T.C.
ERCİYES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

WEB TABANLI YAPAY ARI KOLONİ PROGRAMLAMA YAZILIMININ GELİŞTİRİLMESİ

Hazırlayan
Ceylan BOZOĞULLARINDAN

Danışman
Doç. Dr. Celal ÖZTÜRK

Yüksek Lisans Tezi

Aralık 2018
KAYSERİ

T.C.
ERCİYES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

WEB TABANLI YAPAY ARI KOLONİ PROGRAMLAMA YAZILIMININ GELİŞTİRİLMESİ

Hazırlayan
Ceylan BOZOĞULLARINDAN

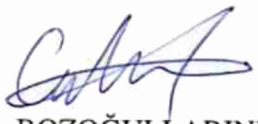
Danışman
Doç. Dr. Celal ÖZTÜRK

Yüksek Lisans Tezi

Aralık 2018
KAYSERİ

BİLİMSEL ETİĞE UYGUNLUK

Bu çalışmadaki tüm bilgilerin, akademik ve etik kurallara uygun bir şekilde elde edildiğini beyan ederim. Aynı zamanda bu kural ve davranışların gerektirdiği gibi, bu çalışmanın özünde olmayan tüm materyal ve sonuçları tam olarak aktardığımı ve referans gösterdiğimi belirtirim.



Ceylan BOZOĞULLARINDAN

"Web Tabanlı Yapay Arı Koloni Programlama Yazılımının Geliştirilmesi" adlı Yüksek Lisans tezi, Erciyes Üniversitesi Lisansüstü Tez Önerisi ve Tez Yazma Yönergesi'ne uygun olarak hazırlanmıştır.

Hazırlayan
Ceylan BOZOĞULLARINDAN

Danışman
Doç. Dr. Celal ÖZTÜRK

Bilgisayar Mühendisliği ABD Başkanı
Prof. Dr. Veysel ASLANTAŞ

Doç. Dr. Celal ÖZTÜRK danışmanlığında **Ceylan BOZOĞULLARINDAN** tarafından hazırlanan **"Web Tabanlı Yapay Arı Koloni Programlama Yazılımının Geliştirilmesi"** adlı bu çalışma, jürimiz tarafından Erciyes Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalında **Yüksek Lisans** tezi olarak kabul edilmiştir.

07 / 01 / 2019

JÜRİ :

Danışman : Doç. Dr. Celal ÖZTÜRK

Üye : Dr. Öğr. Üyesi Beyza GÖRKEMLİ

Üye : Dr. Öğr. Üyesi Fırat İSMAİLOĞLU

ONAY :

Bu tezin kabülü Enstitü Yönetim Kurulunun 03/01/2019 tarih ve ..2019/02-11.. sayılı kararı ile onaylanmıştır.

...08.../...01.../...2019...

Prof. Dr. Mehmet AKKURT

Enstitü Müdürü

ÖNSÖZ / TEŞEKKÜR

Bu çalışmanın yürütülmesinde ve değerlendirilmesinde yardımlarını esirgemeyen değerli danışman hocam sayın Doç. Dr. Celal ÖZTÜRK'e ve yüksek lisans eğitimimde "Otomatik Programlama" dersini aldığım değerli hocam sayın Dr. Öğr. Üyesi Beyza GÖRKEMLİ'ye teşekkür ederim.

Ayrıca bana çalışmalarım süresince her türlü yardımı ve fedakarlığı sağlayan çok değerli eşim Elif BOZOĞULLARINDAN'a teşekkür ederim.

Ceylan BOZOĞULLARINDAN

Aralık 2018, KAYSERİ

WEB TABANLI YAPAY ARI KOLONİ PROGRAMLAMA YAZILIMININ GELİŞTİRİLMESİ

Ceylan BOZOĞULLARINDAN

Erciyes Üniversitesi, Fen Bilimleri Enstitüsü
Yüksek Lisans Tezi, Aralık 2018
Danışman : Doç. Dr. Celal ÖZTÜRK

ÖZET

Bir problemin, çözümüne dair herhangi bir bilgi olmadan, bilgisayarlar tarafından otomatik olarak çözülebilmesi için Evrimsel Hesaplama yaklaşımına sahip çeşitli Otomatik Programlama yöntemleri geliştirilmiştir. Yapay Arı Koloni Programlama (ABCP) yakın zamanda geliştirilmiş bu yöntemlerden biridir. Evrimsel Hesaplama yaklaşımına sahip diğer algoritmalar gibi ABCP de farklı disiplinlerden araştırmacılar tarafından çok çeşitli problemlerin çözümünde tercih edilmektedir. Bu tez çalışmasında, ABCP'nin herhangi bir programlama bilgisine gerek duyulmadan kullanılmasını sağlayan, kullanıcı dostu arayüzlere sahip, bulut tabanlı bir web uygulaması (ABCPWeb) geliştirilmiştir.

