Anatolian Journal of Computer Sciences

Volume:3 No:2 2018 © Anatolian Science

pp: 1- 14 ISSN:2548-1304

Fidan Gelişim ve Lig Şampiyonluk Algoritmaları Kullanarak Aşırı-Akım Rölelerin Optimum Koordinasyonu

Optimal coordination of over-current relays using saplings growing up and league championship algorithms

Ebubekir SEYYARER[1], Cengiz HARK[2], Faruk AYATA[1], Taner UÇKAN[1], Ali KARCI[3]

[1]Van Yüzüncü Yıl Üniversitesi, Bilgisayar Teknolojileri Bölümü, Van, Türkiye (eseyyarer@gmail.com, eseyyarer@yyu.edu.tr, ayatafaruk@gmail.com, taner_uckan@hotmail.com)

[2]Bitlis Eren Üniversitesi, Bilgisayar Teknolojileri Bölümü, Bitlis, Türkiye (cengizhark@hotmail.com)

[3]İnönü Üniversitesi, Bilgisayar Mühendisliği Bölümü, Malatya, Türkiye (ali.karci@inonu.edu.tr)

Gönderim Tarihi:29.02.2018

Kabul Tarihi:23.07.2018

Yayın Tarihi: Eylül 2018

Özetçe: Günümüzde oldukça fazla sayıda metasezgisel algoritma yöntemi önerilmiş ve birçok genel probleme uygulanarak kullanılmaktadırlar. Bu metasezgisel yöntemlerden bazıları ise belirli ve özel problemlere uygulanmışlardır. Genetik Algoritma, Yapay Sinir Ağları, Parçacık Sürü Optimizasyonu vb. 150'ye yakın metasezgisel yaklaşım mevcuttur ve her geçen gün bir başka problemin çözümünde kullanılmaktadırlar. Fidan Gelişim Algoritması (FGA) ve Lig Şampiyonluk Algoritması (LCA) da son yıllarda popüler olan metasezgisel yaklaşımlardır. Enerji sürekliliğinin oldukça önemli olduğu güç sistemlerinde, sürekliliğin sağlanması noktasında çeşitli yöntemler söz konusudur. Bu yöntemlerden birisi de iyi bir aşırı-akım röle koordinasyonudur. Bu çalışmada, elektrik enerjisi nakil hatlarında sürekli bir güç işletmeciliğinin sağlanması, yerinde koruma koordinasyonun kurulması ve dağıtım sistemlerinde meydana gelebilecek olası arızaların önüne geçilebilmesi adına metasezgisel algoritma yöntemlerinden FGA ve LCA kullanılmıştır. Gerekli uyarlama ve kodlamalar .NET platformu kullanılarak gerçekleştirilmiş ve aşırı-akım röle koordinasyon değerlerinde oldukça iyi sayılabilecek sonuçlara ulaşılmıştır.

Anahtar kelimeler: *Aşırı-akım röle koordinasyonu, Fidan gelişim algoritması, Lig şampiyonluk algoritması, C#, .NET.*

Abstract: Nowadays, a large number of metaheuristic algorithm methods have been proposed and used in many general problems. Some of these metaheuristic methods have been applied to specific and private problems. Genetic Algorithm, Artificial Neural Networks, Particle Swarm Optimization, etc. There are close to 150 metaheuristic approaches and they are used day by day to solve another problem. Fidan Developmental Algorithm (FGA) and League Championship Algorithm (LCA) are also popular metaheuristic approaches in recent years. In power systems where energy continuity is very important, there are various methods at the point of maintaining continuity. One of these methods is good over-current relay coordination. In this study, FGA and LCA are used in metaheuristic algorithms in order to provide continuous power management in electrical energy transmission lines, to establish on-site protection coordination and to prevent possible failures in distribution systems. The required adaptations and encodings were implemented using the .NET platform and the results have been reached which can be considered quite good in the over-current relay coordination values.

Keywords: *Over-current relay coordination, Saplings growing up algorithm, league championship algorithm, C#, .NET.*

1 Giriş

Günümüzde bazı sistemlerin matematiksel olarak modellenmesi zordur, bu sistemlerin çözülebilmesi için farklı yaklaşımlar önerilmiştir. Bu geliştirilen yaklaşımlardan en popüler olanları; genetik algoritma, yapay sinir ağları vb. yaklaşımlardır. Bu yöntemlerin avantajları olduğu gibi dezavantajları da bulunmaktadır. Karmaşık matematiksel modellere ihtiyaç duymamaları ve tüm problemlere uygulanmaları en büyük avantajlarıdır. Fakat bazı özel problemlerde zamansal olarak kötü performans göstermeleri ise dezavantajlarıdır. Bu özel problemlere yönelik ise genel olmayan ve yine doğadan esinlenip geliştirilen yöntemler önerilmektedir. Optimizasyon ağırlıklı olan bu yöntemlerin başarı oranları oldukça yüksektir [1].

Herhangi bir problemin tüm çözümleri arasında en iyi çözümü elde etmek, optimizasyon olarak adlandırılır. Mühendislik başta olmak üzere neredeyse her branşta optimizasyon işlemi yapılmaktadır ve bu yüzden birçok optimizasyon tekniği geliştirilmiştir. Zor olmasına rağmen karmaşık sistemler için bu yöntemler matematiksel modele ihtiyaç duymaktadırlar. Modelin kurulması durumunda bile çözüm için gereken zaman maliyeti çok yüksek çıkmaktadır [1].

Yukarıda ismi verilen optimizasyon tekniklerinden Genetik Algoritma; evrim metodolojisinden esinlenerek 1975'te Holland tarafından ortaya atılan ve 1989'da Goldberg'in çalışmalarıyla çok geniş uygulama alanlarında kullanılabilir hale gelmiştir. Bu yöntemin amacı; en iyi bireyin (çözümün) hayatta kalması ilkesine dayanmaktadır [1,2].

Diğer bir yöntem ise; insan beyninin işleyişinden esinlenerek geliştirilmiş olan Yapay Sinir Ağlarıdır. Bu sistem, yazılım mühendisliği modellerinden kara kutu gibi çalışmaktadır. İçyapısı bilinmeyen yöntem kendisine verilen girişlere karşı doğru çıkışlar elde etmeye çalışmaktadır. Daha önce gördüğü örneklerden çıkarım yaparak doğru sonucu bulmaya çalışan insan beyni gibi, yapay sinir ağlarının öğrenme yolu ile doğru sonuca yakınsamaya çalışmaktadır [3,4].

Bu makalede çözülmeye çalışılan mühendislik probleminde literatürde ilk olmak üzere FGA ve LCA algoritmaları kullanılmaktadır.

FGA; 2006 yılında Karci tarafından geliştirilmiştir. Yöntem, isminden de anlaşıldığı gibi fidanların gelişimlerinden esinlenilerek ortaya atılan bir optimizasyon tekniğidir. Bu hesaplama yöntemi fidanların ekilmesi, yetiştirilmesi ve eşleştirilmesine (çiftleştirilmesi) dayanmaktadır. Optimizasyon ve arama problemleri için kullanılmaktadır [5].

FGA'nın başarılı olmasındaki en büyük etken başlangıç popülasyonunun düzenli olarak başlatılmasıdır. Böylelikle başlangıç popülasyonu, çözüm uzayından fazla uzaklaşmadan ve fazla zaman harcamadan en uygun çözüme ulaşmayı hedeflemektedir.

