı en aza indirebilmek ve çıkışı da istenen çıkışa uydurabilmek için neler yapılması gerektiğine karar verilmelidir. Bunun için de hata ve hata değişimi farklı çalışma bölgelerine ayrılarak bu bölgelerde istenen çıkış için alınan çıkışın azaltılması gerektiğine ya da çoğaltılması gerektiğine karar verilmelidir. Çıkış arttırılması gerekiyorsa çıkış değişimi du pozitif olmalı, çıkış azaltılması gerekiyorsa da du negatif olmalıdır.

İkinci dereceden bir sistem için genelleştirilmiş adım cevap hatası Şekil 6‘ daki gibi krisp değerler olan hata ve hata değişimi, negatif (N), sıfır(S), pozitif(P) olmak üzere 3 bulanık alt kümeye bölünür.



Şekil 6. Çalışma bölgeleri ve genelleştirilmiş ikinci dereceden bir sistem için hata ve hata değişiminin cevaplama süresinin bölümlenmesi

Bu kurallar çıkış değişimi du'nun işaretini bulabilmek için aşağıdaki şekilde uygulanır.

1.Bölge: e = '+' ve de= '-'. Hata pozitiftir ve sıfıra doğru azalmaktadır bu nedenle du hatayı azaltmak için pozitif yapılır.

2.Bölge: e = '0' ve de= '-'. Hata sıfırdır ve negatif yöne doğru gitgide artmaktadır, bu nedenle hatayı azaltmak için negatif du gerekmektedir.

3.Bölge: e = '-' ve de= '-'. Hata negatiftir ve azalmaktadır bu yüzden du yine negatif olmalıdır ki hatayı sıfıra doğru azaltsın.

4.Bölge: e = '-' ve de= '+'. Hata hala negatiftir ama azalmaktadır. Negatif hatayı azaltabilmek için negatif du vermeye devapm etmek gerekir.

5.Bölge: e = '0' ve de= '+'. Hata sıfırdır ancak pozitif yönde artmaktadır. Pozitif du hatayı azaltacaktır.

6.Bölge: e = '+' ve de= '+'. Pozitif bir du, pozitif olan ve artan hata için uygundur.

7.Bölge: e = '+' ve de= '+'. 1. Bölgedeki gibi hata pozitiftir ve sıfıra doğru azalmaktadır bu nedenle pozitif du uygulanır.

8.Bölge: e = '-' ve de= '0'. Hata negatiftir ve bir değişiklik olmadığı için sabittir. Bundan dolayı, hatayı azaltmak için negatif du uygulanmalıdır.

9.Bölge: e = '+' ve de= '0'. Hata pozitiftir ve sabittir. Pozitif bir du hatayı azaltacaktır.

10.Bölge: e = ‘0’ ve de= ‘0’. Hem hata hem de hata değişimi sıfırdır ve sabittir. Kontrol sinyalinde hiçbir değişiklik gerekmediğinden du sıfıra ayarlanır.

Yani du nun alması gereken işaretle ilgili bütün bu kurallar aşağıdaki tabloda listelenmiştir. Tablo 2 kısaca eğer e sıfırsa du’ nun işareti de’ nin işaretiyle aynı, e sıfır değilse e’ nin işaretiyle aynıdır şeklinde özetlenebilir.

Tablo 2. Temel kontrol hareketinin yönü

Çalışma Bölgeleri										
	I	II	III	IV	V	VI	VII	VIII	XI	X
e	+	0	-	-	0	+	+	-	+	0
de	-	-	-	+	+	+	-	0	0	0
de	+	-	-	-	+	+	+	-	+	0

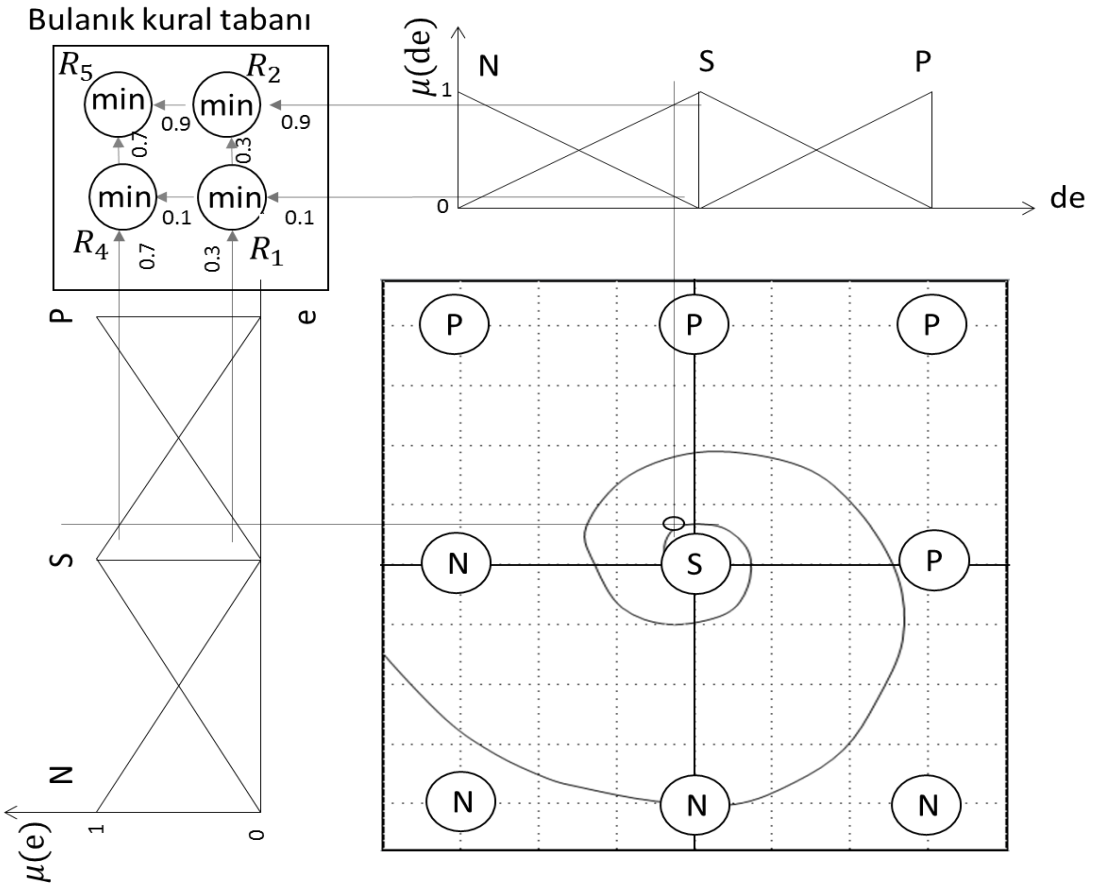
Yukarıdaki bölgelerin tarifinde görüldüğü gibi pozitif (P), negative (N) ve sıfır (S) olmak üzere üç seçenekten bahsedilmiştir ve buna göre Tablo 3’deki gibi 9 kurallı kural tablosu oluşturulabilir. Bu, 9 kurallı tablo bir tablo belki bir çok uygulama için yeterli olacaktır ancak bazı uygulamalar için giriş işaretinin üçten fazla bölgeye ayrıldığında, daha fazla kurallı bir kural tablosu gerekebilir.

Tablo 3. Kural tablosu

E \ DE	N	Z	P
P	P 1	P 2	P 3
Z	N 4	Z 5	P 6
N	N 7	N 8	N 9

Şekil 7’ deki giriş uzayının üç bölgeye ayrılmasında, e’ nin de’ ye çiziminde bölümleme önemlidir çünkü e ve de’nin sınırları birbirlerinden farklıdır. Bu nedenle bölümlemeye karşılık gelen bulanık kümeler de e ve de için farklı olacaktır. Eğer her ikisinde de aynı ölçekleme kullanılsaydı sıfıra yakın değerlerde olan de, e ye göre çok

değişemezdi. Çıkış uzayı du'dan bir nokta belirleyip ona göre kural düzenlemesi yapacak olursak aşağıdaki aşamalar sonuca götürecektir.



Şekil 7. Çıkış uzayı du'nun başlangıç kural düzenlemesi

Seçili noktanın e ve de' deki karşılığı için, Şekil 6' daki kural düzenlemesinden faydalanarak Tablo 2'den kural tabanı için karşılıkları bulunursa 1, 2, 4, ve 5. kurallar bu noktaya uyan kurallardır. Çünkü

K_1: EĞER e Pozitif, de Negatifse du Pozitiftir

K_2: EĞER e Pozitif, de Sıfırsa du Pozitiftir

K_4: EĞER e Sıfır, de Negatifse du Negatiftir

K_5: EĞER e Sıfır, de Sıfırsa du Sıfırdır

Bunlara göre de üyelik fonksiyonları olarak üçgen üyeliği seçilik ve min operatörü kullanılırsa üyelik fonksiyonlarındaki karşılıkları

$\mu_{K1}(du) = \min(\mu_P(e), \mu_N(de)) = (0.1, 0.3) = 0.1$

$\mu_{K2}(du) = \min(\mu_P(e), \mu_S(de)) = (0.3, 0.9) = 0.3$

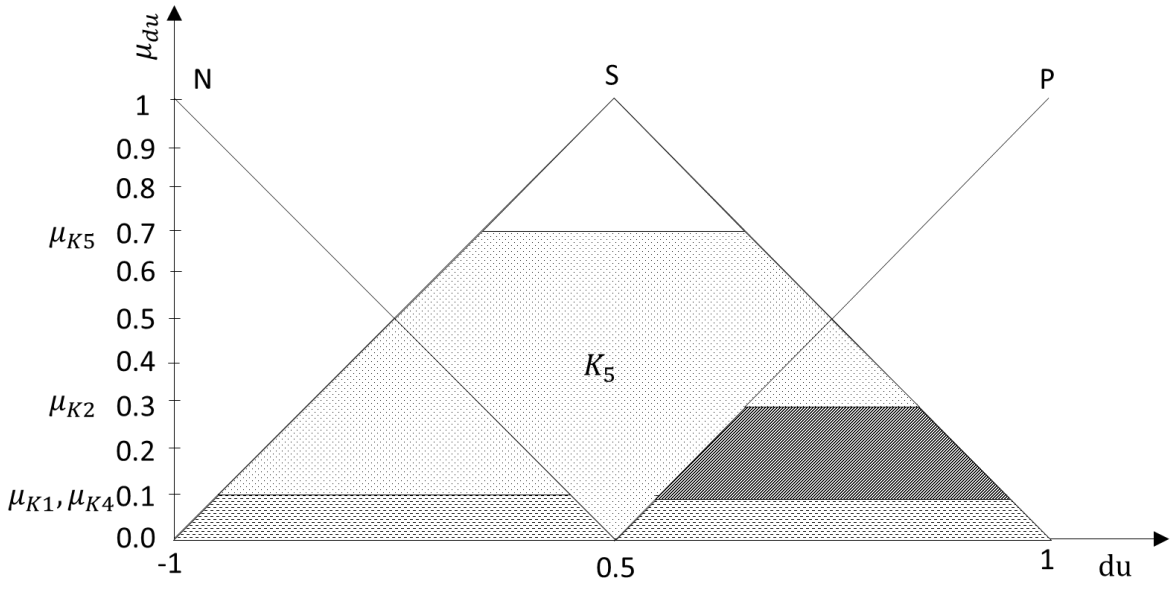
$\mu_{K4}(du) = \min(\mu_S(e), \mu_N(de)) = (0.1, 0.7) = 0.1$

$\mu_{K5}(du) = \min(\mu_S(e), \mu_S(de)) = (0.7, 0.9) = 0.7$

1.2.2.3. Durulaştırma

Bütün bu kural düzenlemelerinden ve bulanık çıkışın elde edilmesinden sonra son aşama olarak elde edilen bu bulanık çıkışın krisp değerlere dönüştürülmesi gerekir.

Bunu yapabilmek için de kullanılan pek çok yöntem vardır ancak Şekil 8' de görülen orta alan yöntemi bulanık mantık uygulamalarında daha çok tercih edilenidir.



Şekil 8. du uzayında bulznık bölümlemeye karşılık gelen üyelik fonksiyonları

Alanların merkezi yöntemi kullanılarak kesin sonuç elde edilebilir.

$$DU_K(k) = \frac{\sum_{i=1,2,4,5} \mu_{K_i}(du_K)\, du_K(K_i)}{\sum_{i=1}^{4} \mu_i(uV_K)} \tag{23}$$

$$DU_K(k) = \frac{0.1(0.0) + 0.7(0.5) + 0.1(0.0) + 0.3(0.0)}{0.1 + 0.7 + 0.1 + 03} = 0.29166$$

$du_K(K_i)$, K_i kuralı için kural karar tablosundaki bir çıkış olan maksimum üyelik derecesiyle alakalı kesin du değeridir.

1.3. Yapay Sinir Ağları

Yapay Sinir Ağları, çok miktarda ağırlaştırılmış bağlantılar üzerinden birinden diğerine sinyal göndererek haberleşen bir dizi basit işlemci birimleri içeren bir hesaplama modelidir. İnsan beyninden esinlenilerek tasarlanmıştır. İnsan beynindeki biyolojik sinir hücreleri, dentrit adındaki alıcılarla bilgileri toplar ve alınan bu verileri aksonlar vasıtasıyla gönderir, akson dallarının sonundaki sinapslarla başka bir sinir hücresine veriyi iletebilir. Sinapsların etkinliği değiştirilerek öğrenme meydana gelir ve böyle bir sinir hücresinin diğeri üstündeki etkisi değişmiş olur. Beyin hücreleri gibi yapay sinir ağları da işlem birimi yani yapay sinir hücreleri ve aralarında bağlantılar yani ağırlıklar içerirler. İşlem birimleri, gelen bilgiyi diğer birimlerle olan bağlantılarıyla aktarır. Yapay sinir ağlarının en önemli özelliği, örneklerden öğrenerek çözülmesi gereken probleme göre uyarlayan, adaptif yapısıdır. YS Ağlarının, güçlü örüntü sınıflandırma ve örüntü tanıma yetenekleri vardır. Giriş bilgisiyle hedef arasında ilgili örüntüleri öğrenebilir ve tanıyabilirler bu nedenle de temel verileri açık olmayan sistemler için çok kullanışlıdırlar. Yapay Sinir Ağları eğitildikten sonra, alakası olmayan girişlerin de çıkışlarını tahmin edebilir. Beynin taklidi oldukları için karmaşık ve linear olmayan sorunlarda hatta kesin olmayan ve gürültülü sistemlerde de işlem yapabilirler [31-34].

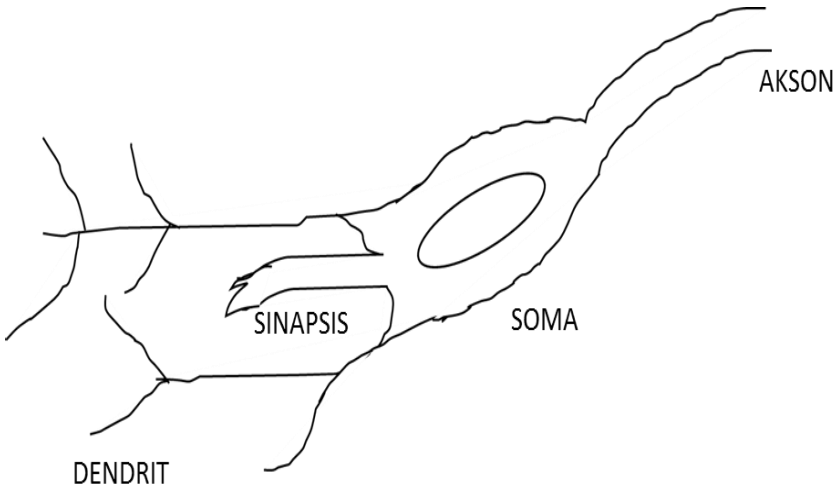
Yapay Sinir Ağları 1943’ te McCulloch ve Pitts in insan beyninin çalışmasını modellemesiyle başlar. Bu modelde ağırlıklar sabit oldukları için örneklerden öğrenememekteydi. 1949’ da Hebb in önerdiği bağlantı ağırlıklarının ayarlanması için bir öğrenme şeması önerdi ve bu da yapay sinir ağlarında öğrenme kuralının temeli oldu. Rosenblatt(1958) perceptron öğrenme kuralıyla ağırlık ayarlamalı perceptron modeli geliştirdi. Widrows ve Hoff ise 1960 yılında ağırlıklarının LMS(Least Mean Square) öğrenme algoritması kullanan ADALINE (Adaptive Linear Element) modeli önerdi. 1982’ de Hopfield tarafından Yapay Sinir Ağlarında geri beslemeye ağırlık vermiştir. Rumelhart ve arkadaşları 1986 yılında, giriş-çıkış örüntü çiftlerinde örtük haritalamayla çok katlı ileri beslemeli yapay sinir ağlarının ağırlık ayarlamasının mümkün olduğunu gösterdi ve bu

kurala da genelleştirilmiş delta kuralı ya da hata geri yayılımı (back propogation) adı verilmiştir. Sonraki yıllarda da hızlı bir şekilde gelişimi devam etmiş [30-33].

1.3.1. Yapay Sinir Hücresi

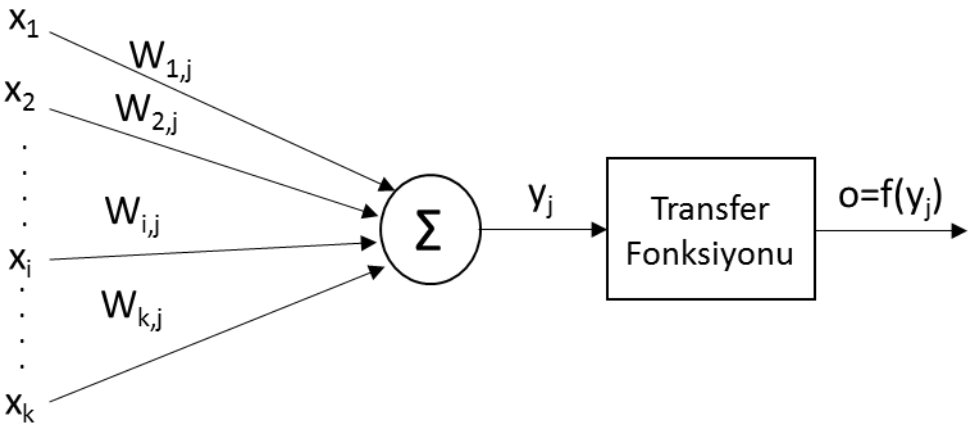
Yapay Sinir Ağları beynin biyolojik modelinden oluşturulmuştur. Bir Yapay Sinir Ağı, giriş hücrelerinden ya da başka sinir hücrelerinden bilgi alan birbirlerine bağlı bir dizi sinir hücrelerinden oluşur. Aldıkları bu girişe bir çeşit dönüşüm uygulayarak çıkışa yani diğer sinir hücrelerine ya da çıkış hücrelerine aktarırlar. YSA, katmanlardan oluşmuştur ve herbiri birbirine bağlı şekilde giriş katmanı bilgiyi alarak sonraki ilgili katmanlarla en sonunda çıkış katmanına ulaştırır.