Uygulamanın doğruluğunu test etmek için ilk olarak ABCP 'nin önceden uyarlandığı kıyaslama problemleri ile çalışılmıştır. Elde edilen sonuçların doğrulanmasının ardından ABCPWeb, ABCP algoritmasının daha önce hiç uygulanmadığı farklı kıyaslama problemleri ile test edilmiştir. Sonuçlar, literatürde yer alan Genetik Programlama yöntemiyle elde edilmiş sonuçlarla karşılaştırılarak ABCP'in başarısı ortaya konmuştur.

Anahtar Kelimeler: Yapay Arı Koloni Programlama (ABCP), Evrimsel Hesaplama, Otomatik Programlama, Bulut Tabanlı Web Yazılım

DEVELOPING WEB BASED ARTIFICIAL BEE COLONY PROGRAMMING SOFTWARE

Ceylan BOZOĞULLARINDAN

Erciyes University, Graduate School of Natural and Applied Sciences
Master Thesis, December 2018
Supervisor: Assoc. Prof. Dr. Celal ÖZTÜRK

ABSTRACT

To solve a problem without any knowledge of its solution, a variety of Automatic Programming methods based on Evolutionary Computation approach have been developed. Artficial Bee Colony Programming (ABCP) is one of the recently proposed automatic programming methods. Like the Evolutionary Computation approach based other algorithms, ABCP has been preferred and applied to various problems by researchers from different disciplines. In this study, a cloud-based web application (ABCPWeb) is developed which allows to use the ABCP with no need to have any programming knowledge.

ABCPWeb is firstly tested on the benchmark problems that are previously solved by ABCP. After verifying the obtained results, it is tested with various benchmark problems that ABCP have never been applied before. The success of ABCP is revealed by comparing the obtaining results with the results of Genetic Programming.

Keywords: Artificial Bee Colony Programming (ABCP), Evolutionary Computation, Automatic Programming, Cloud Based Web Application

İÇİNDEKİLER

WEB TABANLI YAPAY ARI KOLONİ PROGRAMLAMA YAZILIMININ GELİŞTİRİLMESİ

2. BÖLÜM
YAPAY ARI KOLONİ PROGRAMLAMA

3. BÖLÜM
ABCP WEB YAZILIMI

4. BÖLÜM
TARTIŞMA, SONUÇ ve ÖNERİLER

KISALTMALAR

ABC	:	Yapay Arı Koloni algoritması
ABCP	:	Yapay Arı Koloni Programlama
GA	:	Genetik Algoritma
GP	:	Genetik Programlama
EP	:	Evrimsel Programlama
DE	:	Diferansiyel Gelişim Algoritması
ES	:	Evrimsel Strateji
TSP	:	Gezgin Satıcı Problemi
EA	:	Evrimsel Algoritmalar
EC	:	Evrimsel Hesaplama
OCR	:	Optik Karakter Tanıma
AP	:	Otomatik Programlama
AI	:	Yapay Zeka
ML	:	Makine Öğrenmesi
PSO	:	Parçacık Sürü Optimizasyonu
SVM	:	Karar Destek Makinaları
PGM	:	Olasılıksal Grafik Modelleri
NN	:	Sinir ağları
ACP	:	Karınca Koloni Optimizasyonu
IP	:	Tümevarımlı Programlama

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

GİRİŞ

Yapay Arı Koloni Programlama (Artificial Bee Colony Programming - ABCP), son zamanlarda sıkça kullanılan evrimsel hesaplama yöntemlerinden biridir [1]. Arıların besin kaynağı arayışından esinlenilerek, 2005 yılında Karaboğa tarafından geliştirilen Yapay Arı Koloni (ABC) algoritmasını temel alır. Çeşitli problemlere uygulanarak başarısı kanıtlanmıştır [1]. ABCP, sadece bilgisayar bilimi ile ilgilenen araştırmacıların değil, diğer disiplinlerdeki bazı araştırmacıların da ilgisini çekmektedir. Çünkü ABCP, diğer disiplinlerde mevcut olan bazı problemlere de çözüm üretebilecek niteliğe sahiptir.