İris plant database 'den 3 sınıfa (iris-setosa, iris-versicolor, iris-virginica) ait alınan 150 adet veri setine fidan gelişim algoritması (FGA) uygulayarak kümeleme işlemi yapılmıştır. Her sınıftan 20 'si eğitim, 30'u ise test verisi olmak koşuluyla 50 adet örnek alınmıştır. Veri setindeki her bir örneğin 4 niteliği bulunmaktadır. Bu örneklerin her biri, FGA 'da bir fidan olarak temsil edilmektedir. Bu veri setine daha önce uygulanan algoritmalarda alınan başarı oranı %33 iken FGA 'daki başarı oranı %79,33 olarak gözlemlenmiştir [1].

TRANSFAC veri tabanından alınan 4 farklı DNA dizilerinde motif keşfi için fidan gelişim algoritması uygulanmıştır. Bunlar dm01g.fasta, dm01r.fasta, mus05r.fasta, hm15r.fasta DNA dizileridir. Motif keşfi için daha önceden yapılan çalışmalar; AlignACE, MEME, MEME3, MotifSampler, Consensus, Weeder v.b. çalışmalardır. Yapılan çalışmada elde edilen sonuçlar AlignACE, MEME, MEME3, MotifSampler, Consensus, Weeder v.b. çalışmalarda elde dilen sonuçlarla kıyaslama yapılmış ve fidan gelişim algoritmasının çok daha başarılı sonuçlar verdiği gözlemlenmiştir [10].

LCA; Dr. Ali Husseinzadeh Kashan tarafından 2013 yılında önerilmiştir. Yine isminden de anlaşıldığı gibi bir spor ligini taklit ederek ortaya atılan bir yöntemdir.

Örnek bir güç sisteminde çok amaçlı reaktif güç dağıtımı probleminin çözümü için LCA algoritması uygulanmıştır. Statik voltaj kararlılık değerlendirmesi için sistemin modal analizi kullanılır. Gerilim kararlılık marjının kayıp minimizasyonu ve maksimizasyonu hedef olarak alınır. Jeneratör terminal gerilimleri, kapasitör banklarının reaktif güç üretimi ve kademe değiştiren transformatör ayarı optimizasyon değişkenleri olarak alınır. LCA algoritmasını değerlendirmek için IEEE 30 bus sisteminde test edilmiş ve daha önce literatürde bildirilen diğer algoritmalar ile karşılaştırılmıştır. Sonuçlar, tek amaçlı reaktif güç dağıtımı probleminin çözümü için LCA'nın diğerlerine göre daha verimli olduğunu göstermektedir [8].

Bir servis bulutu (IaaS) olarak altyapının görev planlanmasında tamamlanma zamanı optimizasyonu NP-hard problem tipindedir. Bu optimizasyon için spor tabanlı LCA uygulanmıştır ve algoritmanın performansını değerlendirmek için, First Come First Served (FCFS), Last Job First (LJF) ve Best Effort First (BEF) olmak üzere üç tane mevcut algoritma kullanılmıştır. Elde edilen sonuçlar, LCA programlama tekniğinin IaaS bulutunda zamanlanmış görevlerin yapım süresini en aza indirgemede diğer algoritmalardan daha iyi performans gösterdiği gözlemlenmiştir [9].

Santrallerde farklı üretim teknikleri ile elde edilen elektrik enerjisi, iletim sistemlerine ve hemen sonrasında dağıtım sistemlerine aktarılarak tüketicilere ulaştırılır. Elektrik üretim santralleri birçok nedenden ötürü şehirlerden uzaklara konumlanmışlardır. Bu durum elektrik enerjisinin, enerji nakil hatları aracılığı ile taşınması gerekliliğini de beraberinde getirmektedir. Türkiye'de 380kV ve 154kV nominal gerilimler kullanılarak enerji iletimi gerçekleştirilmektedir. Trafo merkezlerinde yer alan güç trafoları aracılığı ile gerilim 154/380 kV düzeyinden 30/36 kV seviyelerine indirgenmektedir. Bütün bu aşamalar geride bırakılarak elektrik enerjisinin tüketicilere ulaştırılmasında süreklilik önem arz etmektedir. Enerjinin sözü edilen bu sürekliliği ancak iyi bir güç işletmeciliği ve yerinde bir koruma koordinasyonu ile elde edilebilmektedir. Örneğin yerinde bir koruma koordinasyonu ile dağıtım sisteminde meydana gelebilecek olası bir arızanın sadece ilgili fider ile sınırlı kalması sağlanabilir. Aksi halde ilgili fiderde meydana gelen arıza iletim sisteminde yer alan güç trafosuna yansıyabilir. İyi planlanmış bir aşırı akım röle koordinasyonu ile yukarıda anlatılan olumsuz durumların üstesinden gelinebilir. Bu çalışmada yerinde bir aşırı akım röle koordinasyonunun sağlanabilmesi için FGA ve LCA algoritmaları kullanılmış ve 154 kV trafo merkezinde ters zamanlı aşırı akım röle koordinasyonu gerçekleştirilmiştir. Hassas röle ayarlarının bu tip algoritmalar ile sağlanması ile iyi bir aşırı akım koordinasyonu elde edilebilir. Bu sebepten ötürü literatürde aşırı akım koordinasyonu elde edilirken sıklıkla algoritma tabanlı çalışmalara rastlanmaktadır [6].

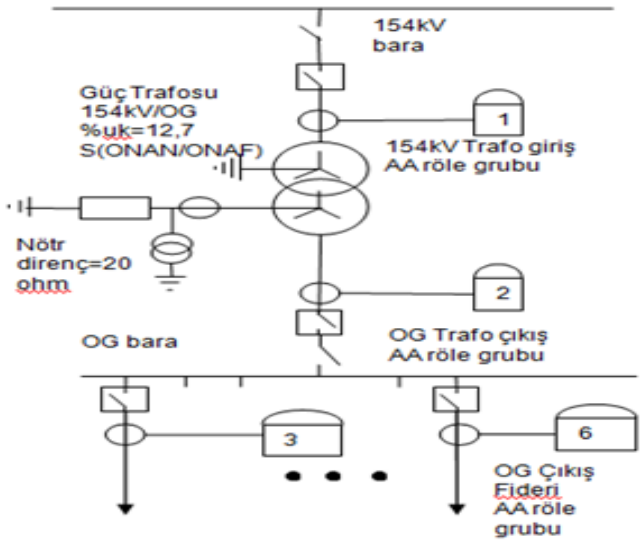
2 Problemin Tanımı ve Kullanılan Yöntemler

2.1 Aşırı Akım Röle Koordinasyonu

Güç sistemlerinde çeşitli akım seviyelerinde arızalar meydana gelmektedir. Bu arızalar genelde mevcut güç sisteminin taşıyacağı maksimum akım seviyesinden fazla olan akım değerleridir. Bu yüzden bu arıza akımlarından güç sistemi teçhizatlarının korunması gerekmektedir. Güç sistemlerinde koruma işlemleri farklı tip sigorta ve röleler ile yapılabilir. Genelde ülkemizde aşırı akımlardan güç sistemi teçhizatları aşırı akım röleleri ile korunur. Özellikle günümüzde eski tip olan mekanik aşırı akım rölelerinden sayısal aşırı akım rölelerine yönelimin artması ile koruma koordinasyonu daha iyi yapılabilmektedir. Dijital röleler akım-zaman eğrilerine göre sabit zamanlı ya da ters zamanlı olarak ayarlanabilir. Bu çalışmada TEİAŞ (Türkiye Elektrik İletim Anonim Şirketi) güç sisteminde genelde aşırı akım koordinasyonunda kullanılan, faz-faz kısa devre (154 kV/OG 3-Faz Aşırı Akım) arızalarında ters zamanlı ayar temel alınarak aşırı akım röle koordinasyonu yapılmıştır. Aşırı akım röleleri ileri ve geri yönde koruma yapabilir. Bu çalışmada TEİAŞ sisteminde fazlası ile kullanılmasından dolayı yönsüz aşırı akım koordinasyonu yapılmıştır [6].