Şekil 9' da görülen bir biyolojik sinir hücresinde dendritler, diğer sinir hücrelerinden alınan sinyalleri sinir hücresinin çekirdeğine iletirler; soma, gelen sinyallerin toplandığı sinir hücresinin merkezidir; akson, hücre çekirdeğinde toplanan bilgiyi sinapsis yoluyla diğer sinir hücrelerine aktarır; sinapsis ise aksondan aldığı toplam bilgiyi işlemden geçirdikten sonra diğer sinir hücrelerinin dendritlerine iletir.



Şekil 9. Biyolojik sinir hücresi

Biyolojik sinir hücresine benzer olarak yapay sinir hücresinde de Şeki 10' daki gibi gelen bilgilerin her biri bir ağırlıkla çarpılarak ağırlaştırılıp toplanır ve matematiksel bir fonksiyona tabi tutularak çıkışa aktarılır.



Şekil 10. Sinir hücresi yapısı

Bunu matematiksel eşitliklerle aşağıdaki gibi gösterebiliriz:
Öncelikle girişler ağırlaştırılıp toplanır.

$$y_j = b_j + \sum_{i=1}^{k} w_{ij} x_i \tag{24}$$

Sonra da buna eşik(bias) eklenerek matematiksel bir fonksiyondan geçirilir.

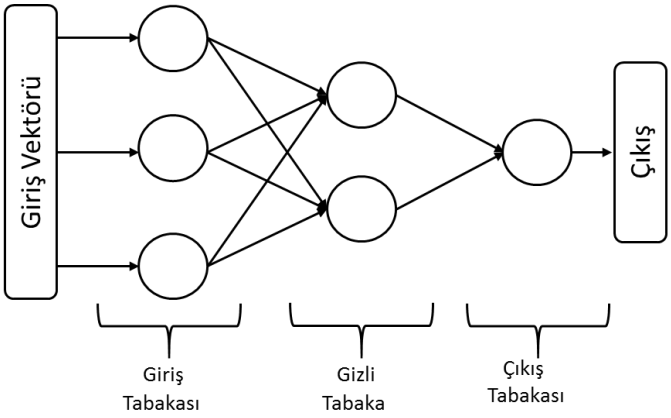
$$f\left(y_j\right) = f\left(b_j + \sum_{i=1}^{k} w_{ij} x_i\right) \tag{25}$$

1.3.2. Yapay Sinir Ağı

Yapay Sinir Ağı pek çok yapay sinir hücresinin birbirlerine bağlanması, iletişim halinde olup işbirliği içinde çalışmasıyla oluşur. Bu hücrelerin birbirlerine bağlanması çok farklı şekillerde olabilir. Ama bunlar genelleştirilip iki farklı şekle ayrılabilirler:

1.3.2.1. İleri Beslemeli Yapay Sinir Ağları

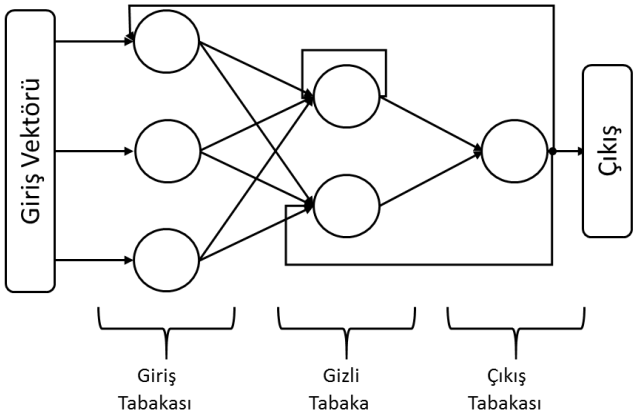
İleri Beslemeli Yapay Sinir Ağlarında, bilgi giriş katmanı, gizli katmanları ve çıkış katı üzerinden bağlantı yolu boyunca akar. Yani herhangi bir katın çıkışının kendini ya da önceki bir katı etkilemez. Grafiksel olarak Şekil 11' deki gibidir.



Şekil 11. Çok katlı ileri beslemeli yapay sinir ağı

1.3.2.2. Geri Beslemeli Yapay Sinir Ağları

Bu Yapay Sinir Ağı çeşidinde en az bir tane geri besleme bulunması gerekir. Bu nedenle, bu tür ağlarda bir kattan diğerine geri besleme alınabileceği gibi kendi çıkışından girişine geri besleme alan bir yapay sinir hücresi de bulunabilir. Grafiksel olarak Şekil 12' deki gibi gösterilebilir.



Şekil 12. Çok katlı ileri beslemeli yapay sinir ağı

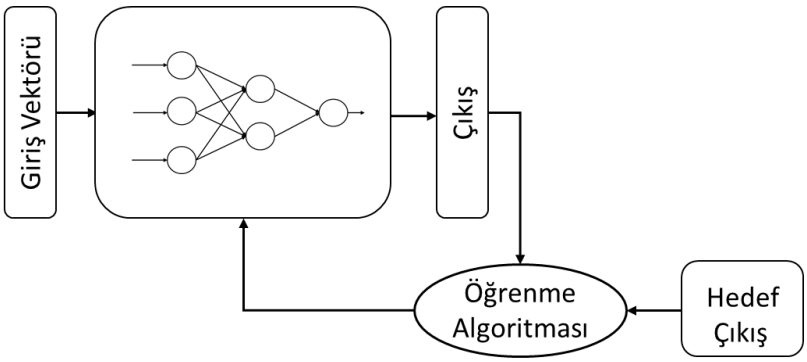
1.3.3. Yapay Sinir Ağlarında Öğrenme

Yapay Sinir Ağlarınının en kullanışlı özelliği çevreden öğrenebiliyor olması ve bu öğrenme yoluyla performansını iyileştirebiliyor olmasıdır. Öğrenme sürecinde ağırlıklar

ayarlanır. Yapay Sinir Ağlarında, denetimli öğrenme (supervised learning), denetimsiz öğrneme (unsupervised learning) ve takviyeli öğrenme(reinforced learning) olmak üzere üç ana öğrenme çeşidi vardır.

1.3.3.1. Denetimli Öğrenme

Bu tür öğrenme çeşidinde, ağı eğitmek için kullanılan her bir girişin, istenen çıkışla yani hedef çıkışıyla alakası olmalıdır. İstenen çıkışla gerçek çıkış arasındaki fark alınarak, istenen çıkışı elde edebilmek için Yapay Sinir Ağının parametrelerinde gerekli ayarlamalar yapılır. Bunu grafiksel olarak yapısı Şekil 13'de görüldüğü gibidir.



Şekil 13. Denetimli öğrenme

Öğrenme Algoritması sistemden alınan çıkış ile istenen çıkış arasında kıyaslama yaparak gerekli ağırlıkları ayarlar. Ağırlık ayarlaması aşağıdaki gibi yapılabilir.

$i.$işlem birimi için $(t+1)$anındaki yeni ağırlık vektörü, (t)anındaki ağırlık vektörüne bağlı olarak denklem (26) 'daki gibi tanımlanabilir.

$$w_i(\mathrm{t}+1) = w_i(\mathrm{t}) + \Delta w_i(\mathrm{t}) \tag{26}$$

Bunu ayrık zamanda tanımlarsak $w_i^{\mathrm{k}+1} = w_i^{\mathrm{k}} + \Delta w_i^{\mathrm{k}}$ olacaktır

Ağırlık değişimi $\Delta w_i(\mathrm{t})$, istenen çıkışla gerçek çıkış arasındaki farkla orantılıdır.

$$\Delta w_i^k = -\eta\left(\frac{\partial E}{\partial w_i}\right) \tag{27}$$

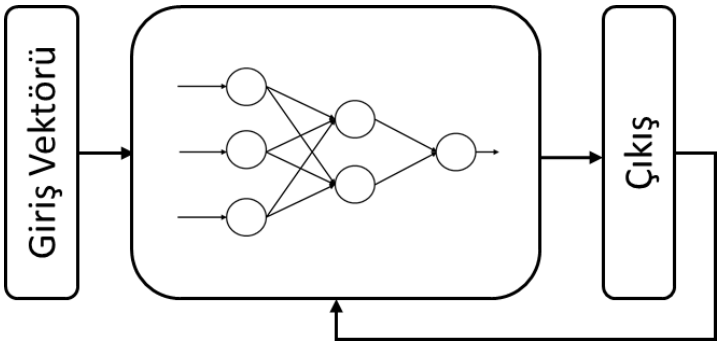
Gerçek çıkış $y_j = \sum_{i=1}^{n+1} f\left(w_{ji}x_i\right)$ ve hata $E = \sum_{t=1}^{T}\left(d_t - y_t\right)$ ve olduğundan ağırlık değişimi

$\Delta w_i^k = 2\eta\left(d_t - y_t\right)\frac{\partial y}{\partial u_t}x_{i,t}$ olacaktır.

Burada u_t net giriştir.

1.3.3.2. Denetimsiz Öğrenme

Denetimsiz öğrenmede, alınan çıkışın istenen çıkış olup olmadığını kontrol edecek bir geri besleme bulunmamaktadır. Şekil 14' de gösterildiği gibi bir denetleme olmadan sistem kendi düzenlemelerini, kendi özelliklerini, bağlantılarını ve sınıfını giriş bilgisinden otomatik olarak bulmalıdır. Yani denetim olmadan girişten kendi ayarını yapar.

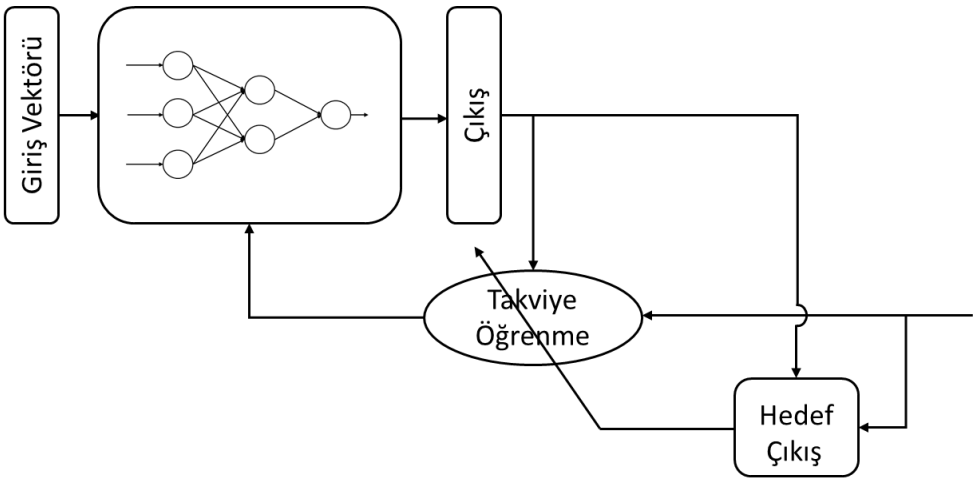


Şekil 14. Denetimsiz öğrenme

1.3.3.3. Takviyeli Öğrenme

Denetimli öğrenmede her bir giriş için bir hedef çıkış vardır. Ama pekçok durumda ayrıntılı bilgi edinilememektedir. Hatta bazı çok karmasık durumlarda çok uzun aralıkla giriş bilgisi verildikten sonra bilginin doğru olup olmadığını gösteren çok az bilgi alınabilmektedir. Takviyeli(Reinforcement) öğrenme böyle durumlar için idealdir çünkü

takviyeli öğrenmede geri besleme çevreden gelmektedir. Ama bu geri besleme öğretici değil sadece değerlendiricidir. Takviyeli Öğrenme Şekil 15' de görülmektedir.



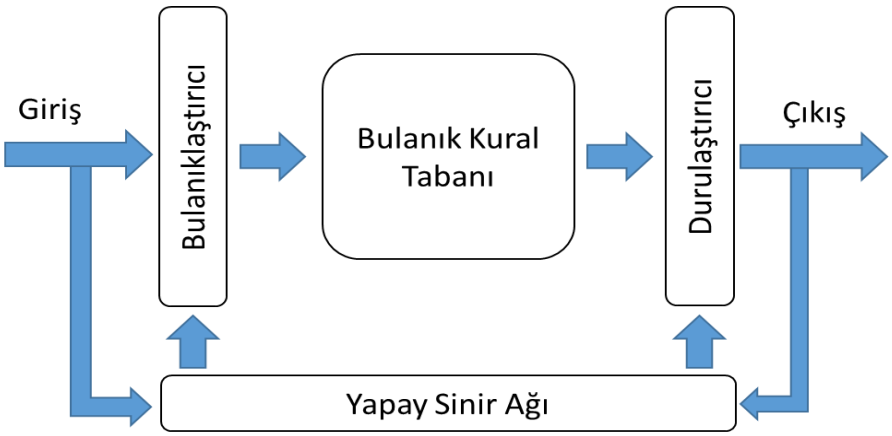
Şekil 15. Takviyeli öğrenme

1.4. Neuro-Fuzzy Sistemi

İki akıllı denetleyici Artificial Neural Network (Yapay Sinir Ağları) ve Fuzzy Logic (Bulanık Mantık) in beraber kullanılması her iki denetleyicide olan bazı sorunları gidermek için iyi bir yol olmuştur. Bulanık Mantık Denetleyiciler, matematiksel modele çok fazla bağımlı olmaması, parametrelerde olan değişikliklerden çabuk etkilenmemesi gibi avantajlarının yanında Bulanık Mantık Kontrolörde kullanılan üyelik fonksiyonlarının şeklinin belirlenmesinde ve dolayısıyla aralıklarının belirlenmesinde, karar tablosunun belirlenmesinde zorluklar oluşturabilmektedir. Yapay Sinir Ağları da bazı durumlarda tek başına yeterli olamayabilmektedir. Bu nedenle ikisinin birleştirilmesiyle pek çok probleme çözüm oluşturulmuştur [34-37].

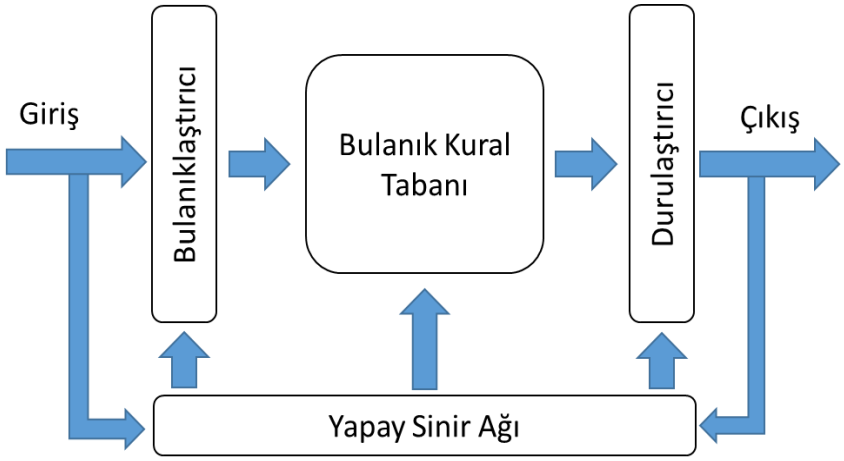
Neuro-Fuzzy Sinir Hücresi 1974 yılnda S. C. Lee ve E. T. Lee tarafından tanıtıldı[38]. Neuro-Fuzzy ile ilgili ilk çalışmalar ise 1990 yılında Lea, Jani ve Berenji ile başladı[39]. 1992 yılında Pal ve Mitra Bulanık Mantık giriş ve çıkışları olan bir Yapay Sinir Ağına uyarladı[40]. Altrock ve Krause 1993 yılında neuro-fuzzy teknolofisini otomotiv sektöründe kullandı[41]. Bu şekilde Yapay Sinir Ağları ve Bulanık Mantığın birleştirilmesiyle oluşan Neuro-Fuzzy günümüze kadar pek çok alanda kullanılarak yaygınlaştı.

Neuro ve Fuzzy pek çok farklı şekillerde ve pek çok farklı amaç için birleştirilmiş şekilde kullanılabilir. Şekil 16' da görüldüğü gibi Yapay Sinir Ağları kullanılarak Bulanık Mantığın Üyelik Fonksiyonları ayarlanabilir.



Şekil 16. Üyelik foksiyonları yapay sinir ağlarıyla ayarlanan bulanık mantık kontrolör şeması

Geliştirilen sistemlerle giriş-çıkış verisi lineer olmayan üyelik fonksiyonlarına ve kural tabanına dönüştürülmüştür. Böylece üyelik fonksiyonlarının yanında Şekil 17'de gösterildiği gibi Kural Tabanı da Yapay Sinir Ağlarıyla iyileştirilebilmiş ve kural tablosunun azaltılması sağlanmıştır.