Bir araştırmacının ABCP 'yi kullanabilmesi için iki seçeneği vardır. Bunlardan ilki ABCP 'yi sıfırdan kodlayarak oluşturmak, diğeri ise daha önce oluşturulmuş bir yazılımı kullanmaktır. Literatürde ABCP algoritmasının kaynak kodu veya ABCP kullanımı için oluşturulmuş bir yazılım bulunmadığından, araştırmacı, ABCP algoritmasını anlayarak sıfırdan bilgisayar ortamında oluşturmak zorundadır. Bu durum, programlama bilgisi olmayan araştırmacıların işini oldukça zorlaştırmaktadır. Programlama bilgisi olan araştırmacılar için ise zaman kaybına neden olmaktadır.

Bu çalışmada, üst paragrafta anlatılan problemi ortadan kaldırmak için, ABCP'nin kolayca kullanılmasına olanak tanıyan bulut tabanlı bir web uygulaması (ABCPWeb) geliştirilmiştir. Bu uygulama ile birlikte araştırmacılar, kullanıcı dostu arayüzler aracılığıyla probleme ilişkin giriş değerlerini belirleyip, çıktıları kolayca alabilirler. ABCPWeb üzerinde gerçekleştirilen her deney tüm detayları ile bulutta saklanır. Bu durum, araştırmacıya, çalıştırılan deneylere ait bilgilerinin olduğu deney arşivine, istediği zaman erişim imkanı sağlar. Deney arşivi ile birlikte çalışmalar daha sistematik ve düzenli yürütülebilir. Uygulama internet tabanlı olduğu için uygulamaya internet bağlantısı olan her yerden erişilebilir. ABCPWeb uygulamasındaki istemci tarafından başlatılan deneyler, bulut sistemde yer alan sunucuda devam eder. ABCPWeb'in bu özelliği ile araştırmacı, zamandan ve kaynaklarından tasarruf eder.

ABCP'nin evrimsel hesaplama yöntemlerinden biri olması nedeniyle ve evrimsel hesaplama yöntemlerinin farklı disiplinler tarafından kullanım durumunu göstermek amacıyla tezin birinci bölümünde, "Evrimsel Hesaplama" ve uygulamaları anlatılmıştır. Literatürde evrimsel hesaplama yöntemleri ile çözüm üretilen problemler, 5 farklı kategoride incelenmiştir: planlama, tasarım, simulasyon, kontrol ve sınıflandırma. Bu bölümde, belirtilen kategorilerde yer alan çeşitli problemler için, evrimsel hesaplama yöntemleri kullanılarak üretilen çözüm örnekleri anlatılmıştır. Ayrıca, tezin konusu olan ABC ve ABCP 'ye olan benzerliklerinden dolayı, evrimsel hesaplama yöntemlerinden GA ve GP 'nin anlatımı da bu bölümde alt başlık olarak yer almaktadır.

Tezin ikinci bölümünde ABCP ve ABC algoritmaları anlatılmıştır. ABCP 'nin temelini oluşturması ve anlatımını kolaylaştırması nedeniyle, bu bölümün ilk kısmında ABC algoritması yer almaktadır. ABC algoritmasının anlatımından sonra ikinci kısımda ABCP algoritmasına yer verilmiştir. ABCP anlatılırken, tekrarı önlemek amacıyla GP ile benzer olan yanları için, birinci bölüm referans gösterilmiştir.

Tezin üçüncü bölümünde, ABCPWeb yazılımı anlatılmıştır. Bu bölümde, kullanılan teknolojilere kısaca değinilmiştir. Kullanıcının ABCPWeb yazılımını kullanırken izleyeceği kullanım durumları alt başlıklar halinde anlatılmıştır: Üye girişi, veri yükleme, çalıştırma ve sonuçları alma. Bu durumlar anlatılırken her durum için tasarlanan arayüzlerin görsellerine de ilgili oldukları başlıklar altında yer verilmiştir. Bölümün son kısmında ise gerçekleştirilen iki farklı deney anlatılmıştır. Bunlardan ilki ABCPWeb yazılımının doğruluğunu test etmek için gerçekleştirilmiştir. Diğer deneyde ise, ABCP, ABCPWeb kullanılarak yeni problemlere uygulanmış, sonuçlar benzer yöntemlerden biri olan GP sonuçları ile kıyaslanmıştır.