2.1.1 154 kV/OG 3-Faz Aşırı Akım Koordinasyonu

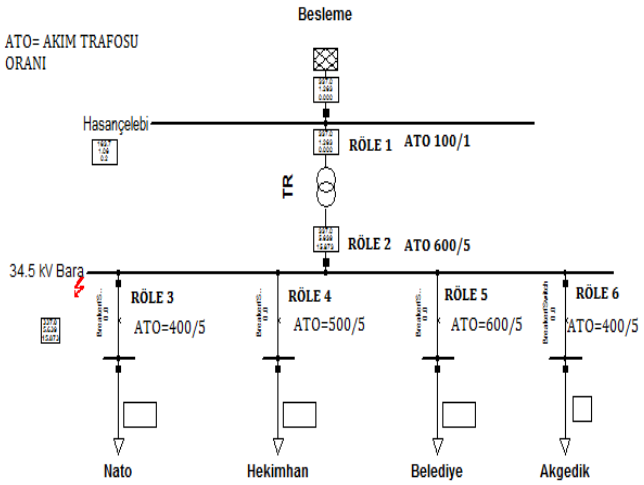
3-Faz aşırı akım koordinasyonu yapılacak sistemin tek hat şeması Şekil 1'de gösterildiği gibidir.



Şekil 1. 154kV/OG sistemi tek hat diyagramı [6].

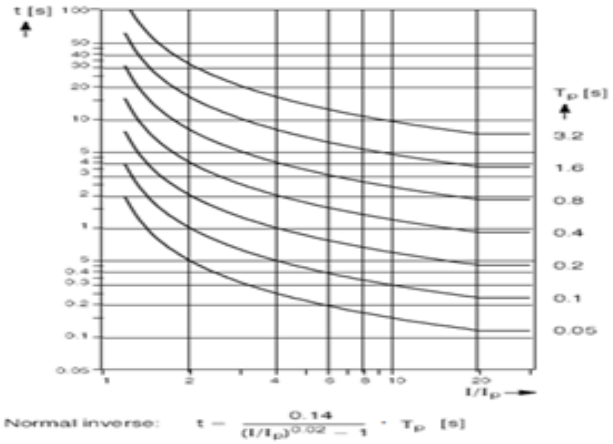
Bu çalışmada 3-faz aşırı akım koordinasyonu Malatya iline ait örnek bir trafo merkezinin TR-A'nın sadece devrede olduğu bir senaryoya bağlı kalarak yapılmıştır [6].

Şekil 2'de verilen röle1/2/3/4/5/6'da ters zaman eğrisine göre aşırı akım ayarı yapılır. Ters zaman eğrisi Şekil 3'te görüldüğü gibi iki değer yardımıyla seçilir. Bu değerler başlatma akım değeri (I_p) ve zaman sabiti t_d(s)'dir. Bu çalışmada aşırı akım röle değerlerinin koordinasyonunda bulunacak değerler lig şampiyonluk algoritması kullanılarak hesaplanmıştır. Böylece en optimum koruma ayarları bulunmaya çalışılmıştır. Daha sonra lig şampiyonluk algoritması ile bulunan değerlerin modellemesi DigSilent programı ile yapılan güç sistemindeki aşırı-akım rölelerine uygulanmıştır ve bu değerler doğrultusunda ayarlanan rölelerin kısa devre arızalarına tepkisi analiz edilmiştir [6].



Şekil 2. Örnek Trafo Merkezi Tek Hat şeması [6].

Bu çalışmada gerçek güç sisteminden farklı olarak sadece akım trafo oranları (ATO) değiştirilmiştir. Diğer değerler gerçek güç sisteminden alınmıştır.



Şekil 3. Ters zaman eğrisi[6].

2.2 Fidan Gelişim Algoritması

Doğada fidan yetiştiriciliği iki aşamadan oluşur; birincisi fidanın ekimi, ikincisi fidanın büyümesidir (dallanma, eşleşme ve aşılama) [7].

Eşleşme operatörü, iki fidan arasında bilgi alışverişi yapan genel bir arama operatörüdür. Dallanma operatörü, olasılıksal olarak bir fidanın dallarını değiştiren yerel bir arama operatörüdür. Aşılama operatörü ise, benzer fidanlar arasında bilgi alışverişini yapan bir arama operatörüdür. Kıyaslanma fonksiyonlarına önerilen bu metodun uygulamasından sonra, bu metodun fonksiyonların değerlendirme sayısı ve daha iyi çözümler bulma bakımından genetik algoritmaya göre daha üstün olduğu gözlemlenmiştir [7].

Fidan ekimi; fidan büyümesi ve fidan eşleşmesine dayanan arama ve optimizasyon problemleri için sunulan yeni bir hesaplama yöntemidir. Fidanların ekilmesi, her yönden birbirine (batı, doğu, kuzey, güney) eşit uzunlukta bir mesafeye sahip olmalıdır. Bu yöntemin ilk basamağıdır. Sonra fidanlar büyür, eşleşir, dallanır ve aşılanır, bu nedenle dört operatör söz konusudur: Eşleştirme, Dallanma, Aşılama ve Hayatta Kalma.

FGA yöntemi aşağıdaki gibi açıklanabilir:

- Ekim Aşaması

Düzgün ekim örneklemesi: Fidanlar uygun olan çözüm alanına eşit şekilde dağılır.

- Büyüme Aşaması

Bu aşama üç operatör içerir:

- ✓ Eşleşme operatörü; mevcut fidanları eşleştirerek yeni fidanlar oluşturmayı amaçlamaktadır. Eşleşme operatörü, iki fidan arasında bilgi alışverişinde bulunan global bir arama operatörüdür.
- ✓ Dallanma operatörü; mevcut dal pozisyonlarına bağlı olarak dal pozisyonunun belirlenmesi için olasılık yöntemini kullanarak mevcut fidanlardan yeni fidanlar üretmeyi amaçlamaktadır.
- ✓ Aşılama operatörü; mevcut fidanlardan benzer yeni fidanlar üretmeyi amaçlıyor. Aşılama operatörü, benzer fidanları kullanan bir arama operatörüdür.

Ekim Aşaması: İlk olarak, iki fidan $S_0 = \{u_1, u_2,\ldots, u_n\}$, $S_1=\{I_1, I_2,\ldots,I_n\}$ olsun, n fidanın boyudur ve bu durumda k = 1 olarak kabul edilir, burada u_i, $1 < i < n$, karşılık gelen değişken için üst sınır değeridir ve I_i, $1 < i < n$, karşılık gelen değişken için alt sınır değeridir. Daha sonra, k'nin bölünme faktörü olduğunu gösteren bir bölen faktör belirlenir. Öncelikle, k = 2 ve iki ilave S_3, S_4 fidanı S_0 ve S_1'den türetilir. Fidan S_0 iki parçaya bölünür (mümkünse eşit uzunlukta), bu durumda 4 fidan ($2^2 = 4$)

S_0'dan türetilebilir. Ancak bunlardan biri S_0 ile aynıdır ve diğeri S_1 ile aynıdır. Daha sonra S_0 ve S_1'den farklı iki fidan türetilebilir.