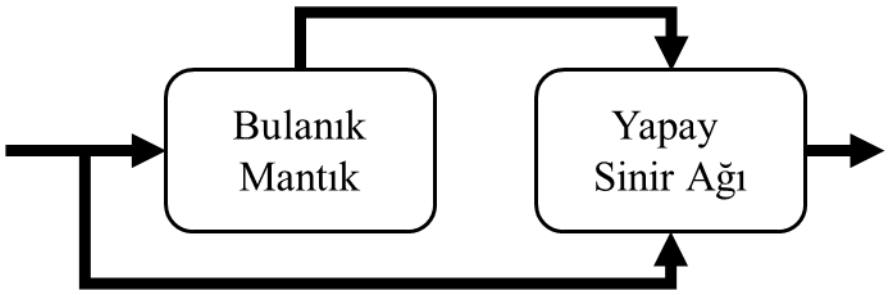
Şekil 17. Üyelik foksiyonlar ve kural tabanı yapay sinir ağlarıyla ayarlanan bulanık mantık kontrolör şeması

1.4.1. Neuro Fuzzy Çeşitleri

Yapay Sinir Ağlarının ve Bulanık Mantığın beraber kullanıldığı bütün sistemlere Neuro-Fuzzy denilebilir. Bu iki akıllı sitemin bir arada kullanışlı olması çok sayıda farklı Neuro-Fuzzy yapısının ortaya çıkmasına sebep olmuştur. Bunları şu şekilde gruplayabiliriz [42]:

1.4.1.1. Yardımlaşmalı Neuro-Fuzzy Sistemler

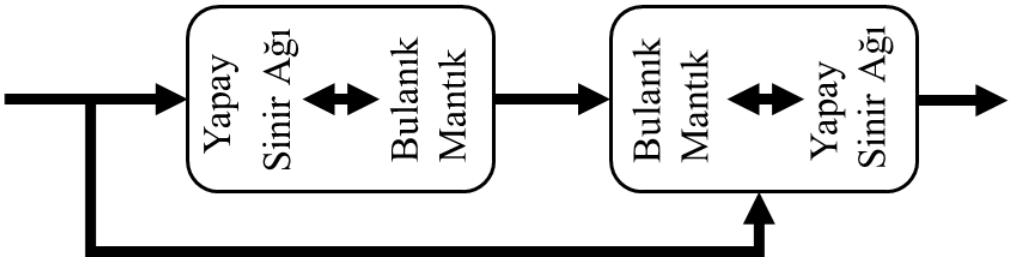
Bu tür sistemlerde Yapay Sinir Ağları sadece üyelik fonksiyonlarının aralıkları, şekilleri gibi Bulanık Mantığın alt bloklarının belirlendiği başlangıç kısmında kullanılır. Eğitim verisi(training data) ile Bulanık Mantık üyelik fonksiyonları ya da kurallar belirlendikten sonra Yapay Sinir Ağları Sistemden çıkarılır ve sadece Yapay Sinir Ağlarıyla ayarlanmış olan bu Bulanık Mantık sistemde kullanılır. Bu durum grafiksel olarak Şekil 18' de ifade edilmiştir.



Şekil 18. Yardımlaşmalı Neuro-Fuzzy sistem

1.4.1.2. Eşzamanlı Neuro-Fuzzy Sistemler

Eş zamanlı Neuro-Fuzzy sistemler Yapay Sinir Ağının ve Bulanık Mantığın sistemin başlangıcından çıkışına kadar sürekli beraber çalıştığı sistemlerdir. Eş zamanlı sistemler Şekil 19' da şematik olarak ifade edilmiştir.



Şekil 19. Eşzamanlı Neuro-Fuzzy sistem

Bu tür sistemde Bulanık Mantığa kullanılacak olan girişler önceden işlenir ve daha sonra Yapay Sinir Ağı eş zamanlı sistemin çıkışını işler ya da bunun tam tersi şekilde de olabilir.

1.4.1.3. Hibrit Neuro-Fuzzy Sistemler

Bu tür sistemlerde Yapay Sinir Ağları, bulanık kümeler, bulanık kurallar, kural ağırlıkları gibi Bulanık Mantığın bazı parametrelerini öğrenmesi için kullanılır. Neuro-Fuzzy dendiğinde, genellikle Neuro-Fuzzynin sadece bu çeşidini kastedilmektedir.

Bulanık Mantık ve Yapay Sinir Ağları hibdit olarak çok farklı şekilde birleştirilebilirler bu nedenle de çok fazla çeşitleri vardır. Bunlardan bazıları Adaptive Network based Fuzzy Inference System (ANFIS) R. R. Jang [43], Fuzzy Adaptive Learning Control Network (FALCON) C. T. Lin and C. S. Lee [44], Generalized Approximate Reasoning based Intelligence Control (GARIC) H. Berenji [45], Fuzzy Inference and Neural Network in Fuzzy Inference Software (FINEST) Tano, Oyama and Arnould [46] dir.

1.5. Genetik Algoritma

Yapay Sinir Ağları gibi Genetik Algoritma da doğadan esinlenilerek oluşturulan hesaplama yöntemlerinden yöntemlerden biridir. Doğadaki evrim yasalarına göre oluşturulan Genetik Algoritmada, en çok da doğal seçim yani ‘Güçlü olan hayatta kalır’ prensibi rol oynar. Doğada, hayatta kalabilmek için gerekli özelliklere sahip olan bireyler daha uzun sure yaşarlar. Böylece güçlü olan bireyler eşleşmiş olur. Bu nedenle de zamanla oluşacak bütün nüfus hayatta kalmak için güçlü özellikleri olan bireylerden meydana

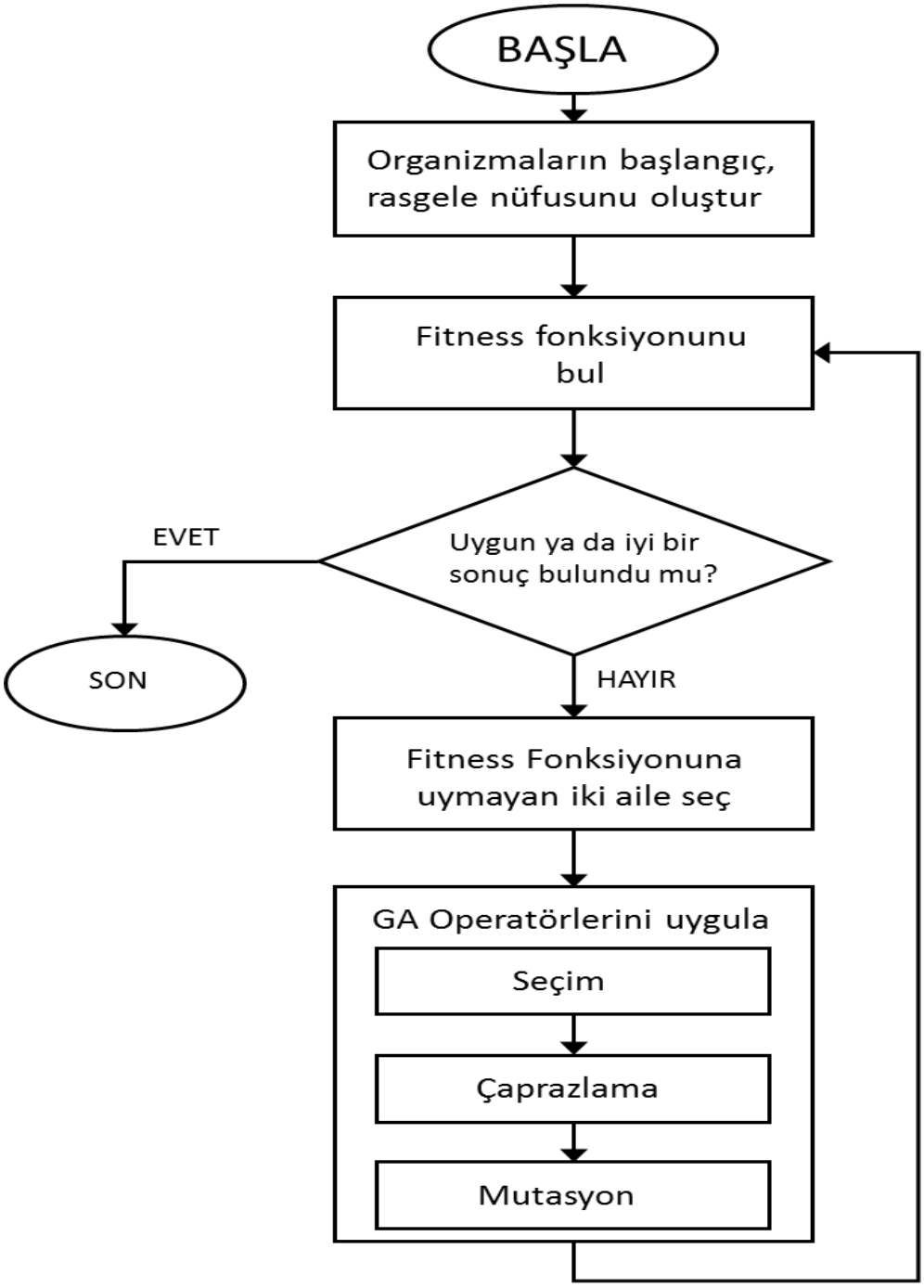
gelecektir. Bu yönüyle de optimizasyon yöntemi olarak iyi sonuçlar vermiştir ve geniş oranda kullanılmaktadır [47-49].

Genetik Algoritmanın geçmişi, Darwin'in ortaya attığı evrim teorilerine dayanır ancak bunun teknolojiye uyarlanacak şekli 1960 yılında Michigan Üniversitesinden Holland [50] tarafından ortaya atılmıştır. Sonraki yıllarda ise Goldberg [51,52] üstünde çalıştı ve uygulamalarında kullandı. Forest de daha fazla örnekler geliştirdi [53].

Hesaplama yönünden basit bir yapısı olan Genetik Algoritma, karmaşık uzaylarda bile sağlam bir sonuç verir. Diğer araştırma yöntemlerinden farklı olarak, tek bir nokta bulmak yerine, noktaların nüfusu için arama yapar ve parametre değerlerini kendileri bulması yerine kod kullanarak parametre ayarı yapar. Gradyan bilgi yani türev ya da yardımcı fonksiyon değil nesnel fonksiyon bilgisi kullanır aynı zamanda olasılıksal kurallarla değil kesin kurallarla işlev yapar.

1.5.1. Genetik Algoritma Temel Yapısı

Genetik Algoritmanın temel yapısını Şekil 20' deki diyagramla ifade edilebilir. Genetik Algoritmayı oluştururken ilk yapılması gereken, problemi tanımlayacak kodalamanın oluşturulmasıdır yani yaygın bir yöntem olarak binary veya string koda dönüştürülmesidir. Bu nedenle ilk adımda, bireyleri uygulanabilir çözümler üretebilecek nüfus oluşturulur. Bu da başlangıç çözümünü oluşturur. Diğer adımda, bu nüfusun bireyleri objektif fonksiyon değerini bulabilmek için değerlendirilir. Sonraki adım objektif fonksiyonunu, nüfustaki herbir üyenin uygunluk değerini hesaplayacak olan bir fitness fonksiyonuna dönüştürecek olan haritalamayı yapar. Dördüncü adım ise Genetik Algoritma operatörlerini uyular.



Şekil 20. Genetik algoritma akış diyagramı

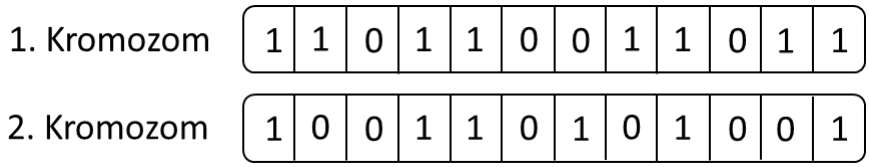
Bu akış diyagramının adımları daha ayrıntılı açıklanması Genetik algoritmanın anlaşılmasını kolaylaştıracaktır [53,54].

1.5.1.1. Kodlama

Genetik Algoritmanın uygulamasında öncelikle verilen problemin türüne göre çözümün kodlanması gerekir. Böylelikle sorun Genetik algoritmanın anlayabileceği şekle dönüştürülmüş olur. Kodlama farklı yöntemlerle yapılabilir.

1.5.1.1.1. İkilik Sistemde Kodlama

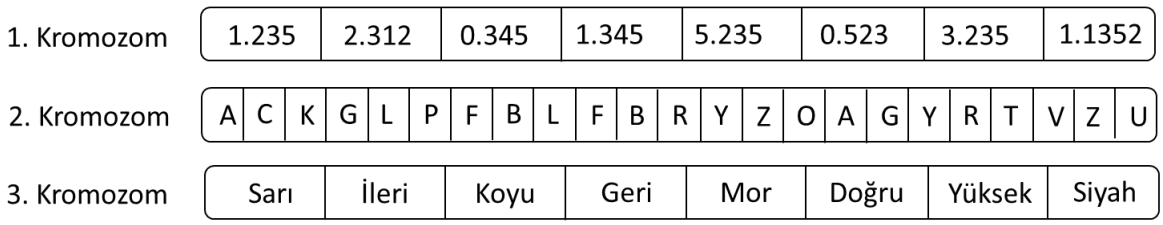
İkilik sistemde kodlama en yaygın olan kodlama şeklinden biridir. Genetik Algoritmada da kolay uygulnamasından dolayı çok tercih edilir. Bu kodlama çeşidinde Şekil 21' deki gibi kromozomlar ikilik sistemden yani 1 ve 0 lardan oluşur.



Şekil 21. İkilik sistemde kodlama örneği

1.5.1.1.2. Değer Kodlaması

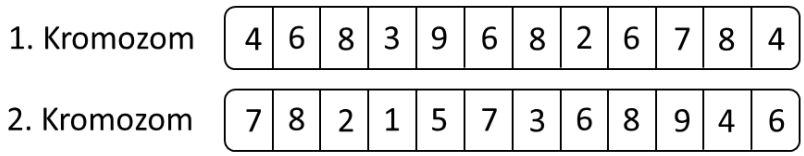
Bu kodlama real sayılar gibi değerlerle çalışıladığıda kullanılır. Bu tür problemlerin çözümünde ikilik sistemde kodlamanın kullanımı zor olacaktır. Bu tür uygulamalarda her bir kromozom bir değer dizisidir. Bu değerler Şekil 22' de görüldüğü gibi gerçel sayı, nesne veya harf olabilir.



Şekil 22. Değer kodlaması örnekleri

1.5.1.1.3. Permütasyon Kodlama

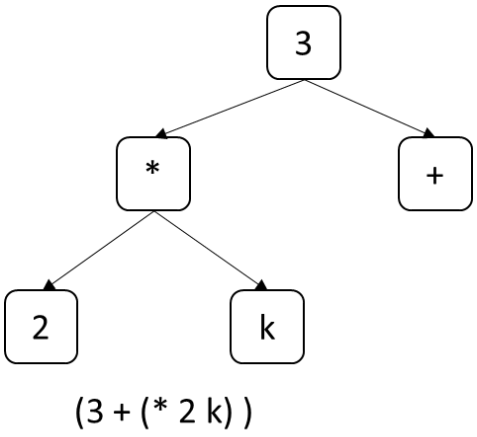
Sıralamada sorunlarının giderilmesinde kullanılan bir kodlama yöntemidir. Permütasyon kodlamada Şekil 23' deki gibi her bir kromozom string sayılardan oluşur ve bir konumu temsil eder.



Şekil 23. Permütasyon kodlama örneği

1.5.1.1.4. Ağaç Kodlama

Ağaç kodlama gelişen programlarla kullanılmaya başlamıştır. Ağaç kodlamada Şekil 24' deki gibi her bir kromozom bir uygulamanın ağacıdır. Yani bu, bir programlama dilindeki fonksiyonlar ya da emirler olabilir.



Şekil 24. Ağaç kodlama örneği

1.5.1.2. Fitness Fonksiyonu

Biyolojide fitness (uygunluk), bireylerin üretkenliğini ölçen bir parametredir. Bu nedenle Genetik Algoritmasını uygularken nüfus belirlemesinden sonra bu nüfustaki

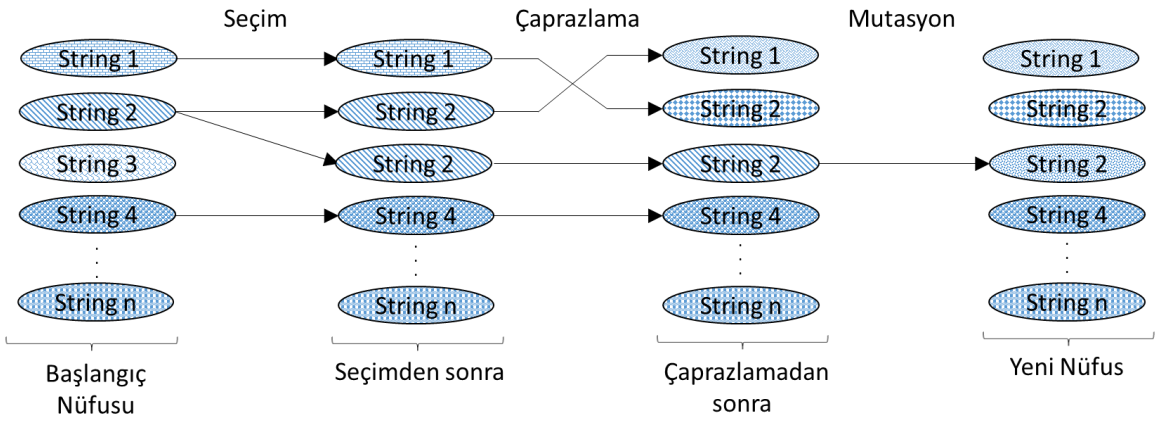
bireylerin istenen şartlara ne derece uygun oldu yani ne kadar iyi olduğunu belirleyebilmek için bir fitness (uygunluk) fonksiyonu belirlemek gerekir. Yani seçilen bireylerin ne kadar iyi olduğu anlamak için bir referans belirlenmiş olur. Bu fonksiyon maksimizasyon problemlerinde objektif fonksiyonla aynıyken yani $F(i) = O(i)$ alınırken minimizasyon problemlerinde aşağıdaki fonksiyon uygulanır.

$$F(x) = \frac{1}{1 + f(x)} \tag{28}$$

Burada $f(x)$ matematiksel bir fonksiyondur.

1.5.1.3. Genetik Algoritma Operatörleri

Genetik Algoritma operatörleri, tasarım verilerini ifade eden rasgele string değerlerinden oluşturulmuş olan bir nüfusla başlar. Bu nüfusa seçim, çaprazlama ve mutasyon adında üç farklı operatör uygulanır. Bu operatörler çeşitli çözümler oluşturarak fitness fonksiyonunu istenene en yakın şekle getirmeye çalışır. Bu operatörler Şekil 25‘ de gösterildiği gibidir.