Tezin son bölümünde ise, sonuç ve öneriler yer almaktadır. Bu bölümü, ABCPWeb yazılımı için ilerde planlanan geliştirmeler ve çalışma ile ilgili genel yorumlar oluşturmaktadır.

1. BÖLÜM

EVRİMSEL HESAPLAMA

1.1. Giriş

Evrimsel hesaplama, evrimin simulasyonu için farklı teknikleri kullanan araştırmacıları bir araya getiren, esnek hesaplamanın alt dallarından biridir ve yapay zeka alanının da bir parçası olarak kabul edilir [2]. Evrimsel programlama (EP), genetik algoritma (GA), genetik programlama (GP), evrimsel strateji (ES), sürü zekasına dayalı algoritmalar ve benzeri yapay zeka tekniklerinin tümü, "Evrimsel Hesaplama" çatısı altında toplanır. Bu tekniklerde ortak olan en temel özellik, birlikte bir popülasyon oluşturan aday çözümlerin, iteratif olarak elde edilmesidir.

Evrimsel hesaplama tekniklerinin, bir problemi çözerken cevaplandırması gereken bir takım ortak temel sorular vardır. Bu sorular aşağıda liste halinde verilmiştir:

1. Algoritmada problem nasıl temsil edilir? Bireylerin gösterim biçimi nasıldır?
2. Başlangıç popülasyonunu oluşturma yöntemi nedir?
3. Bireylerin uygunluk değerlendirmesi nasıl yapılır?
4. Kullanılan genetik operatörler nelerdir ve hangi sırayla kullanılır?
5. Kontrol parametreleri nelerdir ve nerelerde kullanılırlar?

Evrimsel hesaplama mekanizmalarında, evrim süreçlerinden çaprazlama, rastsal değişimler, yarışma ve seçilim vardır. Bu dört sürecin bir arada doğada veya bilgisayar ortamında gerçekleşmesi, evrimi kaçınılmaz kılar [3]. Evrimin simülasyonu için en az bu dört sürecin gerçekleşmesi gerekmektedir.

Şekil 1.1. Evrimsel algoritmalarda bir çevrim

Evrimsel algoritmalar, mutasyon, çaprazlama, doğal seleksiyon ve en iyinin hayatta kalma vb. doğal süreçlerin, iteratif olarak kullanılmasıyla aday çözümlerin oluşturulmasını sağlayan popülasyon tabanlı sezgisel optimizasyon algoritmalarıdır [4]. Evrimsel algoritmalarda, araştırmaya tek bireyden/çözümden değil bireylerin oluşturduğu popülasyondan başlanır ve iteratif olarak çözümler iyileştirilmeye çalışılır. Evrimsel algoritmalar Şekil 1.1'de gösterilen prensiplere göre çalışır.

Bu bölümün ikinci kısmında bazı evrimsel hesaplama problemleri ve uygulama alanları anlatılmıştır. Evrimsel hesaplama yöntemlerinden, GA 'ya üçüncü kısımda, GP 'ye ise dördüncü kısımda yer verilmiştir.

1.2. Evrimsel Hesaplama Uygulamaları

Geçmişten bugüne evrimsel hesaplama yöntemleri, çok fazla alanda çeşitli problemlere uygulanarak başarılı sonuçlar elde edilmiştir. Daha öncesinde bilgisayarla çözümü denenmemiş problemlere de uygulanan evrimsel hesaplama yöntemlerinin gelecekte de uygulama alanlarının artacağı öngörülmektedir.

Bu başlık altında evrimsel hesaplama yöntemleri ile çözüm geliştirilen gerçek hayat problemlerinden bazılarına kısaca değinilmiştir. Evrimsel hesaplama uygulamaları aşağıda gösterilen beş kategoride incelenecektir. Bu kategorilerde yer alan uygulamalar diğer kategorilere de dahil edilebileceğinden, kategori tanımlamaları mutlak değildir.

1. Planlama

2. Tasarım
3. Simülasyon ve tanımlama
4. Kontrol
5. Sınıflandırma

1.2.1. Planlama

Planlama kategorisindeki problemlerin temelinde, bir işin veya bir ürünün yapılması için uyulması gereken düzeni optimize etme gayesi vardır. Bu problemler ve üretilen EC çözümlerinden bazıları bu bölümde kısaca anlatılmıştır.