$S_2=\{l_1+(u_1 - l_1)*r,\ l_2+(u_2 - l_2)*r,\ \ldots,\ l_{n/2}+(u_{n/2} - l_{n/2})*r,\ l_{n/2+1}+(u_{n/2+1}-l_{n/2+1})*(1-r),\ l_{n/2+2}+(u_{n/2+2}-l_{n/2+2})*(1-r)\}$

$S_3=\{l_1+(u_1 - l_1)*(1-r),\ l_2+(u_2 - l_2)*(1-r),\ \ldots,\ l_{n/2}+(u_{n/2} - l_{n/2})*(1-r),\ l_{n/2+1}+(u_{n/2+1}-l_{n/2+1})*r,\ l_{n/2+2}+(u_{n/2+2}-l_{n/2+2})*r\}$

Burada r, $0 < r < 1$ gibi rastgele bir sayıdır. Bahçede kalan fidanlar, k'nin değerinin yükseltilmesiyle aynı yöntemi uygulayarak elde edilecektir. k = 3 durumunda, S_1'den 6 türemiş fidan olacaktır.

$S_4=\{l_1+(u_1 - l_1)*r,\ l_2+(u_2 - l_2)*r,\ \ldots,\ l_{2n/3}+(u_{2n/3}-l_{2n/3})*r,\ l_{2n/3+1}+(u_{2n/3+1}-l_{2n/3+1})*r,\ldots..,l_n+(u_n-l_n)*r\}$

$S_5=\{l_1+(u_1 - l_1)*r,\ l_2+(u_2 - l_2)*r,\ \ldots,\ l_{n/3}+(u_{n/3}-l_{n/3})*r,\ l_{n/3+1}+(u_{n/3+1}-l_{n/3+1})*(1-r),\ \ldots,\ l_{2n/3}+(u_{2n/3}-l_{2n/3})*(1-r),\ l_{2n/3+1}+(u_{2n/3+1}-l_{2n/3+1})*r,\ \ldots,\ l_n+(u_n-l_n)*r\}$

$S_6=\{l_1+(u_1 - l_1)*r,\ l_2+(u_2 - l_2)*r,\ \ldots,\ l_{n/3}+(u_{n/3}-l_{n/3})*r,\ l_{n/3+1}+(u_{n/3+1}-l_{n/3+1})*(1-r),\ \ldots..,\ l_n+(u_n-l_n)*(1-r)\}$

$S_7=\{l_1+(u_1 - l_1)*(1-r),\ l_2+(u_2 - l_2)*(1-r),\ \ldots,\ l_{n/3}+(u_{n/3}-l_{n/3})*(1-r),\ l_{n/3+1}+(u_{n/3+1}-l_{n/3+1})*r,\ \ldots,\ l_n+(u_n-l_n)*r\}$

$S_8=\{l_1+(u_1 - l_1)*(1-r),\ l_2+(u_2 - l_2)*(1-r),\ \ldots,\ l_{n/3}+(u_{n/3}-l_{n/3})*(1-r),\ l_{n/3+1}+(u_{n/3+1}-l_{n/3+1})*r,\ \ldots,\ l_{2n/3}+(u_{2n/3}-l_{2n/3})*r,\ l_{2n/3+1}+(u_{2n/3+1}-l_{2n/3+1})*(1-r),\ \ldots,\ l_n+(u_n-l_n)*(1-r)\}$

$S_9=\{l_1+(u_1 - l_1)*(1-r),\ l_2+(u_2 - l_2)*(1-r),\ \ldots,\ l_{2n/3}+(u_{2n/3}-l_{2n/3})*(1-r),\ l_{2n/3+1}+(u_{2n/3+1}-l_{2n/3+1})*r,\ \ldots,\ l_n+(u_n-l_n)*r\}$

İkili durumda geçerli değer, rasgele bir değerle çarpmak yerine tamamlanır. Fidanların türetilmesi bu şekilde popülasyon tamamlanana kadar devam eder. Örneğin, değişken değerlerinin aralığı [(10, -10), (10, -10), (10, -10), (10, -10)] şeklindedir. O zaman S_0 = {10, 10, 10, 10} ve S_1 = {- 10, -10, -10, -10}. k = 2 durumunda: r = 0.7, 0.4, 0.2, 0.8, S_2 = {4, -2, 6, -6} ve r = 0.5, 0.0, 0.1, 0.9, S_3 = {0, -10, -8, 8}.

Algoritma 1. Başlangıç Popülasyonunu Oluşturmak

//G bir bahçe, I indis kümesi ve Ie genişletilmiş indis kümesi.

1- Dallanma değerlerinin alt sınırlarını içeren G[1] ve üst sınırlarını içeren G[2] gibi iki tane fidan oluştur.
2- Indis←3
3- k←2
4- **While** P doymamış **do**
   ie, Ie'nin bir elemanı olsun, her ie bit değeri ile genişletilmiş ve bu bit değeri parçaya karşılık gelir.
   i←1
   While P doymamış ve özel bir k (i≤2k-2) değeri için tüm fidanlar üretilmemiş **do**
      i bir k-bit sayı ve ie i'nin genişletilmiş değerine karşılık gelir. i'nin her biti G[0] ve G[1]'in karşılık gelen kısmının uzunluğuna kadar genişletilir.
      For j←1 **to** n **do**
         if ie'nin j. biti=1 **then** P[indis]'in j. dallanması=G[1]*r
         else G[indis]'in j. dallanması=G[2]*r
         r=random[0,1] ve reel sayıdır.
      indis←indis+1
      i←i+1
   k←k+1

Fidan Büyümesi: Büyüme aşama üç operatör içerir: Eşleştirme, Dallanma, Aşılama. Eğer bahçe (G) |G| kadar fidan içeriyorsa, mevcut bahçe, mevcut bahçe ve yavru bahçesinden en iyi |G| kadar fidan ile değiştirilir. Yani, yeni nesil bahçeyi seçmek için stokastik bir süreç yok.

Eşleşme: Eşleşme operatörünün amacı, geçici çözümler (Fidan) arasında var olan bilgileri değiştirerek var olan fidanlardan yeni bir fidan üretmektir (Algoritma 2). $S_1=\{s_{11}, s_{12}, ..., s_{1i}, ..., s_{1n}\}$ ve $S_2 = \{s_{21}, s_{22}, ..., s_{2i}, ..., s_{2n}\}$ iki fidan olsun; S_1 ve S_2 arasındaki mesafe, gerçekleşen eşleştirme sürecini etkiler veya etkilemez ve mevcut çift arasındaki mesafeye bağlıdır. Pm (S_1, S_2), S_1 ve S_2 fidanlarının eşleştirme olasılığı olup, lineer veya üstel olabilir.