Şekil 25. Genetik algoritma operatörlerinin genel yapısı

1.5.1.3.1. Seçim Operatörü

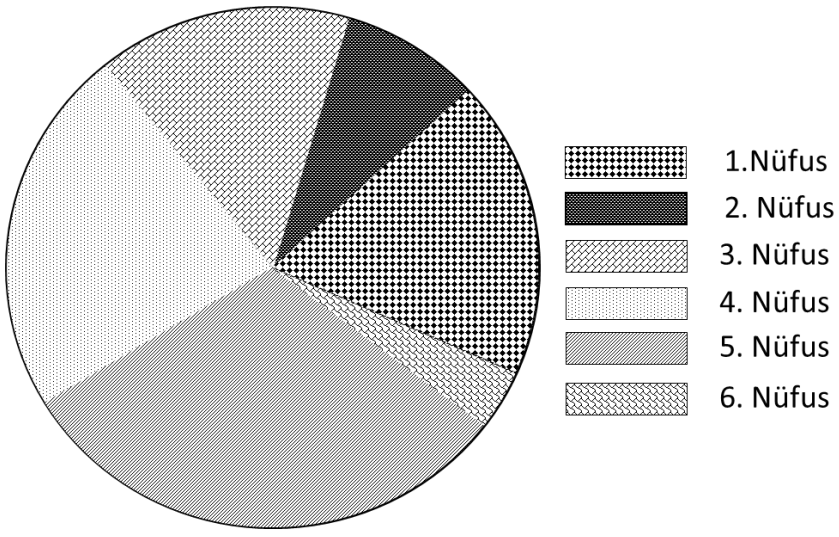
Genellikle nüfusa ilk olarak uygulanan seçim operatörü, yeni oluşturulacak nüfusun daha iyi string değerlerine sahip olabilmesi için uğraşır bu yüzden de nüfustaki en iyi string değerleri seçerek eşleştirme havuzunu oluşturur. Seçim operatörünün bu en iyi bireyleri seçip çoğaltabilmesi için farklı pekçok yöntem vardır ancak bunlardan en çok kullanılanları şunlardır:

1.5.1.3.1.1. Rulet Çarkı Seçim Yöntemi

Olasılıkla oranları ayarlanan Rulet Çarkıyla yapılan seçim operatörü en yaygın kullanılan operatörlerden biridir. Yani nüfustaki i.string, F_i olasılık oranlamasına göre seçilir. Burada n nüfus miktarı ve j çevrilme sayısı olmak üzere i. bireyin seçilme olasılığı olasılık fonksiyonu ise şöyledir:

$$p_i = \frac{F_i}{\sum_{j=1}^{n} F_j} \tag{29}$$

Şekil 26' daki gibi string değerlerinin uygunluğu için her bir string oranı işaretlenmiş bir Rulet çarkı n kez döndürülür ve her seferinde Rulet çarkının ibresiyle bir string seçilir.



Şekil 26. 6 nüfuslu bir rulet çarkı

Çarktaki bu oranlar stringin uygunluna göre çizildiği için bu çark sisteminin, eşleştirme havuzundaki i.string için $F_i / \overline{F}$ kadar çoğaltma yapması gerekir. Bu durumda nüfusun ortalama uygunluğu, (30)' daki gibi olur.

$$\overline{F} = \sum_{i=1}^{n} F_i \tag{30}$$

Buna göre de her bir nüfusun uygunluk değeri farklıdır. Bu çarka göre de 5. Nüfus diğerlerine göre daha yüksek olasılıklı uygunluğa sahiptir. Buların değerleri her bir stringin F_i leri kullanılarak p_i olasılıkları bulunabilir. Bu sayede de listenin başından her bir stringin olasılığını sırayla ekleyerek çoğaltılacak her bir stringin birikimli(kümülatif) olasılık dağılımı P_i bulunabilir. Bu şekilde de uygunluğu(fitness) en büyük olan stringin kümülatif olasılığı daha yüksek olacaktır ve eşleştirme havuzundan çoğaltılmak üzere seçilme olasılığı daha yüksek olacaktır. Aynı şekilde uygunluk değeri küçük olan stringin çoğaltılma olasılığı da küçük olacaktır.

1.5.1.3.1.2. Rasgele Kalan Seçim Yöntemi

Rasgele kalan seçim yönteminin temeli, üreme sayısına dayanan stringleri kaldırmak ya da çoğaltaktır. Buna da her bir stringle bağlantışı olan üreme sayısının hesaplanmasıyla ulaşılır.

Üreme sayısı, Rasgele kalan seçim yöntemine dayanan fitnessla hesaplanır. Öncelikle seçimin olasılığı denklem (31) olarak hesaplanır.

$$p_s = \frac{F_i}{\sum F_i} \tag{31}$$

Her bir stringin bireylerinin sayıları, N nüfus boyutu olmak üzere $e_i = p_s x P$ olacaktır. Her bir stringin e_i sinin ondalık kısmı onun olasılığı olarak kullanılarak nüfustaki bütün bireyler Bernoulli testine tabi tutulur. Nüfustaki bütün bireyler

incelemeden geçene kadar test devam eder. Sıfır üreme sayılı bireyler nüfustan atılır, diğer bireyler ise kendi üreme sayılarına sahip olan diğer nüfusa kopyalanırlar.

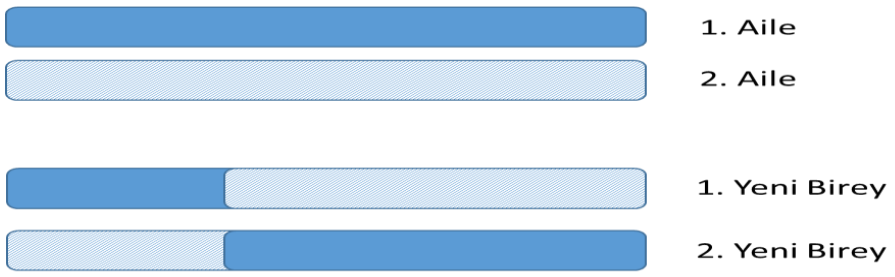
1.5.1.3.2. Çaprazlama

Çaprazlama operatörü, daha iyi bir string elde edebilmek amacıyla iki stringin yeniden birleştirilmesi için kullanılır. Yani önceki nesilden iki bireyin bileştirilmesiyle yeni neslin farklı bireyleri oluşturulur. Seçim aşamasında, nüfustaki iyi bireylerden olasılıksal olarak çoğaltılır ya da eşleştirme havuzu oluşturulur. Bu eşleştirme havuzundaki bireyler arasından da bilgiler değiştirilerek çaprazlama yapılır ve yeni string oluşturulur. Bu oluşan yeni stringler önceki stringlerden daha iyi de olabilir daha kötü de olabilir. Bu çaprazlamanın uygun kısımda yapılıp yapılmamasına göre değişir. Ancak iyi bir string oluşmaması çok da önemli değildir çünkü sonraki çaprazlamalarda oluşturulan eşleştirme havuzunda daha pekçok kopyalama yani çoğaltım işlemi yapılacaktır. Yani eşleştirme havuzundaki bütün stringler çaprazlamaya uğramaz çünkü zaten iyi olan stringlerin çaprazlanmasına gerek yoktur. Çaprazlamanın uygulanacağı stringlere aile; oluşan yeni stringlere de çocuklar denilebilir.

Literatürde farklı pekçok çaprazlama yöntemi bulunmaktadır. Temel üç tanesi şunlardır:

1.5.1.3.2.1. Tek Noktada Çaprazlama Yöntemi

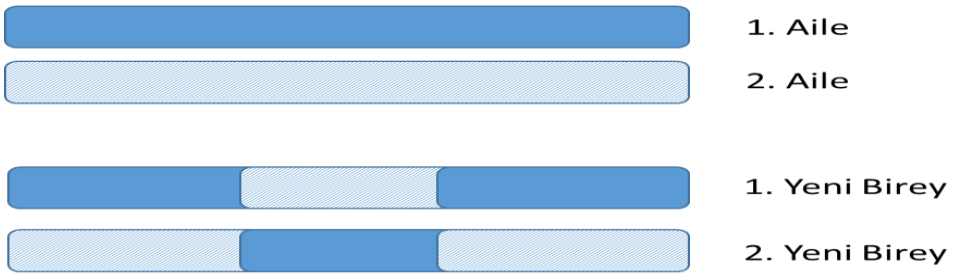
Tek Çaprazlama yönteminde önceki stringden rastgele seçilen bir noktadan sonraki herşey yeni stringe kopyalanır. Kısa stringli çaprazlamalar için daha uygun bir çaprazlama yöntemidir. Tek noktada çaprazlama yöntemi Şekil 27'de görülebilir.



Şekil 27. Tek noktada çaprazlama yöntemiyle yeni bireyleri oluşturulması

1.5.1.3.2.2. İki Noktada Çaprazlama Yöntemi

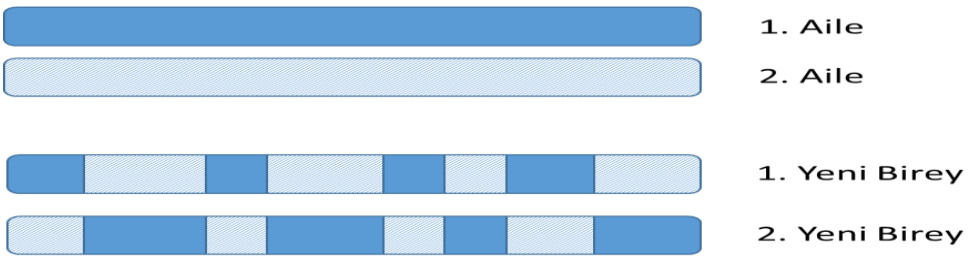
Bu çaprazlama yönteminde ise her iki ailede tek nokta yerine iki farklı nokta belirlenir ve bu noktalar arasında kalan kısım iki aile arasında çaprazlanır. Uzun stringli çaprazlamalar için tek noktada çaprazlamadan iki noktalı çaprazlama daha uygundur. Bu şekilde yeni bireyler oluşturulur. Şekil 28'de ne şekilde olduğu görülmektedir.



Şekil 28. Tek noktada çaprazlama yöntemiyle yeni bireyleri oluşturulması

1.5.1.3.2.3. Üniform Çaprazlama Yöntemi

Karıştırma oranı olarak bilinen bir olasılıksal yöntemle yeni bireyler için hangi aileden ne kadar genin alınacağı belirlenir. Şekil 29'da üniform çaprazlama görülmektedir.



Şekil 29. Üniform çaprazlama yöntemiyle yeni bireyleri oluşturulması

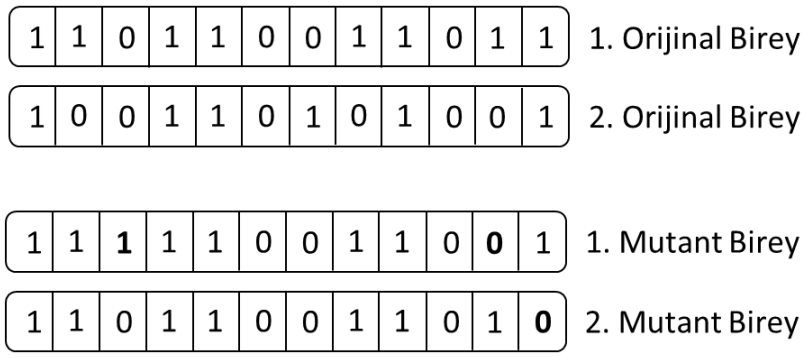
1.5.1.3.3. Mutasyon

Mutasyon, genetik arama sürecine rasgele bir şekilde yeni bilgiler ekler ve tek bir yerel optimada sıkışıp kalmaktan kurtarır. Nüfusa tekrar tekrar uygulanan seçim ve çaprazlama operatörlerinden dolayı zamanla nüfusun tekdüze olmaya başlaması durumunda mutasyon operatörü imdada yetişir ve nüfusta çeşitlilik oluşturur. Mutasyonla

yeni oluşacak bireylerin kromozom değerleri bir ya da iki gen farklı olabilir. Bu da eşleştirme havuzuna eklenecek tamamen farklı gen değerleri demektir. Böylece Genetik Algoritma, bu yeni gen değerleriyle öncekilerle oluşturabileceğinden daha iyi sonuçlar oluşturabilir. Bu anlamda mutasyon, Genetik Algoritmanın tek bir lokal optima etrafında dolaşıp durmasını engelleyerek daha iyi sonuçlar arayabilmesini sağlar. Mutasyon farklı bir çok şekilde Genetik Algoritmaya uygulanabilir. Bunlardan en yaygın olanları aşağıda açıklanmıştır.

1.5.1.3.3.1. Flip-Bit Mutasyon Yöntemi

Flip-Bit mutasyon yöntemi seçilen bir genin dönüştürülmesini sağlar. Yani birse sıfır, sıfırsa bir yapılır. Şekil 30 bu durumu ifade etmektedir. Ancak bu mutasyon yöntemi sadece binary genler için kullanılır.



Şekil 30. Flip-Bit yöntemiyle mutasyon

1.5.1.3.3.2. Üniform Mutasyon Yöntemi

Kullanıcı tarafından belirlenen alt ve üst sınırlar çerçevesinde, rasgele belirlenen üniform aralıklarla genlerin değiştirilmesiyle oluşturulan mutasyon yöntemidir. Bu mutasyon operatörü sadece integer ve float genlerle kullanılır.

1.5.1.3.3.3. Sınır Mutasyon Yöntemi

Bu mutasyon yönteminde değiştirilmesi gereken gen rasgele seçilerek alt ya da üst sınırlardan biriyle değiştirilir. Hangi sınırla değiştirileceği de rasgele belirlenir. Bu mutasyon yöntemi sa uniform mutasyon yöntemi gibi sadece integer ve float genlerle kullanılır.

1.5.1.3.3.4. Gauss Mutasyon Yöntemi

Gauss Mutasyon yönteminde, mutasyon operatörü değiştirilmesi için seçilen gene verilecek random sayıyı belirlemesi içi bir Gauss kullanır. Rechenberg and Schwefel taradından geliştirilen mormal mutasyon yöntemlerine Gauss eklenerek oluşturulur [44]. Bu yöntemde kullanılan Gauss olasılık yoğunluk fonksiyonu şöyledir:

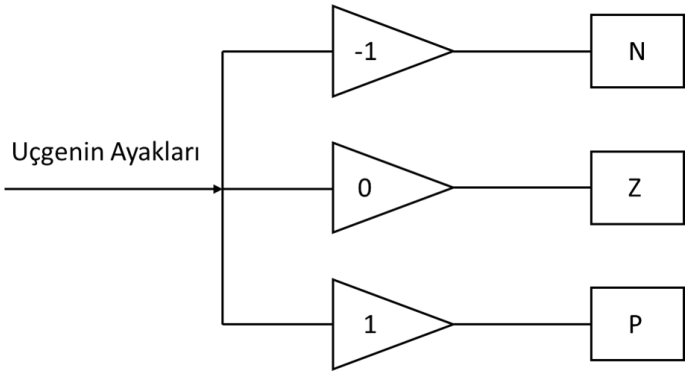
$$f_{gauss(0,\sigma^2)}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}} \tag{32}$$

2. YAPILAN ÇALIŞMALAR

2.1. Genetik Algoritma Optimizeli Yapay Sinir Ağlarıyla Ayarlanan Bulanık Mantık Denetleyici

Literatür taramasından da bahsedildiği gibi akıllı denetleyiciler, sistem değişikliklerindeki dayanıklılığı, istenen çıkışı takip yeteneğini, adaptif edilmesindeki kolaylığı, hızlı cevaplama süreleri ve hızlı işlem yapma süreleri gibi özelliklerinden dolayı geleneksel akıllı olmayan Oransal, Türev, İntegral ve bunların kombinasyonları olan denetleyicilerden daha çok tercih edilir duruma gelmişlerdir. Bu akıllı sistemlerin de bir arada kullanılmasıyla daha iyi sonuçlar elde edilebilmiştir.

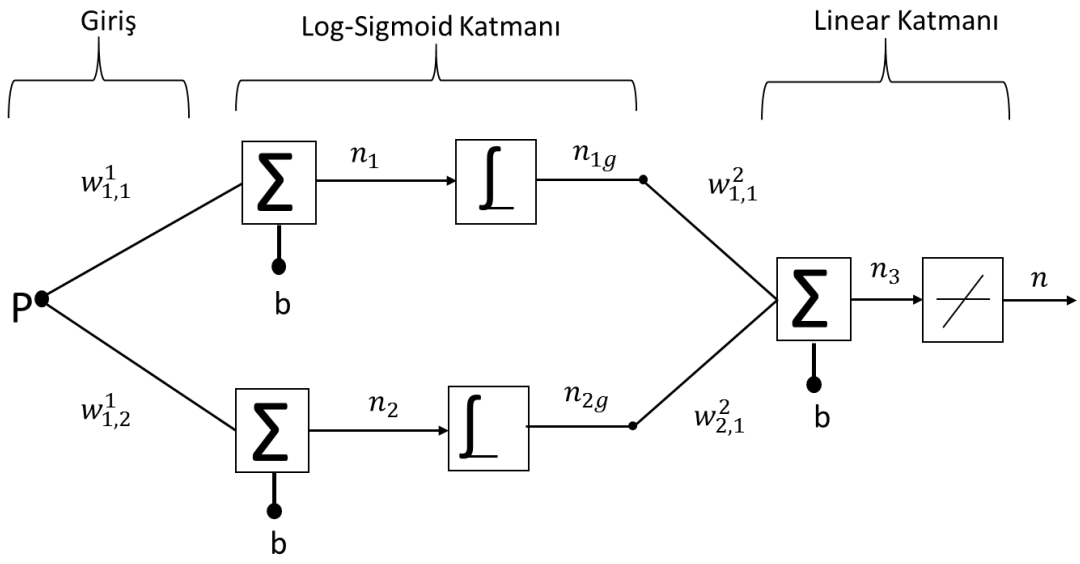
Genetik Algoritma Optimizeli Yapay Sinir Ağlarıyla Ayarlanan Bulanık Mantıkyönteminde Yapay Sinir Ağları ve Bulanık Mantık bir arada kullanılmıştır. Böylece Bulanık Mantığın olumlu özelliklerinin yanında olumsuz özelliği olan üyelik fonksiyonlarının aralıklarının belirlenmesindeki zorluk sorunu çözümlenmeye çalışılmıştır. Bu yöntemde kullanılan Bulanık Mantık denetleyicinin denetlenen üçgen ayakları Şekil 31'de [8] gösterilmiştir.