En çok bilinen kombinasyonel planlama problemlerden biri gezgin satıcı problemidir (TSP) [5–7]. Bu problemdeki senaryoda, bir satıcı birçok şehri en kısa sürede ziyaret edip tekrar evine dönmeyi amaçlar. Satıcının bu amacını gerçekleştirebileceği optimum bir seyehat planına ihtiyacı vardır. Bu plan, yani gideceği şehirlerin sıralaması nasıl olmalıdır? sorusuna cevap aranır.

Gezgin satıcı problemindeki gezgin sayısının birden fazla olması durumuna benzeyen araç rotalama problemi de evrimsel hesaplama teknikleri ile çözülebilecek başka bir problemdir [8]. Bir depoda konumlanan araç filosu, müşterilere teslimat yapmak zorundadır. Araçların minimum maliyetle teslimatlarını yapabilmeleri için en optimum rota ne olmalıdır? sorusuna cevap aranır. Bu problemde araç kapasitesi ve teslimat zamanı gibi kısıtlar da mevcuttur [9, 10].

Yukarıdaki iki probleme benzer başka bir problem ise nakliye/taşıma problemidir. Bir ürün, farklı müşterilere farklı depolardan teslim edilmelidir. Bir müşteriye bir veya daha fazla depodan teslimat yapılabilir. Teslimat minimum maliyetle nasıl gerçekleştirilebilir? [11] sorusuna en uygun cevap bulunmaya çalışılır.

Zamanın planlanması ile ilgili problemler de yine planlanma kategorisi altında incelenebilir. Gerçek hayatta karşılaşılan bu problemlerden biri de atölye tipi çizelgeleme problemidir [12–15]. Bu problemde senaryo şu şekildedir; üretim yapan bir fabrikada farklı tiplerde makine ve bitirilmesi gereken belli sayıda iş kümesi vardır. Bu işlerin her birinin altında farklı operasyonlar yer alır. Her bir operasyona özel bir makine

ve operasyonun tamamlanması için özel bir zaman dilimi vardır. Yani bir operasyon belirli zaman aralığında belirli bir makine ile gerçekleştirilmektedir. Ayrıca operasyonlar belirli bir sıraya göre gerçekleştirilmektedir. Bu durumda, minimum maliyetle, tüm operasyonların tamamlanması için gerekli optimum planlama nasıl olmalıdır? sorusuna cevap aranır.

Eğitim kurumlarında sıkça karşılaşılan problemlerden biri de her ders/sınav döneminden önce, ders veya sınav takviminin/programının hazırlanmasıdır [16, 17]. Program, ders öğretmenlerinin ve öğrencilerinin memnuniyeti, sınıf kapasiteleri, ders mevcutları vb. kısıtlamalar göz önüne alınarak hazırlanmalıdır. En optimum program nasıl olmalıdır? Burada optimum durum, öğretmenlerin uygunluk durumu, sınıf mevcutları, dersi alan öğrenci sayısı gibi kriterlere göre değerlendirilir.

Evrimsel algoritmalar (EAs) paketleme ile ilgili problemlere de uygulanmıştır. Bu problemlerden biri de sırt çantası problemidir. Belli kapasiteye sahip bir sırt çantası, ve bu sırt çantasına yerleştirilecek belli boyutta ve sayıda eşyalar vardır. Burada amaç, sırt çantasına sığabilecek maksimum eşya setini bulmaktır.

İki boyutlu paketleme problemleri de vardır. Bunlara örnek bir senaryo şu şekilde verilebilir; Metal veya kumaş gibi maddelerden kesilerek üretilen ürünlerin, üretimi sırasında, kullanılmayan madde miktarının en aza indirilmesi gerekmektedir [18, 19]. Örneğin, elbise üreten bir fabrikada, büyük top kumaştan elbiseler kesilerek çıkarılacaktır. Maliyetin azalması için, geriye kalan kullanılamayan kumaş parçalarının en aza indirgenmesi gerekmektedir. En optimal kesim nasıl yapılmalıdır? sorusuna cevap aranır.

Üç boyutlu paketleme problemlerine örnek olarak malların teslimat için en çok mal alacak şekilde tırlara yüklenmesi gösterilebilir [20].

1.2.2. Tasarım

Bu kategoride yer alan problemlerin amacı, bir ürünün veya sistemin amacına en uygun şekilde üretilmesi için gereken optimum tasarımı bulabilmektir. Literatürde bu kategoriye dahil edilebilecek, EC uygulamaları bulunan bazı problemler, bu başlık altında toplanmıştır. Problemlere kısaca değinilerek detayı için referans gösterilmiştir.