Algoritma 2. Eşleşme(S1, S2)

1. j←1, …, n
2. Hesapla Pm(S1,S2) =1- $\frac{(\sum_{j=1}^{n}(s_{1,j}-s_{2,j})^2)^{1/2}}{R}$
3. i←1, …, n
4. **if** Pm(S1,S2)≥random(0,1) **then**
   S1←S1-s1,j ve S2←S2-s2,i, S1←S1+s2,j ve S2←S2+s1,i

Dallanma: S_1 = s_{11}, s_{12}, ..., s_{1i}, ..., s_{1n} bir fidan olsun. S_{1i} noktasında bir dal oluştuğunda, s_{1j} noktasında meydana gelen bir dal olasılığı i != j doğrusal ve doğrusal olmayan iki şekilde hesaplanabilir. S_{1i} ve s_{1j} arasındaki uzaklık, | j-i | veya | i-j |.

Algoritma 3. Dallanma(S1, S2)

1. i←1, …, n
2. j←i+1, …, n
3. **if** dallanma yoksa P($s_{1,j}|s_{1,i}$)=1
   ve dallanma işlemi uygulanır.
4. **else**
   P($s_{1,j}|s_{1,i}$)= $1-\frac{1}{(|j-i|)^2}$, i≠j
   Veya
   P($s_{1,j}|s_{1,i}$)= $1-\frac{1}{e^{(|j-i|)^2}}$
5. **if** P($s_{1,j}|s_{1,i}$)≥random(0,1) **then** s1,j bir dallanma olacak.

Aşılama: Aşılama işlemi, bu algoritmada fidanların birbirine benzememesi durumunda iki farklı fidan arasında gerçekleşir. Fidanların farklılığı, aşılama işleminin başarısını etkiler ve ayrıca aşılama başarısı, her iki fidanın farklılığı ile orantılıdır.

Algoritma 4. Aşılama(S1, S2)

1. i←1, …, n
2. Dis(S1,S2)=$\sum_{i=1}^{n} s_{1,i} \oplus s_{2i}$
3. **if** Dis(S1,S2)≥random **then**
4. $$S_1' \begin{cases} s_{1i} & \text{if } s_{1i} = s_{2i} \\ \text{random}(1) & \text{if } s_{1i} \neq s_{2i} \end{cases}$$
   Ve
   $$S_2' \begin{cases} s_{2i} & \text{if } s_{2i} = s_{1i} \\ \text{random}(1) & \text{if } s_{2i} \neq s_{1i} \end{cases}$$

Fidan gelişim algoritması, daha iyi fidan üretmek için bahçedeki benzerliği kullanır (eşleştirme süreci). Bu durum, hesaplamalı kültürel etkileşim türüdür. Aşılama işlemi, eşleştirme sürecinin tersidir, çünkü aşılama operatörü bahçede farklılığı kullanıyor. Dolayısıyla, aynı zamanda kültürel etkileşimin de tersidir, çünkü bireylerin benzer düşüncesi kültürel olarak etkileşime girer. Dallanma süreci tek bir operatöre bağlıdır ve yeni çözümleri mevcut çözümler setine yerleştirmeyi ve bilgiyi mevcut çözüm setinden kaldırmayı hedeflemektedir.

Fidan kalitesini belirlemek için, genetik algoritmanın aksine hedef fonksiyonu kullanılır. Hedef fonksiyon, her bir fidenin bir çözüm olarak gördüğü ve soruna bir çözüm olarak uygulanan ve elde edilen sonuç, hedef fonksiyonun değeri olan bir fonksiyondur Fidan gelişim algoritması iki farklı yoldan uygulanabilir: FGA-1 ve FGA -2.

FGA-1 algoritması, operatörleri sıralı olarak uygular. Elde edilen fidanlar geçici çözüm olarak görülür. Eşleştirme, dallanma ve aşılama operatörleri uygulandıktan sonra, elde edilen geçici çözümler değerlendirilir. Yeni nesil, mevcut nesil ve geçici çözümlerden m fidan seçerek elde edilecektir. Şekil 8'de algoritma, FGA -1'in sözde kodudur.

Algoritma 5. FGA-1

1. t←0 // Başlama Zamanı
2. Fidan Ekimi(G(t)) // Başlangıç Popülasyonu
3. Uygunluk Fonksiyonunu hesapla(G(t))
4. **While** Sonlandırma kriteri ile karşılanmadı **do**

4.1. G1(t) ←Eşleşme(G(t))
4.2. G2(t) ←Dallanma(G(t))
4.3. G3(t) ←Aşılama(G(t))
4.4. Uygunluk Fonksiyonunu hesapla (G1(t)∪G2(t)∪G3(t))
4.5. G(t+1) ←Seç(G1(t)∪G2(t)∪G3(t))
4.6. t←t+1

FGA-2 algoritması, operatörleri ayrı ayrı uygular. Önce eşleştirme operatörü uygulanır ve elde edilen fidanlar değerlendirilir. Bir sonraki kısmi nesil Gm (sonraki bahçe), eşleştirme operatörünün uygulanmasından sonra elde edilen mevcut bahçeden ve fidanlardan en iyi fidanları seçerek elde edilir. İkincisi, Gm'ya dallanma operatörü uygulanır ve elde edilen fidanlar değerlendirilir. Bir sonraki kısmi nesil Gb, Gm'den en iyi fidanı ve dallanma uygulanmasıyla elde edilen fidanları seçerek elde edilir. Son olarak, Gb'ye aşılama operatörü uygulanmaktadır. Yine, elde edilen fidanlar değerlendirilir ve en iyi fidanlar deterministik olarak aşılama uygulanarak elde edilen Gb ve fideler arasından seçilir ve sonraki nesil G (t + 1) elde edilir, Aşağıdaki algoritma, FGA -2'nin sözde kodudur.

Algoritma 6. FGA-2

1. t←0 // Başlama Zamanı
2. Fidan Ekimi(G(t)) // Başlangıç Popülasyonu
3. Uygunluk Fonksiyonunu hesapla(G(t))
4. **While** Sonlandırma kriteri ile karşılanmadı **do**

4.1. G1(t) ←Eşleşme(G(t))
4.2. Uygunluk Fonksiyonunu hesapla (G1(t))
4.3. Gm(t) ← Seç(G1(t)∪G(t))
4.4. G2(t) ←Dallanma(Gm(t))
4.5. Uygunluk Fonksiyonunu hesapla (G2(t))
4.6. Gb(t) ← Seç(G2(t)∪ Gm(t))
4.7. G3(t) ←Aşılama(Gb(t))
4.8. Uygunluk Fonksiyonunu hesapla (G3(t))
4.9. G(t+1) ←Seç(G3(t)∪Gb(t))
4.10. t←t+1

Çözüm elde edilinceye kadar FGA süreci bu şekilde devam eder. Genetik algoritmanın aksine hedef fonksiyonunun kullanılması, uygunluk fonksiyonunun tanımlanmasının zorluğundan kaynaklanmaktadır [7].

2.3 Lig Şampiyonluk Algoritması

Birkaç cümle ile LCA'daki takım oyunları terminolojisinden bahsedelim. Bir spor ligi belirli bir alanda rekabeti sağlamak adına var olan ve takımlara göre düzenlenmiş bir organizasyondur. Lig, bireysel sporlara değil de takım sporlarına ait bir kavramdır. Takımlar, rakip takımlar ile belirli sayıda karşılaşma yapabilmektedir ve lig şampiyonluğuna belirli şekillerde itiraz edilebilmektedir. Müsabakalar zincirinde kazan-kaybet-eşitlik sistemi ve çeşitli kriterleri karşılayan ekipler için bonus puanların elde edilmesine göre en iyi skorlara sahip olan takım şampiyonluğa ulaşır.