Şekil 31. Üçgen aralıkları

Bulanık Mantıktaki e(hata), de(hata değişimi) ve du(çıkış değişimi) üyelik fonksiyonlarının aralıkları optimizasyon yöntemi olarak Genetik Algoritma kullanan Yapay Sinir Ağlarıyla ayarlanmaktadır. Kullanılan Yapay Sinir Ağı Şekil 32' deki gibi ilk aynı girişi kullanan katında lineer öğrenme fonksiyonlu n nöron, ikinci katında ise sigmoid fonksiyonlu n girişli bir çıkışlı nöron kullanan bir ağdır. Buradaki n nöron sayısı sistemin

ihtiyacına göre değişir. Eğer basit bir sistemse az girişle de sistem denetlenebilir ancak karmaşık sistemler için işlem süresinin artması göze alınarak n sayısı arttırılabilir. Yapay Sinir Ağının ağırlıklarının ve biaslarının en uygun çıkış verecek şekilde ayarlanması gerekir. Bu da zor ve uzun süren bir iştir. Bu nedenle Genetik Algortima kullanılarak hem bu zorlu işten kurtarır hem de Yapay Sinir Ağını, Genetik Algortima ile optimize etmek, Genetik algoritmanın diğer optimize yöntemlerine göre girişteki değişikliklerde ve gürültüde kolayca bozulmayan bir çıkış elde edilmiş olur.



Şekil 32. Sistemde metodunda kullanılan yapay sinir ağı

Bu ağın denklemlemleri aşağıdaki şekilde yazılabilir.

$$n_1 = pw^1_{1,1} + b^1_{1,1} \tag{33}$$

$$n_2 = pw^1_{1,2} + b^1_{1,2} \tag{34}$$

Fonksiyon uygulandıktan sonraki çıkışlar:

$$n_{1g} = f_{\tan sig}(n^1_{1,1}) = f_{\tan sig}(\mathrm{p}\, w^1_{1,1} + b_1) \tag{35}$$

$$n_{2g} = f_{\tan sig}(n_{1,2}^{1}) = f_{\tan sig}(\mathrm{p}\, w_{1,1}^{1} + b_2) \tag{36}$$

Birinci katmanın çıkışları ikinci katmanın girişleri olduğuna göre n_3, denklem (37) ve (38)' deki gibi olur.

$$n_3 = n_{1g} w_{1,1}^{2} + n_{2g} w_{2,1}^{2} + b_3 \tag{37}$$

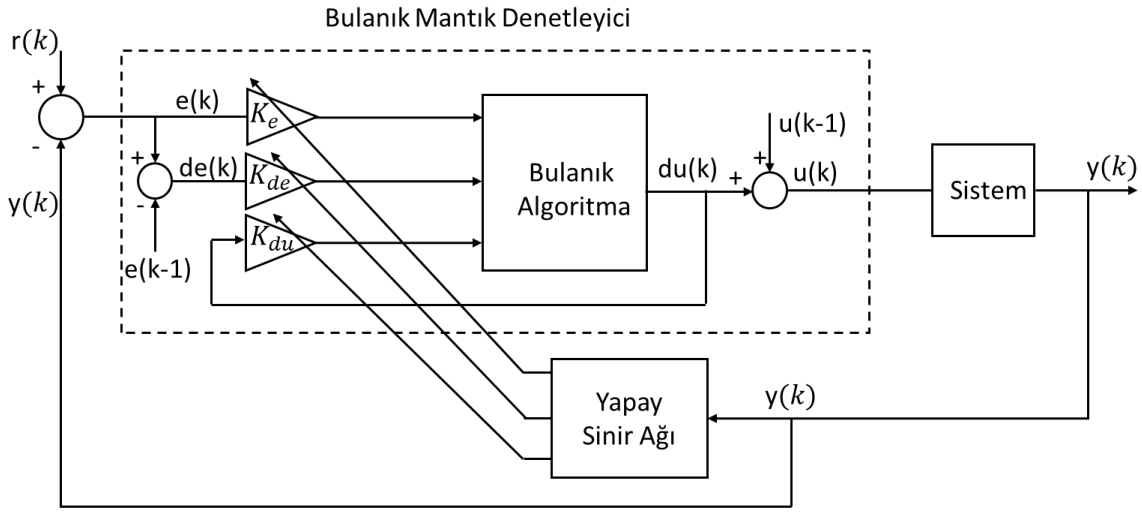
$$n_3 = f_{\tan sig}(\mathrm{p}\, w_{1,1}^{1} + b_1) w_{1,1}^{2} + f_{\tan sig}(\mathrm{p}\, w_{1,1}^{1} + b_2) + \mathrm{b}_3 \tag{38}$$

İkinci katmana fonksiyon uygulandıktan sonra sinir ağının çıkışı denklem (39), (40)' da elde edilmiştir.

$$y = f_{purelin}(y_3) \tag{39}$$

$$y = f_{purelin}(f_{\tan sig}(x w_{1,1}^{1} + b_1) w_{1,1}^{2} + f_{\tan sig}(x w_{1,1}^{1} + b_2) + \mathrm{b}_3) \tag{40}$$

Önerilen, Genetik Algoritma Optimizeli Yapay Sinir Ağlarıyla Ayarlanan Bulanık Mantık Denetleyici yönteminde sistem çıktısı öğretmen, Yapay Sinir Ağı da öğrenci gibidir. Öncelikle bir kere çalıştırılarak sırasıyla bilginin öğretilmesi(train data), kontrol edilmesi(check data) ve test(test data) edilmesi kısımları yapılır. İstenen veriler öğretildikten sonra, üçgen olarak belirlenen bulanık mantığın üyelik fonksiyonlarının aralıkları kontrol edilebilir. Bu öğretim aşaması için kurulması gereken sistemin genel yapısı Şekil 33' deki gibidir.

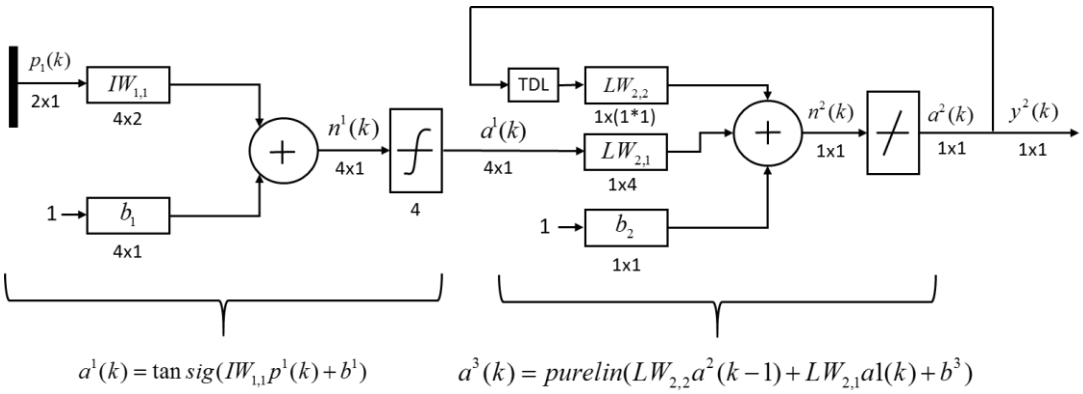


Şekil 33. Önerilen yöntemin öğrenme sisteminin genel yapısı

Değişken parametrelerimizin değerlerini rastagele seçildiği($K_e = 1$, $K_{de} = 0.1$, $K_{du} = 1$) önerilen yöntemde, öğretme aşaması için öncelikle giriş değerleri ($x_1,...,x_n$) , çıkış değerleri ($y_1,...,y_n$) ve başlangıç ağırlık değerleri ($w_1,...,w_n$) matrisel olarak tanımlanır. Çalışmadaki Yapay Sinir Ağı, giriş olarak gelen hatayı sıfırlayacak şekilde eğitilecektir. Hatayı sıfırlayabilmek için de Bulanık Mantık Denetleyicinin üyelik fonksiyon aralıklarını ayarlayacak ve hatayı sıfırlayan en iyi değerleri bulmaya çalışacaktır. Yani Bulanık Mantık Denetleyiciyi optimize edecektir. Aynı zamanda Yapay Sinir Ağı sistemin çıkışından geri besleme aldığı için ve istenen çıkış ile elde edilen çıkış arasındaki farkı alarak Genetik Algoritma yardımıyla ağırlıkları güncellediği için adaptif bir sistem oluşturacaktır.

Burada kullanılan yapay sinir ağlarının modellenmesi şu şekilde açıklanabilir:

1) Sistemin çıkışının Yapay Sinir Ağlarına öğretilebilmesi için öncelikle çıkışın örneklenmesi gerekir. Örnekleme zamanının, çıkış için yeterli seçilmesi gerekir. Aksi takdirde farklı referanslar için istenen çıkışı vermeyebilir ve dolayısıyla optimize işleminde yetersiz kalabilir.
2) Sonraki adımda Matlab'da 'netwok' komutuyla boş bir Yapay Sinir Ağına açılmıştır. Kullanılacak Yapay Sinir Ağı farklı şekillerde ve katlarda olabilir ancak bu çalışmada kullanılan Şekil 34' deki gibi tek giriş ve çıkışlı Yapay Sinir Ağıdır.



Şekil 34. Sistemde kullanılan yapay sinir ağlarının temel yapısı

Burada $p_1(\mathrm{k})$ giriş olarak tanımlanmıştır ve işlem süresinin kısalması için girişin hangi aralıklar arasında olması gerektiğini belirlemekte fayda vardır. Sistemde kullanılan giriş -5 ve 5 değerleri arasında alınmıştır.

Şekil 34'de başlangıçtaki ağırlıklar ve bayaslar 'Nguyen-Widrow(initnw)' elde edilmiştir. Matlap üzerinde öğrenme metodu trainlim olarak kulanılmıştır.

Nguyen-Widrow(initnw) yönteminin çalışma şekli aşağıdaki gibidir.

a- Ilk adımda bu metodu uygulamak ağırlıkların (-1,1) arasında rastegele sayılar alınıyor ve $\beta = 0.7h^{\frac{1}{i}}$ değeri hesaplanacaktır. Burada h gizli tabakanın neuron sayısı ve i giriş neuron sayısıdır.

b- Formül (41) 'deki Öklid normu her tabakada bütün ağırlıklar için hesaplanacaktır.

$$n = \sqrt{\sum_{i=0}^{i<w_{\max}} w_i^{\,2}} \tag{41}$$

Buna göre de yeni ağırlık değerleri 3)' deki matlab yazılımının 7.satır eşitliğiyle hesaplanacaktır. Aynı eşitlik eşiklerin ayarlanmasında da kullanılır. Bir kere ağırlıklar ve eşikler ayarlandığında Yapay Sinir Ağları kullanıma hazırdır.

3) Ağırlık değerleri elde edildikten sonra şebekeyi çıkış değerlerinin Yapay Sinir Ağlarına öğretilmesi ve istenen değerlerle kıyaslanması gerekir. Öğretme algoritması şu şekilde Matlabda elde edilmiştir:

$w^0 = 1$

for t=1...T

for l=1...N

$g_{k,l} = f_k(\mathrm{x}_l, \mathrm{y}_l) - \mathrm{E}_{P_{r(\mathrm{y}'|\mathrm{w})}} f_k(\mathrm{x}_l, \mathrm{y}')$ *k=1....K*

end

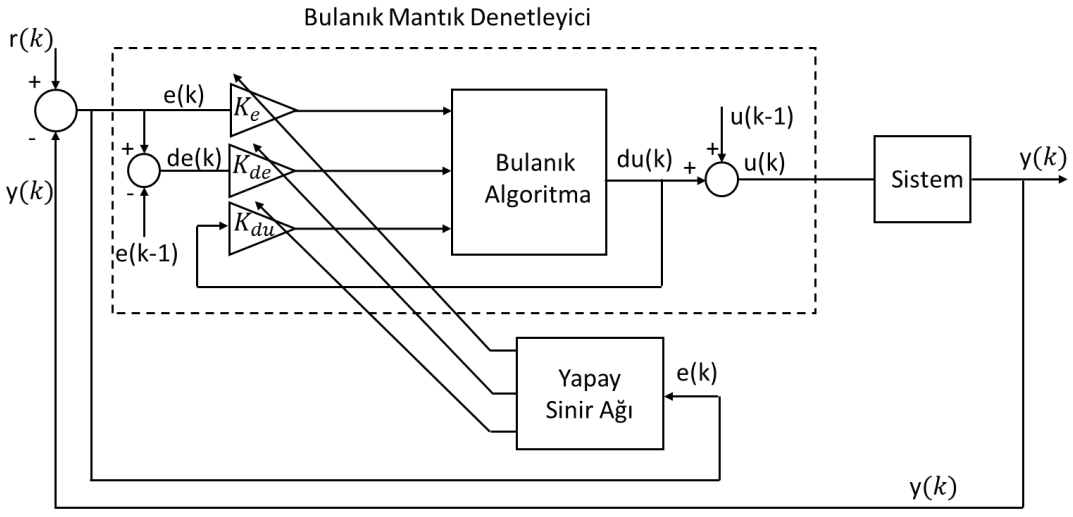
$g_k = \sum_l g_{k,l}$ *k=1....K*

$W^t{}_k = W^{t-1}{}_k + \beta_t (g_k - 2w^{t-1}{}_t / c)$

exit if $\|g\| \cong zero$

end for

Öğrenme tamamlandıktan sonra Genetik Algoritma Optimizeli Yapay Sinir Ağlarıyla Ayarlanan Bulanık Mantık Denetleyici, bir kontrolör olarak kullanıma uygun hale gelmiş olur. Bu denetleyici, referans girişinde meydana gelebilecek herhangi bir değişiklikte kendiliğinden öğrenir ve parametreleri ona göre ayarlar. Bu özelliği de kullanıcıyı her değişiklikte yeniden öğretme sorunundan kurtarır. Önerilen bu yöntemin temel yapısı Şekil 35' deki gibidir.



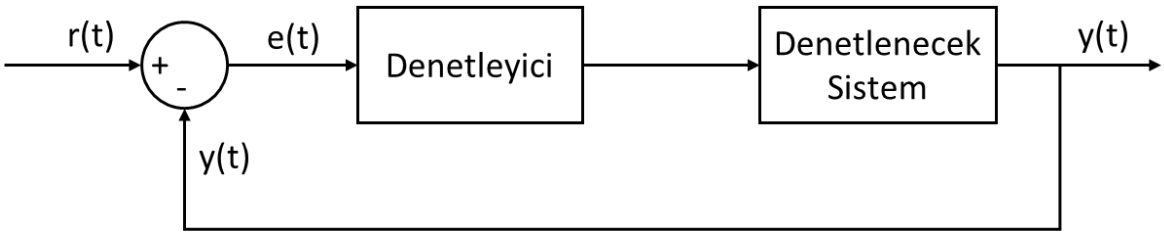
Şekil 35. Önerilen denetleyicinin genel yapısı

Bu sistemde ağırlıkların ayarlanması için Genetik Algoritma kullanılmıştır. Kullanılan Genetik Algoritmasının işleyiş şekli şöyledir:

1) İlk başlangıçta elde ettiğimiz hata değerini -5x4 ve 5x4 arasında alınır. Bunun sebebi ise hesaplama süresinin azaltılmasıdır.
2) Nüfus kısmında değişken sayısına göre gen alınması gerekir. Her gen bir bit olarak tanımlanmıştır. Buradaki değişkenler k_e , k_{de} , k_{du} ve W dir. Bu nedenle 4 gen kullanılmıştır.
3) Kromozomlara rastgele değer verilmiştir. Kromozom sayısı sistemin dikkatine bağlıdır.

2.2. DA Motorunun Önerilen Yöntemle Hız Kontrol Benzetim Uygulaması

Kontrol uygulamalarını günlük hayatta farkında olarak ya da olmayarak her alanda kullanılmaktadır. Kontrol işlemleri genellikle insan müdahalesi olmadan otomatik olarak yapılırlar. Kontrol uygulamasının nasıl yapıldığını görmek için Şekil 36' da görülen genel kontrol şemasını hatırlamakta fayda vardır.

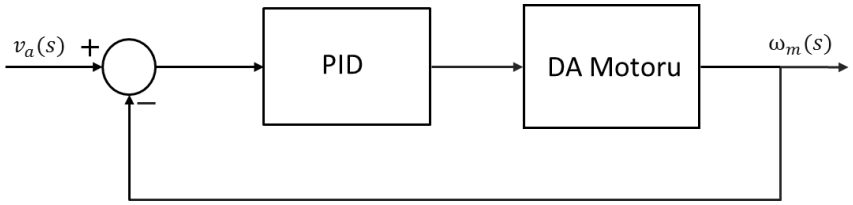


Şekil 36. DC motor genel kontrol şeması

Doğru Akım(DA) motoru, hız kontrolü kolay olduğu için kontrol uygulamalarında tercih edilmektedir. DA motor hız kontrolünde farklı yöntemler tercih edilebilir. Günümüze kadar farklı yöntemlerle uygulanmıştır. Bu çalışmada da DC motor, PID, Bulanık Mantık Denetleyici, Yapay Sinir Ağı, Neuro-Fuzzy yöntemleriyle ve bu yöntemlere kıyasla daha iyi sonuç vereceği düşünülen Genetik Algoritma Optimizeli Yapay Sinir Ağlarıyla Ayarlanan Bulanık Mantık Denetleiyici ile denetlenmiştir. Bu yöntemlerle denetlenen DC motorun hız çıktıları alınmıştır.

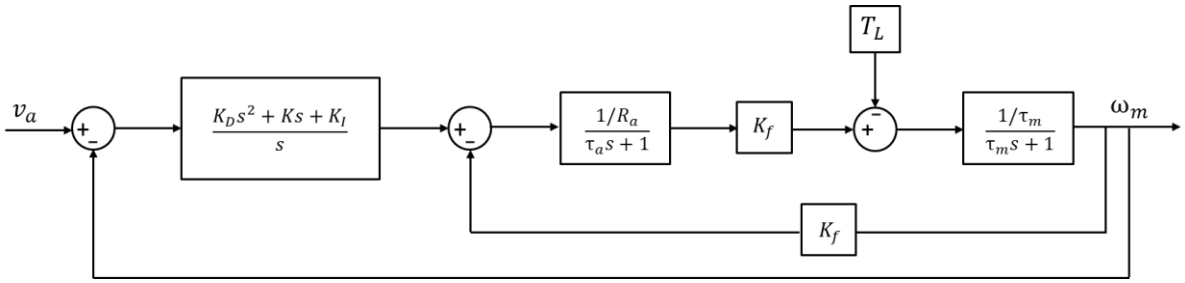
2.2.1. DA Motor PID Denetleyici ile Kontrolü

PID denetleyicide daha once bahsedildiği gibi oransal, türev ve integral katsayıları bulunan bir blok kullanılır. Şekil 37'de PID ile bir motorun kontrolünün gösterildiği kapalı bir çevrim görülmektedir.