Genel olarak her takım, önceden belirlenmiş taktiklerine göre mücadele edebilmesi için oyuncuları sahada konumlandırmanın metodu olarak nitelendirilebilen ve oyun sırasında ortaya çıkan bir

oluşuma sahiptir. Örneğin, futbolda kullanılan en sık oluşumlar 4-4-2, 4-3-3, 3-2-3-2, 5-3-2 ve 4-5-1 arasındaki varyasyonlardır [6]. Çoğunlukla her takımın antrenörünün kullanabileceği oyuncu tipi ile ilişkili iyi bir oluşuma sahip olması gerekir. Spor takımlarının her karşılaşmaya uygun oluşumlarını ve oyun planlarını tasarlaması oldukça önemlidir. Her karşılaşmanın sonrasında oyunlarını geliştirmek, zayıf noktalarını iyileştirmek, bir sonraki rakiplerinin oyunlarını gözlemlemek veya kendi güçlerini arttırmak amacı ile antrenörler tarafından geleceklerini planlamak için analizler gerçekleştirilir. Sözü edilen analizler ekibin benzersiz analizleri ile ortaya çıkan fırsat ve tehditlerin ortaya konulmasıdır. Analiz çalışmasında güçlü ve zayıf yönlerin (strengths/weaknesses) oluşturduğu iç faktörlere ilaveten fırsat ve tehditlerin (opportunities/threats) oluşturduğu dış faktörler incelenir. İç ve dış faktörlerin incelendiği çalışma SWOT (strengths/weaknesses/ opportunities/threats) analizi olarak adlandırılmaktadır. SWOT analizi ile hedeflenen konum ile şu an bulunan konum arasındaki farkın incelendiği bir yaklaşım elde edilmiş olur. Bu farkın elde edilme süreci, mevcut duruma ilişkin detaylı bir analizin elde edilmesi ile daha iyi bir adım için temel oluşturulmuş demektir. Hedeflenen konum ile şu an bulunan konum arasındaki farkın incelenmesi süreci sorunlu alanların belirlenmesinden önerilen çözümlere geçilmemesini sağlamak için kullanılmaktadır. Algoritmaya, L-1 haftadan oluşan her mevsim için S × (L-1) haftalık yarışmalara neden olacak belirli sayıda "mevsim" (S) ile son verilmektedir [6].

2.3.1 Lig Programı Oluşturma

Bir spor ligi şampiyonasını taklit eden LCA' da maçlara ilişkin bir takvim oluşturulmaktadır ve sözü edilen takvimde çevrimsel olarak her takımın mevcut sezonda diğer tüm takımlarla karşılaşması esasına dayanmaktadır. L boyutunda bir lig için, (L-1) turda (hafta) ve her turda (L/2) karşılaşma eşzamanlı olarak gerçekleşeceğinden (L tek ise, (L-1) / 2 eşleşen L hafta olacak ve her hafta bir takım karşılaşma yapmayacak) çevrimsel turnuvasının bir diagrafta bir kenar boyama problemi olarak modellenebilmektedir. LCA akış diyagramı Şekil 10'da detaylıca şematize edilmiştir [6].

(T+1). haftada bir takım oluşturmak için aşağıda verilen denklemlerden biri kullanılabilmektedir;

Eğer her iki takım maçlarını kazanmışsa;

$$X_U = X_b + Y\left(F_1 r_1 (X_c - X_d)\right) + F_1 r_2 (X_e - X_f)$$

Eğer birinci takım kazanmış, ikinci takım kaybetmişse;

$$X_U = X_b + Y\left(F_2 r_1 (X_c - X_d)\right) + F_1 r_2 (X_e - X_f)$$

Eğer birinci takım kaybetmiş, ikinci takım kazanmışsa;

$$X_U = X_b + Y\left(F_1 r_1 (X_c - X_d)\right) + F_2 r_2 (X_e - X_f)$$

Eğer her iki takım da kaybetmişse;

$$X_U = X_b + Y\left(F_2 r_1 (X_c - X_d)\right) + F_2 r_2 (X_e - X_f)$$

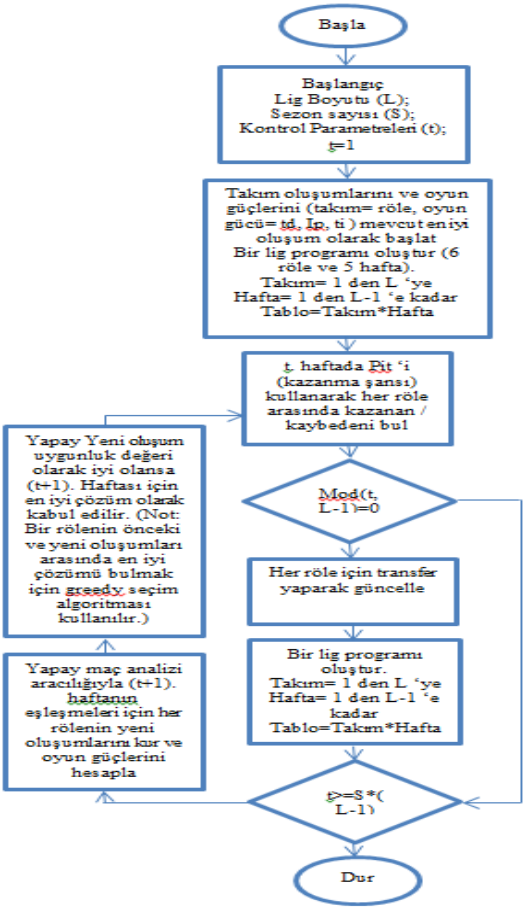
Burada F_1 ve F_2 ölçeklendirme katsayılarıdır ve r_1 ve r_2 rasgele sayılardır. Y ise bir ikili sayıdır [6].

2.3.2 LCA Sözde kodu ve Akış diyagramı

Lig Şampiyonluk Algortiması (LCA)

1. Lig boyutu (L)ve Sezon sayısını (S) başlat; t = 1;
2. Bir lig programı oluştur (6 röle ve 5 hafta);
3. Takım oluşumları ve oyun güçleri (takım = röle, oyun gücü = td, Ip, ti) mevcut en iyi oluşum olsun;
4. While t =< S ∗ (L − 1)
5. t. haftada Pit 'i (kazanma şansı) kullanarak her röle arasında kazanan/kaybedeni bul;
6. t = t + 1;

7. For i = 1 to L
8. Yapay maç analizi aracılığıyla (t + 1). haftanın eşleşmeleri için her rölenin yeni oluşumlarını kur ve oyun güçlerini hesapla;
9. Yapay Yeni oluşum uygunluk değeri olarak iyi olansa (t + 1). haftası için en iyi çözüm olarak kabul edilir. (Not: Bir rölenin önceki ve yeni oluşumları arasında en iyi çözümü bulmak için greedy seçim algoritması kullanılır.);
10. End for
11. If mod(t, L − 1) = 0
12. Bir lig programı oluştur;
13. End if
14. End while



Şekil 4. LCA Akış Diyagramı [6].

3 Problemin Formülasyonu

Fidan Gelişim Algoritması ile ters zamanlı yönsüz aşırı akım röle koordinasyonu için kullanılan amaç fonksiyonu, 1 nolu denklemde tanımlanmıştır. n; toplam röle sayısı iken t_i; her bir rölenin çalışma zamanıdır. Bu çalışmada ana hedef aşırı akım rölelerinin çalışma zamanlarını minimize etmektir. Bu çalışmada tüm röleler aynı kabul edilerek karakteristik optimizasyon yapılmıştır.

$$\text{Amaç Fonksiyonu} = \min \sum_{i=1}^{n} t_i \quad (1)$$

$$t_i = \frac{0,14.t_d}{(\frac{I_{kd}}{I_p})^{0,02}-1} \quad (2)$$

Burada I_{kd}= kısa devre akım değeridir bu değer sabittir. I_p değeri başlatma akımı değeridir, t_i=röle çalışma zamanı, t_d=çalışma eğrisinin değerleridir. Optimizasyon için kısıtlamalar TEİAŞ sisteminde kullanılan değerler baz alınarak bu çalışmada öngörülen bir aralıkta aşağıdaki gibi ifade edilebilmektedir;

I_{kd} değeri sabit değerdir. Bu değer röle 1 için I_{kd}=1263A, röle2,3,4,5,6 için=5639A.

1.kısıt $t_{imin} \leq t_i \leq t_{imax} = 1 \text{ sn} \leq t_i \leq 2,2 \text{ sn}$

1.kısıt için $t_6,t_5,t_4,t_3 \leq t_2 \leq t_1$ bağıntısı vardır. Bunun anlamı röle 1'in zamanı olan t_1, röle 2 zamanı olan t_2'den fazla olacaktır. Röle 2 zamanı olan t_2 ise diğer rölelerin zamanından fazla olacaktır. Ama röle 3,röle 4, röle 5, röle 6 zamanları kendi aralarında bir sıralamaya tabi değildir. Sadece bu röle değerleri t_2 ve t_1 'den küçük olacaktır. Ayıca t_1 t_2'den 0,3 sn, t_2 de diğer röle zamanlarından 0,3 sn fazla olacak şekilde ayarlanmalıdır.

2.kısıt $t_{dmin} \leq t_d \leq t_{dmax}$ $(0,2 \leq t_d \leq 1)$

3. kısıt bu kısıt başlama zamanları içindir yani I_p için. Bu değer her rölede farklı bir sınıra sahiptir,

Röle 1 için I_p= 117,152 ≤ I_p ≤ 128,65

Röle 2 için I_p = 523 ≤ I_p ≤ 575

Röle 3 için I_p = 400 ≤ I_p ≤ 420

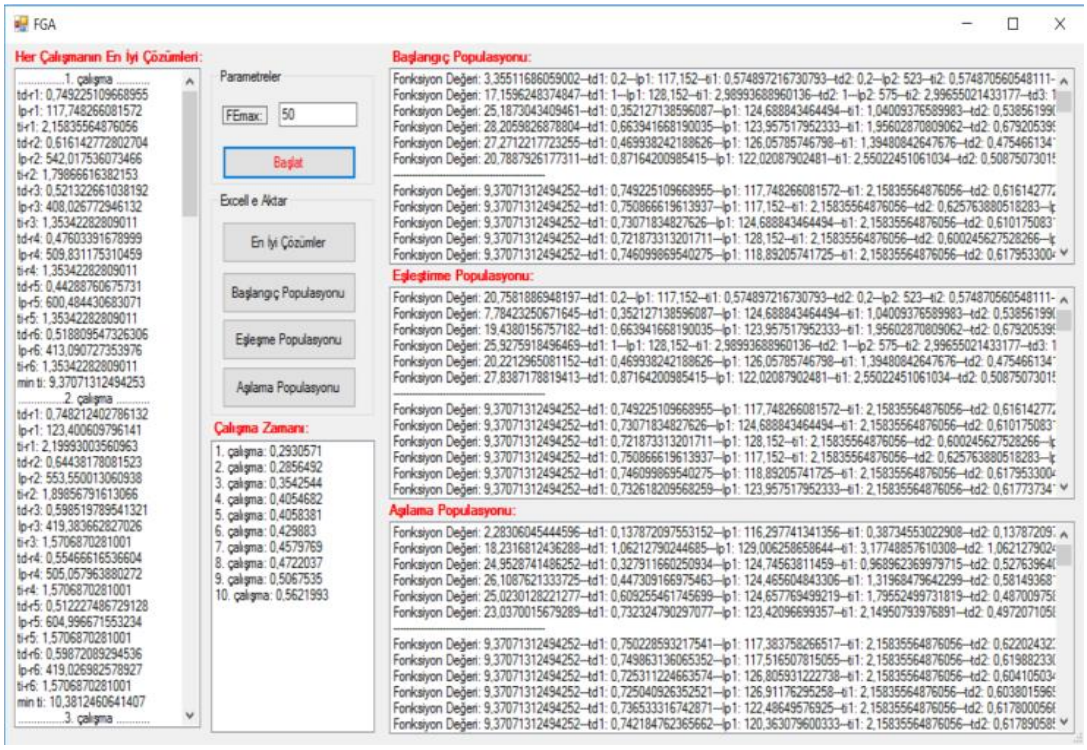
Röle 4 için I_p = 500 ≤ I_p ≤ 510

Röle 5 için I_p = 600 ≤ I_p ≤ 610

Röle 6 için I_p = 400 ≤ I_p ≤ 420

Sonuç olarak bu kısıt değerlerine göre t_d ve ti ve I_p değerleri hesaplanacaktır [6].

4 Yapılan Uygulamalar ve Sayısal Sonuçları



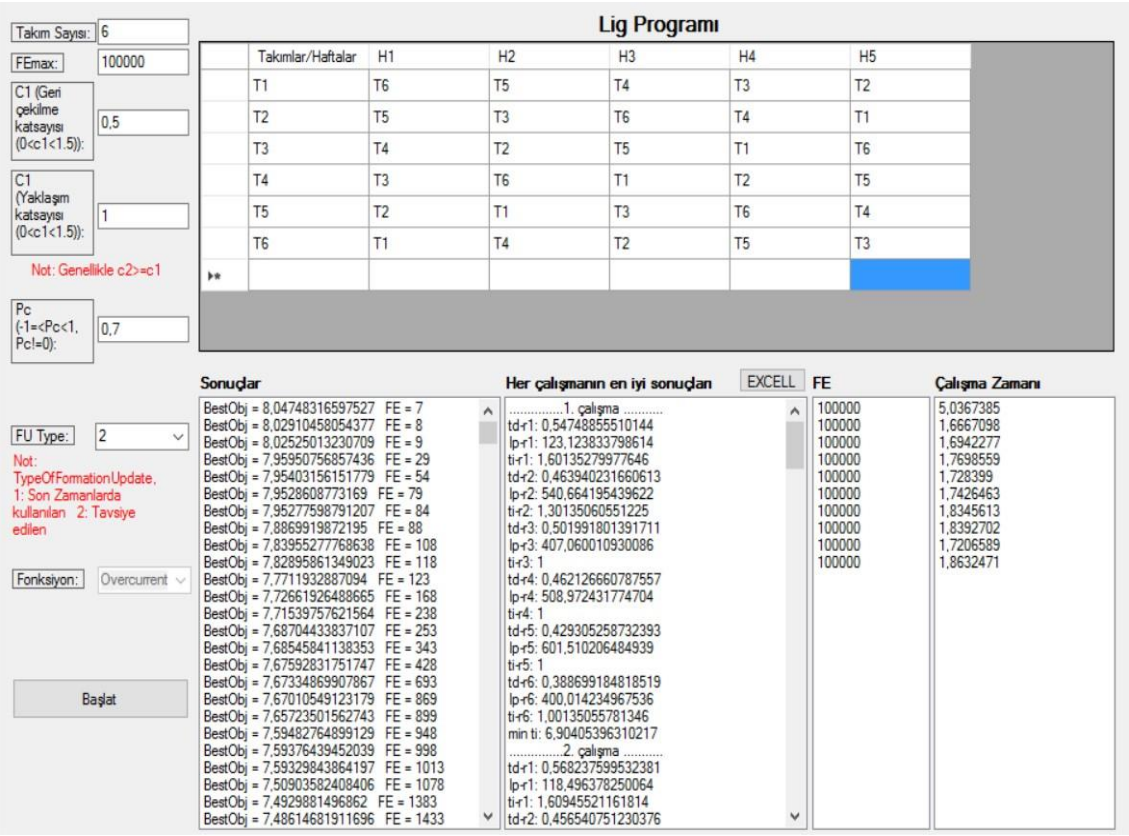
Şekil 5. FGA'nın ekran görüntüsü.