Şekil 37. DA moturunun PID ile kontrolünün genel şeması

DA motorun Matlab/Simulink ortamında denetlenmesi Şekil 38'de görüldüğü gibidir.



Şekil 38. DA moturunun PID ile kontrolümüm benzetim modeli

Burada elektriksel kısımdaki zaman sabiti $\tau_a = \frac{L_a}{R_a}$ ve mekaniksel kısımdaki $\tau_m = \frac{J}{\mathrm{B}}$ dir.

PID denetleyicinin parametreleri ise

$K_P = 0.0036$,

$K_I = 0.1139$,

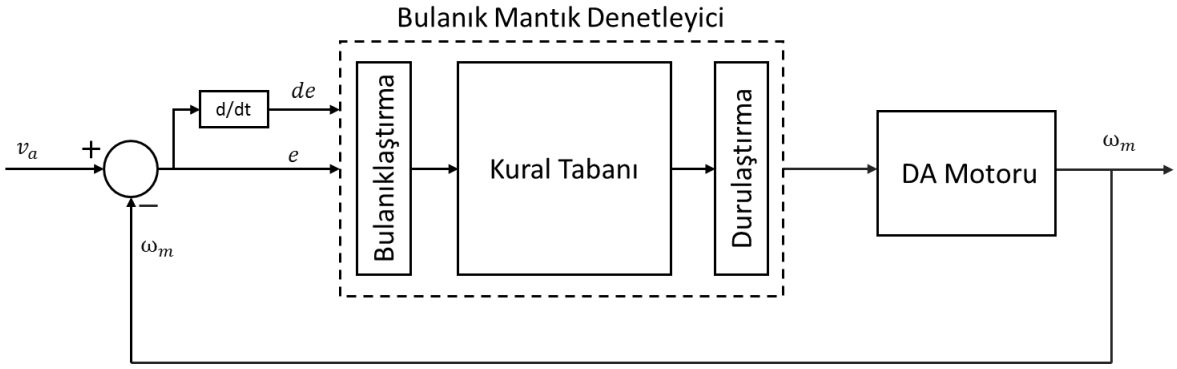
$K_D = \text{-}3.8573\text{e}^{\text{-}7}$ olarak alınmıştır.

Böylece oturma süresi yaklaşık 0.3 olmuştur. Ancak bir çok uygulama için istenmeyen bir durum olan aşma mevcuttur ve salınım fazladır. Denetleyiciden beklenen özelliklerden biri de referansta meydana gelebilecek ani değişikliklerden çok etkilenmemesidir.

2.2.2. DA Motorun Bulanık Mantık ile Denetlenmesi

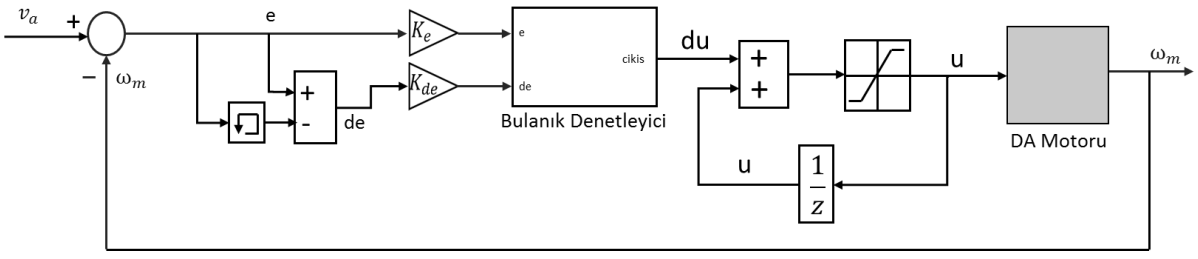
PID denetleyiciler doğrusal olmayan sistemlerin modellenmesinde ve kesin bir matematiksel model gerektiren bazı durumlarda yeterli olamamaktadır. Bunun için akıllı yöntemler karmaşık sistemlerde daha kullanışlı olabilmektedir. Akıllı bir sistem olan Bulanık Mantık da sistem parametre değişikliklerine ya da çevredeki bozucu etkilere karşı daha dayanıklı olabilmektedir.

DA motorunun Bulanık Mantık Denetleyici ile denetlenmesini gösteren genel kontrol diyagramı Şekil 39' da verilmiştir.



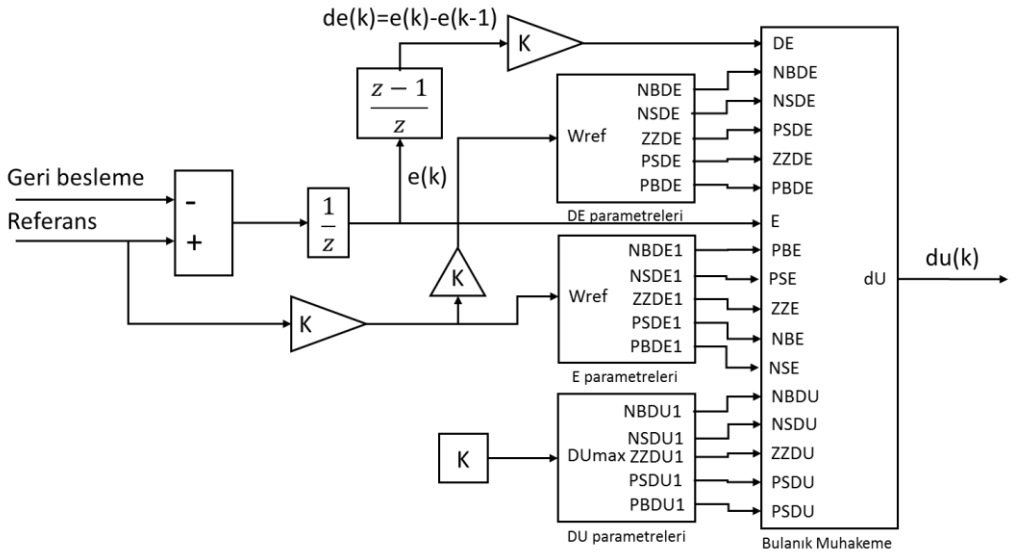
Şekil 39. DA Motorunun bulanık mantık ile kontrolünün genel şeması

Bu sistemin Matlab/Simulinkte oluşturulan şekli Şekil 40' ta gösterilmiştir.



Şekil 40. DA moturunun bulanık mantık ile kontrolünün benzetim modeli

Bu Bulanık Mantık modelinde kullanılan Bulanık Denetleyici bloku [8] Şekil 41’ de görüldüğü gibidir.



Şekil 41. Sistemde kullanılan bulanık mantık denetleyici bloku

Sistemde kullanılan kural tablosu ise Tablo 4’ deki gibidir.

Tablo 4. Sistemde kullanılan Bulanık Mantık kural tablosu

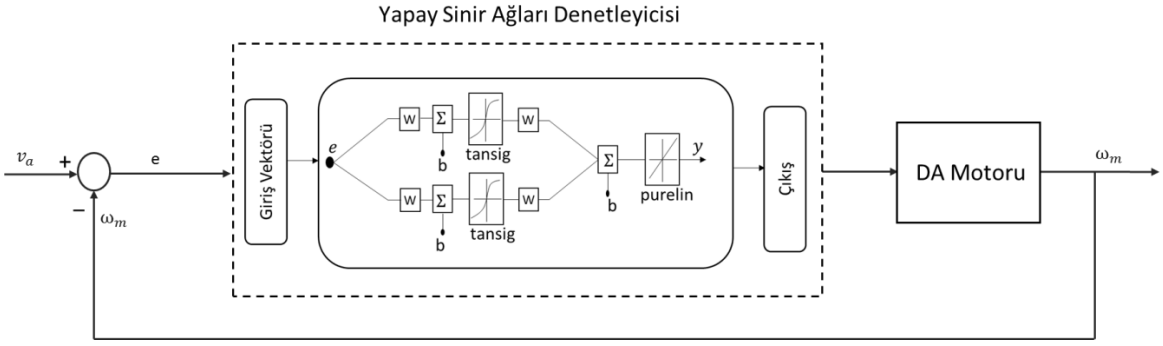
e \ de	NB	NS	Z	PS	PB
PB	Z	PS	PS	PB	PB
PS	NS	Z	PS	PS	PB
Z	NS	NS	Z	PS	PS
NS	NB	NS	NS	Z	PS
NB	NB	PB	NS	NS	Z

Bulanık Denetleyicinin sabit hızda çalışması Şekil 44’ de verilmiştir.

2.2.3. DA Motorunun Yapay Sinir Ağları ile Kontrolü

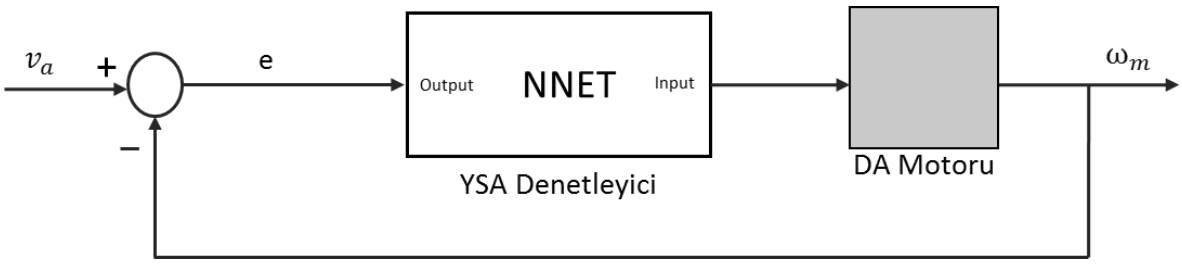
Doğru akım motorunu control etmek için kullanılabilecek akıllı sistemlerden bir diğeri de Yapay Sinir Ağlarıdır. Yapay Sinir Ağları, parametre değişiklikleinden çabuk etkilenmemesi ve karmaşık sistemdlerde iyi sonuç vermesi özelliklerinden dolayı, DA motorunun kontrolünde tercih edilebilir yöntemlerden biri olmuştur. Aynı zamanda Bulanık Mantıkta üyelik fonksiyonlarının şeklinin ve kural tablosunun belirlenmesindeki zorluklardan dolayı da Yapay Sinir Ağlarına yönelim olmuştur.

DA motorunun Yapay Sinir Ağlarıyla kontrolünü gösteren genel diyagramı Şekil 42' deki gibi oluşturulabilir.



Şekil 42. DA Motorunun yapay sinir ağları ile kontrolünün genel şeması

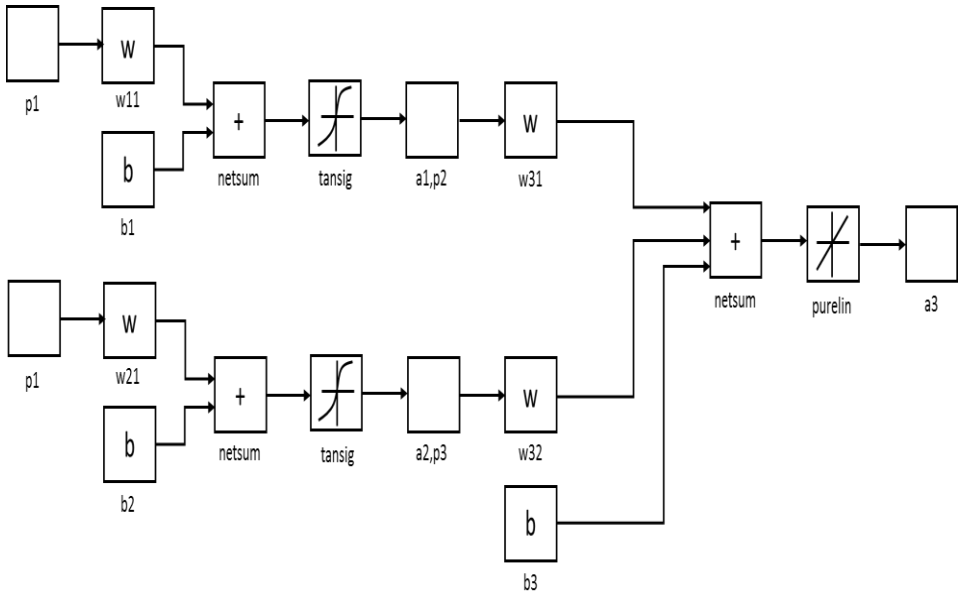
Bunun Matlab/SIMULINKte uygulaması Şekil 43'deki gibi olacaktır.



Şekil 43. DA Motorunun yapay sinir ağlarıyla ile kontrolünün benzetim modeli

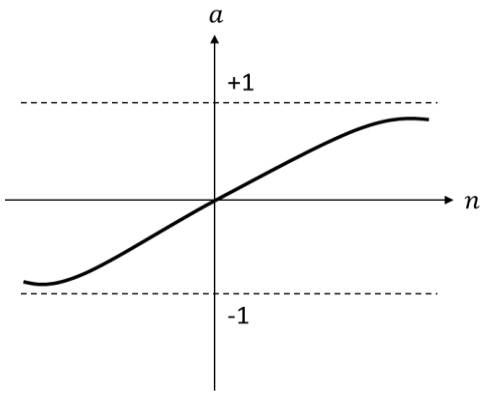
Burada, girişler beraber üç katmanı bulunan ikinci katmanında tansig fonksiyonu kullanan tek giriş tek çıkışlı iki neuronlar bulunan ikinci katmanında purelin fonksiyonunu kullanan iki giriş bir çıkışlı neuron bulunan Yapay Sinir Ağı kullanılmıştır.

Bu Yapay Sinir Ağının oluşturulmuş genel modeli Şekil 44'de gösterilmiştir.



Şekil 44. Sistemde kullanılan yapay sinir ağı modeli

İlk katmandaki fonksiyon, Yapay Sinir Ağlarının non-lineerliklerin üstesinden gelebilmesini sağlayan tansig fonksiyonudur. Bu fonksiyonun grafiği Şekil 45' de ve fonksiyonun denklemleri de (44) ve (45)' de verilmiştir.

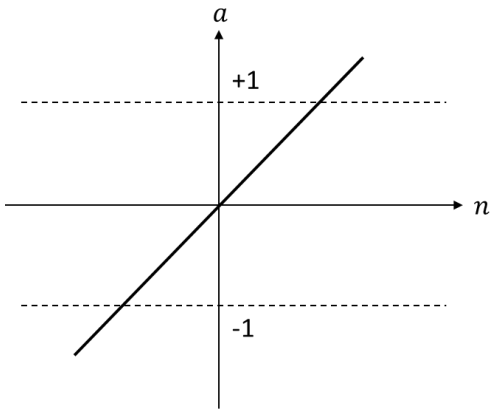


Şekil 45. Tansig fonksiyon grafiği

$$a = \tan sig(n) \tag{44}$$

$$a = \frac{2}{1+e^{(-2n)}} - 1 \tag{45}$$

İkinci katmandaki ise lineer bir fonksiyon olan purelin fonksiyonun grafiği Şekil 46' da ve denklemleri (46) ve (47)' de görülmektedir.



Şekil 46. Purelin fonksiyon grafiği

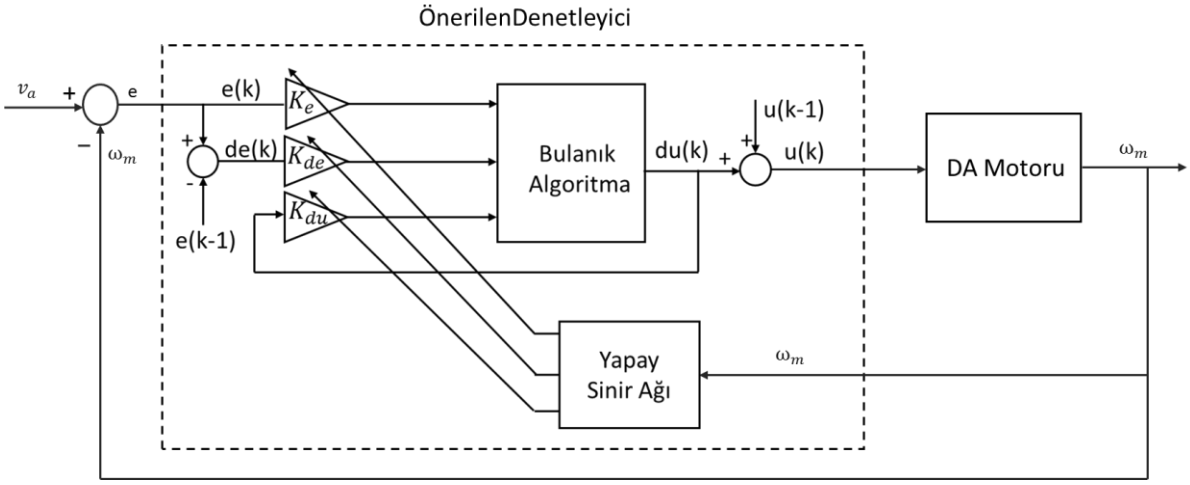
$$a = purelin(n) \tag{46}$$

$$a = n \tag{47}$$

2.2.4. DA Motorunun 'Genetik Algoritma Optimizeli Yapay Sinir Ağlarıyla' Ayarlanan Bulanık Mantık Denetleyici ile Kontrolü

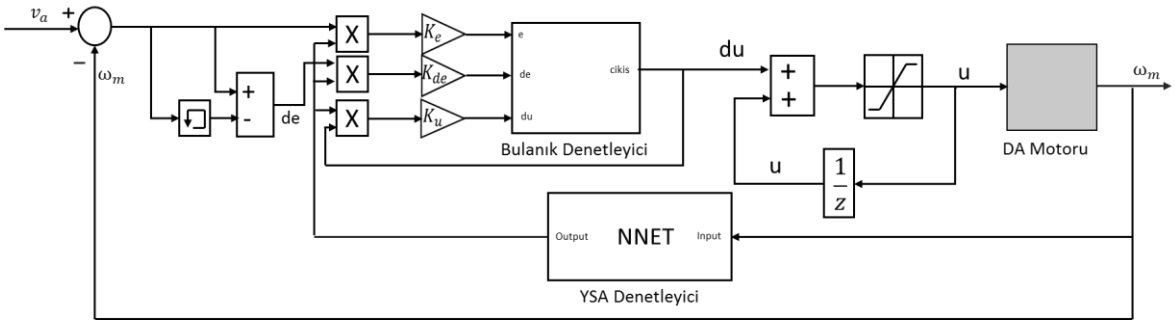
Genetik Algoritma Optimizeli Yapay Sinir Ağlarıyla Ayarlanan Bulanık Mantık Denetleiyici yöntemi, Bulanık Mantıktaki üyelik fonksiyonlarının ayarlanması problemini ortadan kaldırmak ve Yapay Sinir Ağlarıyla birlikte çalıştırılarak değişen parametrelere karşı daha dayanıklı ve takip kabiliyeti daha iyi bir sistem tasarlanması amaçlanmıştır. Önerilen bu yöntemle Doğru Akım Motoru kontrol edilerek, üyelik fonksiyonlarının aralıklarının belirlenmesi için zaman kaybedilmemiş ve daha kararlı bir sistem hedeflenmiştir. Bunun ayarı için kullanılan Yapay Sinir Ağının ağırlıkları da Genetik Algoritma ile ayarlanmıştır.

Bu yöntemde öncelikle Genetik Algoritma ile optimizeli Yapay Sinir Ağına çıkış bağlanılarak sistemin çıkışının nasıl olacağı öğretilir. Bunu yapacak sistemin yapısı Şekil 47' deki gibidir.



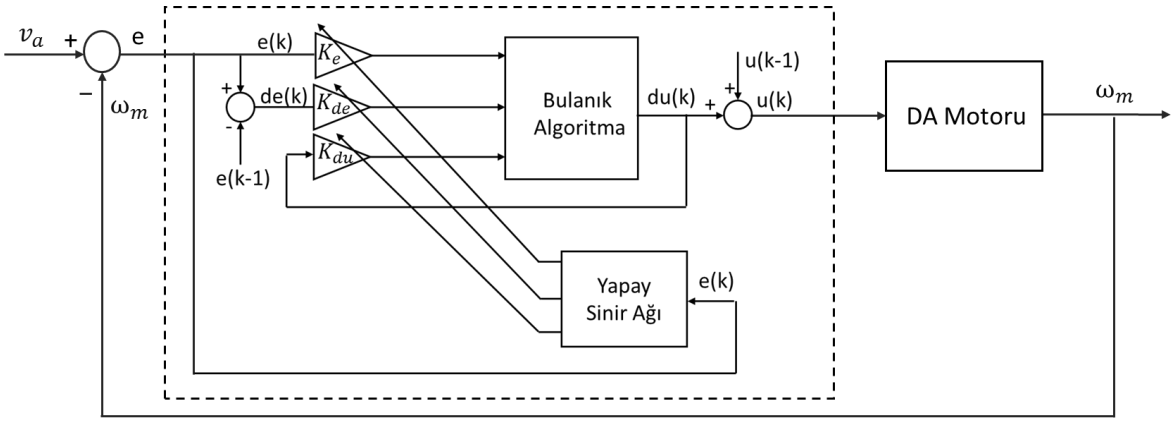
Şekil 47. DA Motorunun çıkışının önerilen denetleyiciye öğretilmesinin genel şeması

Bunun, Matlab/SIMULINK'te uygulanması ise Şekil 48'deki gibidir.



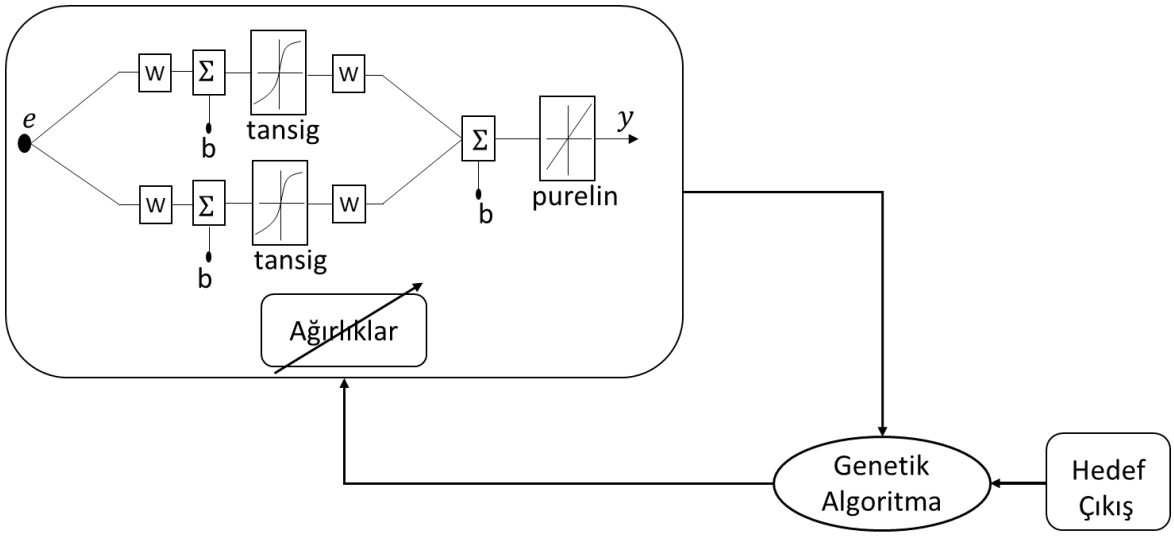
Şekil 48. DA Motorunun çıkışının önderilen denetleyiciye öğretilmesinin benzetim modeli

Öğrenme tamamlandıktan sonra Yapay Sinir Ağı, Bulanık Mantığın aralıklarını ayarlamak üzere kullanılabilir. Bunun için Şekil 49' daki sistem tasarlanmıştır.



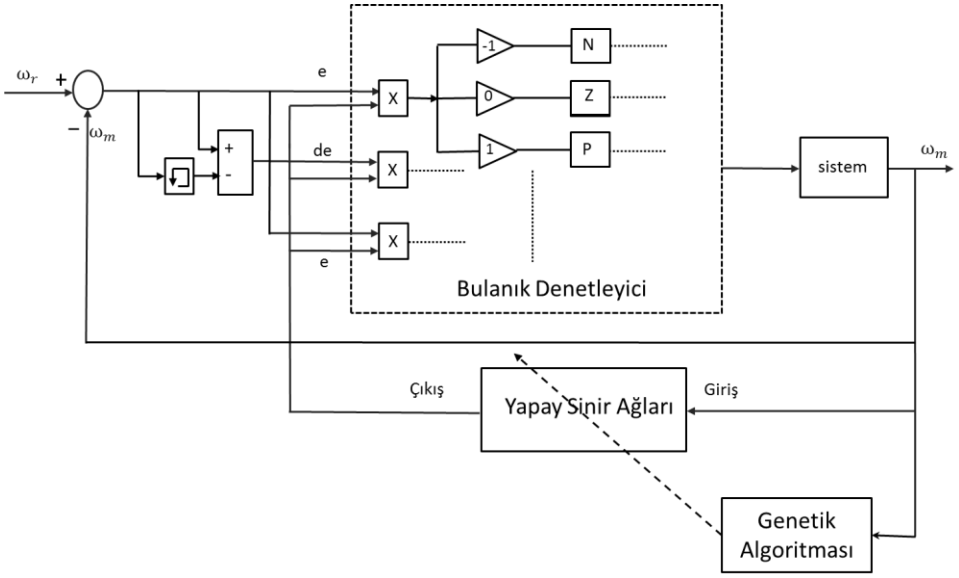
Şekil 49. DA Motorunun önerilen denetleyici ile kontrolünün genel şeması

Burada gösterilen Yapay Sinir Ağı Ağırlıklarının ağırlıkları Genetik Algoritma ile ayarlanarak istenilen çıkışı alabilmek için gerekli en uygun ağırlık belirlenir. Böylece istenen çıkışa göre kendini ayarlayabilecek ve adaptif bir sistem elde edilmiş olacaktır. Bunu yapacak Yapay Sinir Ağının yapısı Şekil 50' deki gibi ifade edilebilir.



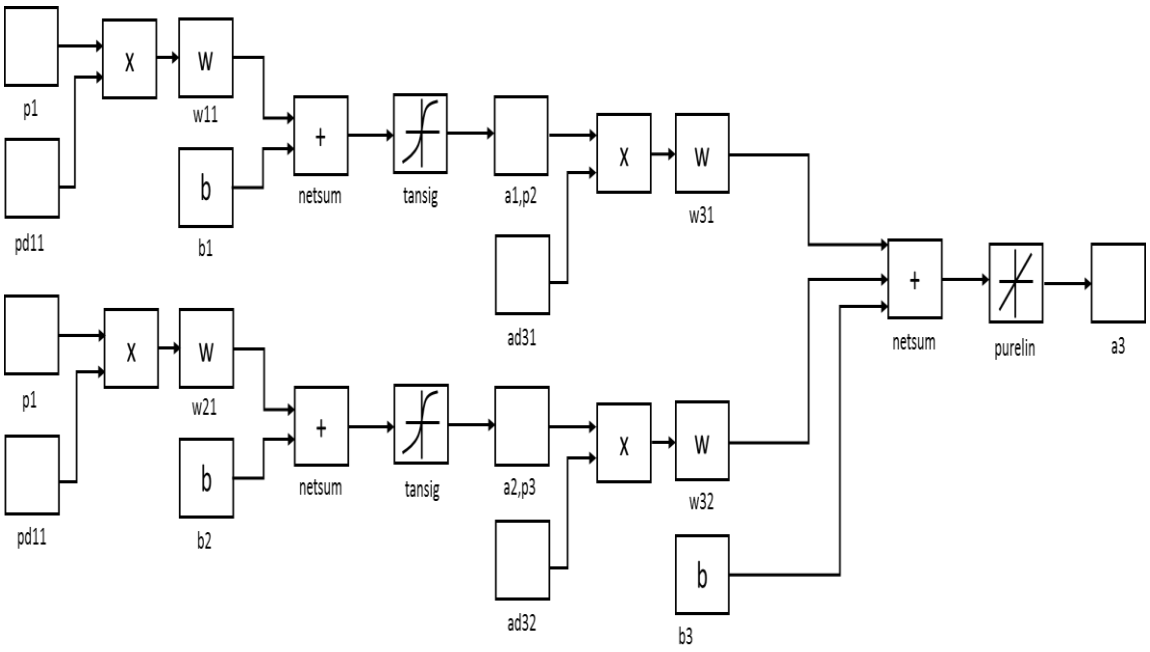
Şekil 50. Önerilen denetleyicide kullanılan yapay sinir ağının yapısı

DA Motorunun Genetik Algoritma Optimizeli Yapay Sinir Ağlarıyla Ayarlanan Bulanık Mantık Denetleyici ile kontrolünün Matlab/SIMULINKte uytgulanması ise Şekil 51'deki gibidir.



Şekil 51. DA Motorunun önerilen denetleyici ile kontrolünün benzetim modeli

Bu sistemde kullanılan, ağırlıkları Genetik Algoritma ile ayarlanan Yapay Sinir Ağının modeli ise Şekil 52'de verilmiştir.

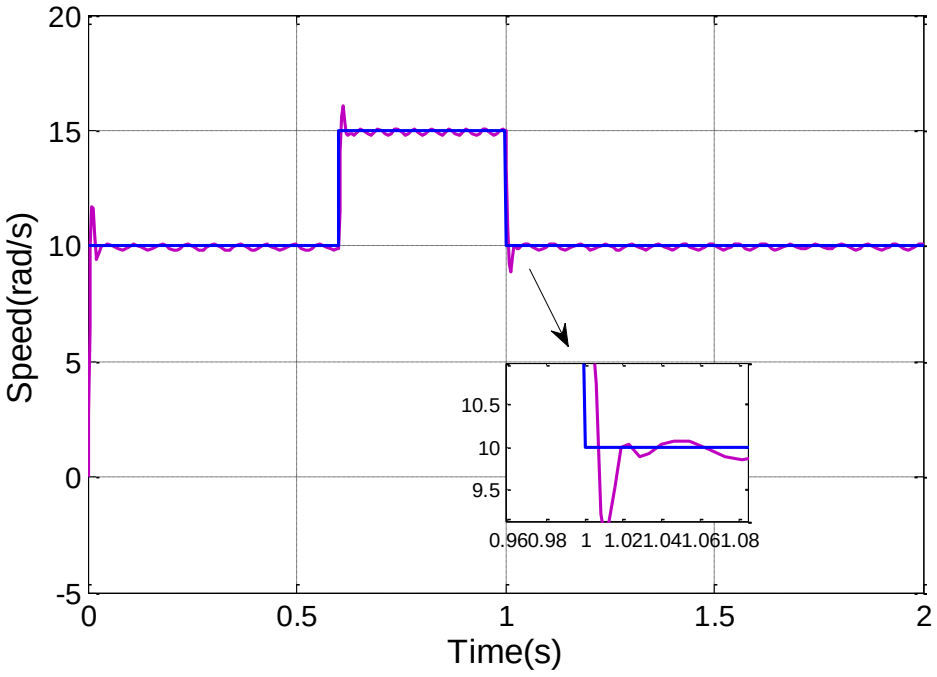


Şekil 52. Önerilen denetleyicide kullanılan yapay sinir ağının benzetim modeli

Önerilen bu yöntemi önce Genetik Algoritma ile optimize etmeden çıktıları elde edilebilir. Böylece Genetik Algotirmanın sistemdeki rolü ve önemi anlaşılmış olacaktır.

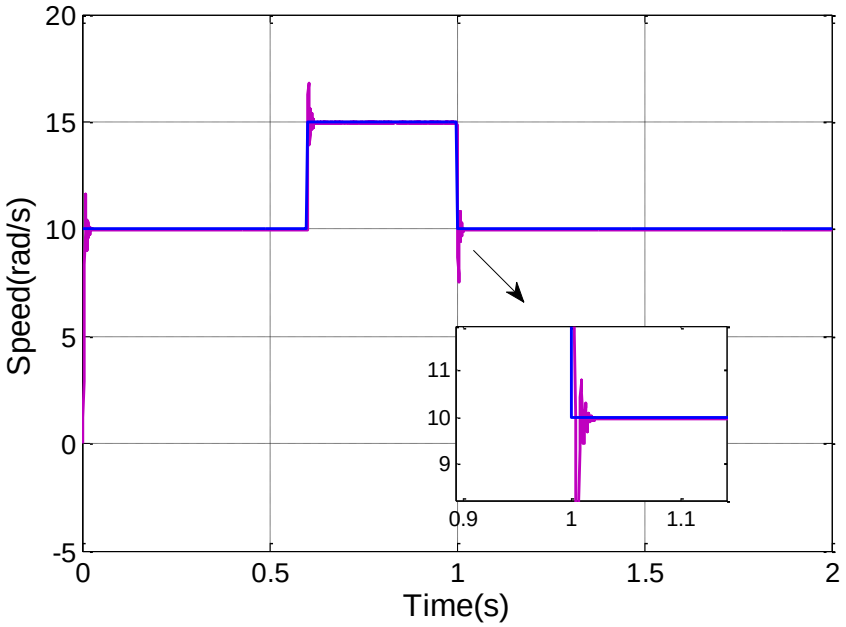
3. TARTIŞMA VE SONUÇLAR

Tezin sonucunda elde edilen çıktılar göstermiştir ki PID denetleyicinin parametreleri ayarlandığında ve referans sabit olduğunda kabul edilir sonuç vermektedir. Ancak PID denetleyici ile kontrol ederken için farklı her referans için yeniden parametre ayarı yapmak gerekmektedir. Bu değerleri bulmak çok zor olduğundan zaman kaybına sebep olmaktadır ve bazen bu değerler çok yüksek ve çok düşük olabilir. Bu da gerçeklikten uzaklaşmakta ve gerçek hayatta uygulanabilirliğini zorlaştırmaktadır. Değişken yük için ise PID ile kontrolde elde edilen çıktı, yükün değiştiği anda fazla bozulmakta ve takip zorluğu çekmektedir. Şekil 53' te ise değişken bir referansta PID denetleyicinin takipte zorlanıp aşma yaptığı görülmektedir.



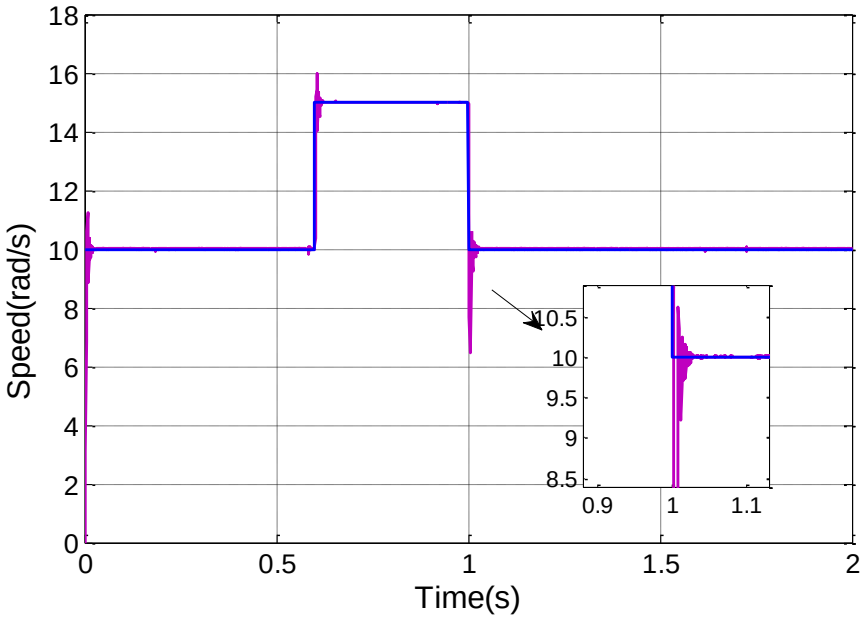
Şekil 53. PID ile DA motoru hız kontrolü çıktısı

Bulanık Mantık akıllı bir sistem olduğundan Şekil 54 gösteriyor ki PID'ye göre daha iyidir ve değişken yükte bozulması daha az olmaktadır. Ancak sistemin oturma süresi PID ye göre daha uzundur ve Bulanık Mantık Denetleyicide parametrelerin ayarlanması sorun olduğundan bu uygulama için tek başına kullanılması çok da uygun olmamaktadır.