Şekil 5'teki FGA uygulama görüntüsünde her çalışmanın en iyi çözümleri, parametreler, çalışma zamanı, başlangıç popülasyonu, eşleştirme popülasyonu, aşılama popülasyonu ve excell'e aktar kısımları mevcuttur.

Parametreler kısmında fonksiyon iterasyon sayısı verilmektedir ve 10 çalışma olarak dizayn edilmiştir. Tablo 1'de FGA ile elde edilen sonuçlar gösterilmiştir.

Tablo 1. FGA Sonuçları.

	t_d	I_p(A)	T_i(sn)
Röle-1	0,557619	119,673	1,617
Röle-2	0,450755	538,247	1,311
Röle-3	0,386524	410,620	1,005
Röle-4	0,355225	505,108	1,005
Röle-5	0,327050	609,073	1,005
Röle-6	0,385509	413,380	1,005



Şekil 6. LCA'nın ekran görüntüsü.

Şekil 6'daki LCA uygulama görüntüsünde her çalışmanın en iyi çözümleri, parametreler, çalışma zamanı, başlangıç popülasyonu, eşleştirme popülasyonu, aşılama popülasyonu ve excell'e aktar kısımlarından oluşmaktadır.

Parametreler kısmında fonksiyon iterasyon sayısı verilmektedir ve 10 çalışma olarak dizayn edilmiştir. Tablo 2'de LCA ile elde edilen sonuçlar gösterilmiştir.

Tablo 2. LCA Sonuçları.

	t_d	I_p(A)	T_i(sn)
Röle-1	0, 56928	117,741	1,605
Röle-2	0,45687	524,616	1,305
Röle-3	0,49254	406,211	1,001
Röle-4	0,455225	501,01	1,0015
Röle-5	0,419473	600,432	1
Röle-6	0,390291	400,003	1,005

Örnek bir trafo merkezinin röle koordinasyon problemi için C# programlama dili ile yapılan programda hesaplanan sonuçlar bu bölümde tablolar halinde gösterilmiştir. Bu iki yöntem sonucunda amaç fonksiyonumuzun alabileceği minimum t_i değerleri tablo 3'te gösterilmiştir.

Tablo 3. Yöntemlerin sonuçlarının karşılaştırılması.

Yöntem	toplam t_i(sn)
FGA	6,948
LCA	6,9175

İki yöntemin sonucu da verilen kısıtlarda alt sınırlara çok yakın çıkmıştır. Bu sonuçlar yöntemlerin başarılı olduğunu ortaya koymaktadır. İki algoritma yöntemi arasındaki fark; FGA'nın her çalışması 50 iterasyon ile bu sonuçları vermektedir, LCA'da ise her çalışma 100000 iterasyon ile bu sonuçlara ulaşmaktadır. Fakat iki yöntemde her çalışmanın çalışma zamanı yaklaşık olarak aynıdır.

5 Sonuç ve Gelecek Çalışmalar

Fidan Gelişim Algoritması, hem yerel hem de genel arama adımlarını içerir. Herhangi bir ilave fonksiyon kullanmamaktadır. Parametre olarak çok az girdi almıştır ve oldukça başarılı sonuçlar vermiştir. Başarının en büyük sebebi; başlangıç popülasyonunun rasgele olarak seçilmemesidir.

Lig Şampiyonluk Algoritması'nın başarısı ise çok kısa sürede çok yüksek boyuttaki iterasyon sayısına ulaşabilmesidir.

Sonraki adımlarda başka optimizasyon algoritmalarının çalışmaya dahil edilerek daha düşük iterasyon ve optimal değerlere ulaşılmaya çalışılması amaçlar arasında yer almaktadır. Bu bağlamda aşırı-akım röle koordinasyon problemini Diferansiyel Gelişim Algoritması'ndan faydalanarak çözmek, elde edilen çıktı değerlerinin bu çalışmaya konu olan FGA ve LCA algoritmaları ile elde edilen değerler ile karşılaştırmak gelecek hedefler arasındadır.

Ayrıca çalışma kapsamında elde edilen yazılım ürününün ileri seviyelere taşınarak bir Graphical Unit Interface'e (GUI) dönüştürülmesi de gerçekleştirilmesi planlanan çalışmalar arasındadır.

Kaynaklar

[1] Demir M, Karcı A. "Veri kümelemede fidan gelişim algoritmasının kullanılması", Elektrik Elektronik Bilgisayar Biyomedikal Mühendisliği 12. Ulusal Kongresi ve Sergisi, Eskişehir 2007.

[2] Erdoğan M, Alataş B, Karcı A. "Uniform Population and Uniform Operator in Genetic Algorithms". Afyon Kocatepe Üniversitesi Fen Ve Mühendislik Bilimleri Dergisi 3 (2015): 11-26 http://dergipark.gov.tr/akufemubid/issue/1595/19817

[3] Öztemel E. Yapay Sinir Ağları. Papatya Yayıncılık, 2003

[4] Haykin S. “Second Edition Neural Networks A Comprehensive Foundation”, Simon&Schuster/A Viacom Company, 1999.
[5] Akyol S, Alatas B. “Plant intelligence based metaheuristic optimization algorithms”, Artif Intell Rev (2017) 47: 417. https://doi.org/10.1007/s10462-016-9486-6
[6] Seyyarer A, Akdağ O, Hark C, Karci A, Yeroğlu C. (2017, September). “Overcurrent relay coordination of 154/34, 5 kV Hasançelebi substation by league championship algorithm”. In Artificial Intelligence and Data Processing Symposium (IDAP), 2017 International (pp. 1-6). IEEE.
[7] Karci A. (2007b) “Natural inspired computational intelligence method: saplings growing up algorithm”. In:IEEE international conference on computational cybernetics. ICCC2007.19–21Oct.pp221–226.doi:10.1109/ ICCCYB.2007.4402038.
[8] Lenin K., Reddy B. R., & Kalavathi M. S. (2013). “League championship algorithm (LCA) for solving optimal reactive power dispatch problem”. International Journal of Computer and Information Technologies, 1(3), 254-272.
[9] Abdulhamid S. M., Latiff M. S. A., Idris I. (2015). “Tasks scheduling technique using league championship algorithm for makespan minimization in IAAS cloud”. arXiv preprint arXiv:1510.03173.
[10] Demir M, Karcı A, Özdemir M. (2014). “Fidan Gelişim Algoritması Yardımı ile DNA Motiflerinin Keşfi”. Cankaya University Journal of Science and Engineering, 8 (1), 115-128. Retrieved from http://dergipark.gov.tr/cankujse/issue/4035/53233