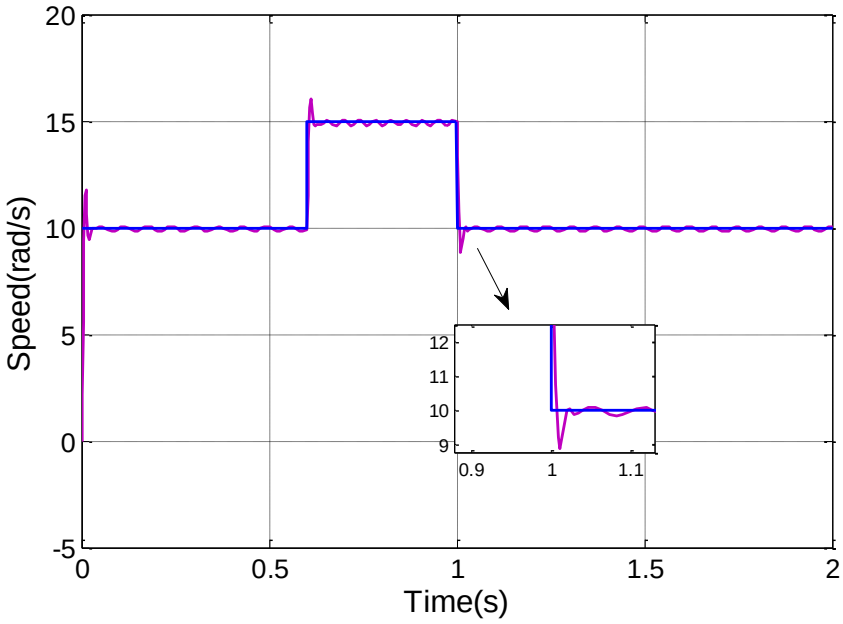
Şekil 54. FLC ile DA motoru hız kontrolü çıktısı

Şekil 55'den görüldüğü gibi Yapay Sinir Ağlarında cevap verme süresi diğerlerine nispeten daha yüksektir ve değişken yüke karşı daha dayanıklıdır. Ancak yeterli eğitim verisi (training data)ni bulabilmek de Yapay Sinir Ağlarının sorunu olduğu için bu da tek başına yeterli değildir ve salınımı diğer sistemlere göre daha fazla olması Yapay Sinir Ağlarının bu uygulama için tek başına yeterli olmamasının bir kanıtıdır.



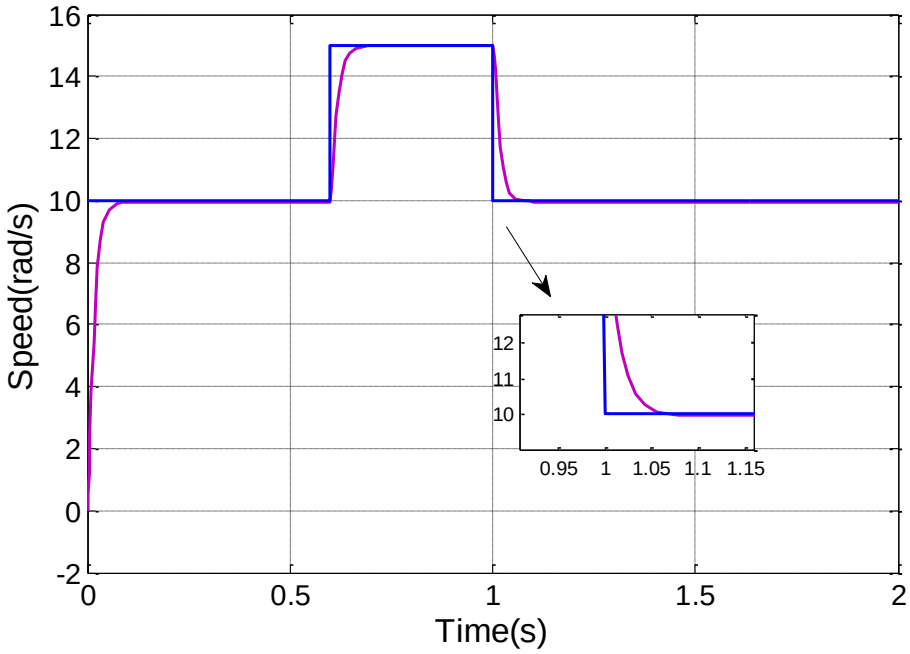
Şekil 55. Yapay sinir ağları ile kontrol edilen doğru akım motorunun çıktısı

Bulanık Mantıktaki parametre ayarlama sorunu için Yapay Sinir Ağları parametre ayarlayıcı olarak kullanılmıştır. Adaptif olması yönüyle diğer yöntemlere göre daha makul bir denetleyici elde edildiği Şekil 56'da görülmektedir.



Şekil 56. Genetik algoritma ile optimize edilmemiş YSA ile ayarlanan BMD'nin DA motor kontrol çıktısı

Değişken referansta daha dayanıklı bir sistem arayışı sonucu, Genetik Algoritma ile optimize edilen Yapay Sinir Ağlarıyla üyelik fonksiyon aralıklarının ayarlandığı bir Bulanık Mantığın hem değişikliklere karşı daha dayanıklı olacağı hem de akıllı sistemlerdeki sorunları da giderebileceği düşünülmüştür ve bu düşünceyle yeni bir dentleyici olarak önerilmiştir. Şekil 57' de gösteriyor ki değişikliklerde daha az bozulan ve sonrasında hemen sistemi takip edebilen bir denetleyici elde edilmiştir.



Şekil 57. Genetik algoritma ile optimize edilmiş YSA ile ayarlanan BMD'nin DA motor kontrol çıktısı

4. ÖNERİLER

Tezde kullanılan Bulanık Mantıkğı üyelik fonksiyonlarının aralıkları ayarlanmıştır. Ancak aralıkları belirlemenin yanında üyelik fonksiyonlarının hangi şeklinin daha iyi sonuç vereceğini belirleyebilecek bir optimize metodu geliştirilebilir. Ya da Bulanık Mantıktaki kural tablosunu belirleyebilecek bir yol oluşturulabilir.

Bu çalışmada kullanılan Yapay Sinir Ağlarının en iyi neuron sayısı denenerek belirlenmiştir. Denemek yerine en iyi neuron sayısını belirleyebilecek bir optimize yöntemi eklenebilir.

Genetik Algoritmasında çaprazlama ve mutasyon dinamik hale getirilebilir. Yani en iyi çaprazlama ve mutasyon oranını belirleyebilecek bir algoritma ile sistem adaptif hale getirilebilir.

5. KAYNAKLAR

1. Sharaf, A.M. ve Altaş, İ.H., A Multi-Zonal Rule Based Variabe Structure Speed Control Scheme for DC Motor Control, The Canadian Conference on Electrical and Computer Engineering, Eylül 1992, Toronto, Ontario, Canada, TA8.19.1- TA8.19.4.

2. Altaş, İ.H., A Fuzzy Logic Controlled Tracking System for Moving Targets, 12th IEEE International Symposium of Intelligent Control, Temmuz, 1997, Istanbul, Bildiriler Kitabı: 43-48.

3. Cakir, B., Yildiz, A. B., Abut, N. ve Inanç, N., DC Motor Control by Using Computer Based Fuzzy Technique, Applied Power Electronics Conference and Exposition, Mart 1999, Texas, Bildiriler Kitabı: 391- 395.

4. Rubaai, A.ve Kotaru, R., Online Identification and Control of a DC Motor Using Learning Adaptation of Neural Networks, IEEE Transactions On Industry Applications, 36, 3 (2000) 935-942.

5. Baruch, I. S., Garrido, R., Flores, J. M. ve Martinez, J.C., An Adaptive Neural Control Of A Dc Motor, IEEE International Symposium on Intelligent Control, Eylül 2001, Mexico City, Bildiriler Kitabı: 121-126.

6. Dzung, P. Q. ve Phuong, L. M., Control System DC Motor with Speed Estimator by Neural Networks, Power Electronics and Drives Systems, Kasım 2005, Kuala Lumpur, Bildiriler Kitabı: 1030-1035.

7. Mosavi, M. R., Rahmati, A. ve Khoshsaadat, A., An ANFIS-Based Neuro-Fuzzy Controller with Supervisory Learning for Speed Control of Brushless DC Motor, International Review on Modelling and Simulations, 4, 5 (2011) 2246-2254.

8. Sabahi, K., Application of ANN Technique for DC- Motor Control by Using FEL Approches, Fifth International Conference on Genetic and Evolutionary Computing, Ağustos 2011, Xiamen, Bildiriler Kitabı: 131-134.

9. Faramarzi, A. ve Sabahi, K., Recurrent Fuzzy Neural Network for DC- Motor Control, Fifth International Conference on Genetic and Evolutionary Computing, Ağustos 2011, Xiamen, Bildiriler Kitabı: 93-96.

10. Shi, A., Yan, M., Li, J., Xu, W. ve Shi, Y., The Research of FUZZY PID Control Application in DC Motor of Automatic Doors, Electrical and Control Engineering (ICECE), Eylül 2011, Yichang, Bildiriler Kitabı: 1354-1358.

11. Uğuz, N. ve Gökkaya, M., Elektrik Makineleri (Doğru Akım), Nüve Matbaası, Ankara, 1981.

12. http://genteknik.net/dc-motorlar, 19 Ocak 2014

13. Bal, G., Doğru Akım Makinaları ve Sürücüleri, Seçkin Yayınları, Ankara, 2008.

14. Altaş, İ.H., Endüvi ve Uyarma Birleşik Kontrolünde DA Motorlarının Modellemesinde Sebep-Sonuç Yaklaşımı, Otomasyon, 189, 3 (2008) 102-110.

15. MEGEP(Mesleki Eğitim ve Öğretim Sisteminin Güçlendirilmesi Projesi), Endüstriyel Otomasyon Teknolojileri PID Kontrolü, Milli Eğitim Bakanlığı Ankara, 2009.

16. G. Cao, C. Lou ve D. An, Application of Fuzzy Self-Tuning PID Control to Air Conditioning Systems, Heating Ventilating and Air Conditioning, 34, 10 (2014) 106-109

17. Wang, J., An D. ve Lou C., Application of Fuzzy-PID Controller in Heating Ventilating and Air-Conditioning System, Proceedings of the IEEE International Conference on Mechatronics and Automation, Haziran 2006, Luoyang, Bildiriler Kitabı: 2017-2222.

18. Altas, I. H. ve Sharaf, A. M., A Generalized Direct Approach for Designing Fuzzy Logic Controllers in Matlab/Simulink GUI Environment, International Journal of Information Technology and Intelligent Computing (Int. J. IT&IC), 1,4 (2007).

19. Zadeh, L. A., Outline of a New Approach to the Analysis of Complex Systems and Decision Processes, IEEE Trans. Systems, man, and Cybernetics, SMC-3, 1 (1973) 28-44.

20. Lee C. C., Fuzzy Logic in Control Systems: Fuzzy Logic Controller I, IEEE Trans. Systems, Man, and Cybernetics, 20, 2 (1990) 404 –418.

21. Lee C. C., Fuzzy Logic in Control Systems: Fuzzy Logic Controller II, IEEE Trans. Systems, Man, and Cybernetics, 20, 2 (1990) 419-435.

22. Maiers J. ve Sherif Y. S., Applications of Fuzzy Set Theory, IEEE Trans. Systems, Man, and Cybernetics, SMC-15, 1(1985) 175-189.

23. Zadeh L. A., Fuzzy Sets, Information and Control, 8, 3 (1965) 338-353.

24. Altas I. H. ve Sharaf A. M., A Fuzzy Logic Power Tracking Controller For A Photovoltaic Energy Conversion Scheme, Electric Power Systems Research, 25, 3 (1992) 227-238.

25. Eminoglu I. ve Altas I. H., A Method to Form Fuzzy Logic Control Rules for a PMDC Motor Drive System, Electric Power Systems Research, 39, 2 (1996) 81-87.

26. Altas I. H., Stabilization of A Two-Link Robot Manipulator Using Fuzzy Logic, The Proceedings of the 3rd International Mechatronic Design and Modeling Workshop, Eylül 1997, Ankara, Bildiriler Kitabı: 41-51.

27. Altas I. H. ve Sharaf A.M., A Novel Fuzzy Logic Controller For Maximum Power Extraction From a PV Array Driving a Threephase Induction Motor, 7th Mediterranean Electrotechnical Conference (MELECON'94), Nisan 1994, Bildiriler Kitabı: 853-856.

28. Mamdani E. H. ve Assilian S., An Experiment in Linguistic Synthesis with A Fuzzy Logic Controller, Int. J. Man-Machine Studies, 7, 1(1975) 1-13.

29. Altas I. H, The Effects of Fuzziness in Fuzzy Logic Controllers, 2nd International Symposium on Intelligent Manufacturing Systems, Ağustos 1998, Sakarya, Bildiriler Kitabı: 211-220.

30. Rojas, R., Neural Networks A Systematic Introduction, Springer-Verlag, Berlin, 1996.

31. Altaş, İ. H.,Neural Fuzzy Systems Ders Notları, Karadeniz Teknik Üniversitesi, 2010.

32. Kumar A., Artificial Neural Networks, Indian Agricultural Research Institute, New Delhi, 2012.

33. Khere, A. ve Patankar, M., Artificial Intelligence: Artificial Neural Network Simplified, H.V.P.M.'s C.O.E.T.,1,1 (2012) 13-17.

34. Wilamowski, B. M., The Industrial Electronics Handbook: Intelligent Systems, CRC Press, USA, 2011.

35. Vieira, J., Dias, F. M. ve Mota, A., Neuro-Fuzzy Systems: A Survey, 5th WSEAS NNA International Conference on Neural Networks and Applications, Kasım 2012, Udine.

36. Kelly ve W. E., Neuro-Fuzzy Control of A Robotic Arm, Yüksek Lisans Tezi, Texas A&M University Kingsville, 1994

37. Kumar, S. ve Baranwal, V., Speed Control of Separately Excited DC Motor Using Neuro Fuzzy Technique, Lisans Tezi, National Institute of Technology Rourkela Rourkela-Orissa, 2009-10.

38. Lee, S. C. ve Lee, E. T., Fuzzy Sets and Neural Networks, Journal of Cybernetics, 4., 2 (1974) 83 -103.

39. Lea, R., Jani, Y. ve Berenji, H. , Fuzzy Logic Controller with Reinforcement Learning for Proximity Operations and Docking, Fifth IEEE International Symposium on Intelligent Control, Eylül 1990, Philadelphia, Bildiriler Kitabı: 903.

40. Pal, S. K. ve Mitra, S., Multilayer Perceptron, Fuzzy Sets, and Classification, IEEE Transactions on Neural Networks, 3, 5 (1992) 683-697.

41. Altrock, C. ve Krause, B., Fuzzy Logic and Neurofuzzy Technologies in Embedded Automotive Applications, Industrial Fuzzy Control and Intelligent Systems, Aralık 1993, Houston, Bildiriler Kitabı: A113-1 - A113-9.

42. Nauck, D., Klawon, F. ve Kruse, R., Foundations of Neuro-Fuzzy Systems, J.Wiley & Sons, New York, 1997.

43. Jang, R., Neuro-Fuzzy Modelling: Architectures, Analysis and Applications, Doktora Tezi, University of California Berkley, 1992.

44. Lin, T. C. ve Lee, C. S., Neural Network Based Fuzzy Logic Control and Decision System, IEEE Transactions on Computers, 40, 12 (1991) 1320-1336.

45. Berenji, H. R. ve Khedkar, P., Learning and Tuning Fuzzy Logic Controllers Through Reinforcements, IEEE Transactions on NeuralNetworks, 3, 5 (1992) 724-740.

46. Tano, S., Oyama, T. ve Arnould, T., Deep Combination of Fuzzy Inference and Neural Network in Fuzzy Inference, Fuzzy Sets and Systems, 82, 2 (1996) 151-160.

47. Chambers L., The Practical Handbook of Genetic Algorithm: Application Volume I, Chapman & Hall/CRC, USA, 2000.

48. Reeves, C.R. ve Rowe, Genetic Algorithms: Principles and Perspectives A Guide to GA Theory, Kluwer Academic Publishers, Dordrecht, 2003.

49. Mathew, T.V., Genetic Algorithm, Yüksek Lisans Projesi, Indian Institute of Technology Bombay,1999.

50. Holland, J., Adaptation in Natural and Artificial Systems, Ann Arbor: The University of Michigan Press, 1975.

51. Goldberg, D.E., Genetic Algorithm in Search, Optimization, and Machine Learning, Addision-Wesley, Reading, Mass., 1989.

52. Goldberg, D.E., Genetic and Evolutionary Algorithms Come of Age, Communications of the ACM, 37, 3(1994) 113-119.

53. Forest, S., Genetic Algorithms: Principles of Natural Selection Applied to Computation, Science, 261, 5123 (1993) 872-878.

54. Baeck, T. ve Schwefel, H.-P., An Overview of Evolutionary Algorithms for Parameter Optimization, Evolutionary Computation, 1, 1(1993) 1–23.

ÖZGEÇMİŞ

19.04.1980 Tarihine Iran'da doğdu. İlk, ortaokul ve lise öğrenimini Khoy'da MODARES'da tamamladı. 2001 yılında başladığı Sahand Tabriz Teknik Üniversitesi Mühendislik Fakültesi Elektrik-Elektronik Mühendisliği Bölümünden 2006 yılında Elektrik-Elektronik Mühendisi Ünvanıyla mezun oldu. 2006 yılından 2010 yılına kadar araba fabrikası Fork Lift Sahand'de çalıştı. 2010 yılında Karadeniz Teknik Üniversitesi Fen Bilimler Enistitüsü Elektrik-Elektronik Mühendisliği Anabilim Dalı'ndan yüksek lisans öğreniminee başladı. Yüksek lisans öğreniminin bir dönemini erasmus öğrenci değişimi programı kapsamında Kassel Üniversitesinde geçti. Yabanci dil olarak Farsça Türkçe İngilizce ve Almanca bilmektedir.