Örnek problemlerden ilki, istenen frekansta cevap üretebilmesi için elektronik veya dijital filtrelerin tasarımı ile ilgilidir. Problemin detayı ve EC'ye dayalı geliştirilen çözüm [21]'de yer almaktadır.

Evrimsel algoritmalar, sinyal işleme sistemlerinin tasarımının optimizasyonunda [22] ve entegre devre tasarımında [23] da kullanılmıştır. Ayrıca entegre devre tasarım problemi ile benzer olan ve iki boyutlu olan departmanlar arasındaki bilgi akışının gerçekleştiği mesafenin minimize edilmesini amaçlayan "eşit olmayan alanlı tesis düzenleme" problemi için EC'ye dayalı bir çözüm [24]'de anlatılmaktadır.

Literatürde, birçok çalışmada, yapay sinir ağlarının tasarımında da EC yöntemleri kullanılmıştır. Hem ağ topolojisinin tasarımında hem de optimum ağırlık değerlerinin bulunmasında [25] kullanılan bu teknikler, ayrıca Kohonen ağ tasarımı için de uygulanmıştır [26].

1.2.3. Simulasyon ve Tanımlama

Bir sistemin bazı durumlarda bilinmeyen davranışlarını anlayabilmek için o sistemin modelinin veya tasarımının simule edilmesi gerekir. Davranışı bilinen bir sistemin ise doğruluğunu test etmek için simulasyonu gerçekleştirilir. Bu başlık altında literatürde EC tekniklerinin uygulandığı simulasyon ve tanımlama ile ilgili bazı problemlere kısaca değinilmiştir.

Kimya ve biyoloji alanlarında bazı bileşenlerin yapısının simulasyonuna ihtiyaç duyulmaktadır. Bazı durumlarda bu ihtiyacı karşılayacak çözümler, EC yöntemleri kullanılarak üretilmiştir. Mesela biyolojide, evrimsel stratejiler kullanılarak verilen aminoasit dizisinden oluşabilecek üç boyutlu protein yapılarının simulasyonu gerçekleştirilmiştir [27].

Tanımlama, simulasyonun tersidir. Girdi olarak sistemin davranışlarını alır ve sistemin olası tasarımını çıktı olarak verir. Sistem tanımlama yapmanın bir nedeni, verilen giriş değerlerine karşılık çıkış değerlerinin tahminlemesini yapabilmektir. Örneğin, klinik deneylerde hayatta kalma analizleri için istatistiksel fonksiyonların tahminlemesinde EC

yöntemleri kullanılmıştır [28].

1.2.4. Kontrol

Bir sistemin işleyişinin kontrolü ile ilgili problemlerin çözümü için kullanılan EC uygulamalarında dolaylı ve doğrudan kontrol olmak üzere iki tip kontrol vardır. Dolaylı kontrolde, EA algoritmaları, kontrolör tasarımında kullanılması söz konusudur. Bu durumda, kontrol işleminde evrimsel hiçbir şey yoktur, sadece kontrolör tasarımında vardır. Doğrudan kontrolde ise, kontrol işleminin aktif bir parçası olarak EA algoritmaları kullanılır. Evrimsel kontrolörün avantajı, zamanla değişen karakteristiğe sahip sistemlerde değişimin kademeli veya aniden olması farketmeksizin adaptasyonu sağlayabilmesidir.

Örneğin, Fonseca ve Fleming 1993 yılında gaz türbini motor kontrolörlerinin tasarımında EA kulanmıştır [29]. Ayrıca navigasyon sistemlerinin kontrolünde de EC yöntemleri kullanılmıştır [30].

1.2.5. Sınıflandırma

Sınıflandırma problemlerinde temel amaç, nesnelerin, olayların veya durumların kategorize edilmesi veya sınıflarına ayrılmasıdır. Literatürde bazı araştırmalarda, sınıflandırma problemleri için EC yöntemlerinin kullanıldığı çözümler geliştirilmiştir. Görüntü işlemede, OCR uygulamalarında, biyolojide zor bir işlem olan ikincil yapıdaki protein yapılarının belirlenmesinde [31], askeri uygulamalarda [32] ve daha birçok sınıflandırma probleminde EC yöntemlerinin kullanıldığı bilinmektedir.

1.3. Genetik Algoritma

Genetik algoritma (GA), optimizasyon problemlerinde, optimum çözümü bulmayı hedefleyen, olasılığa dayalı bir tür arama metodudur. Evrimsel hesaplamanın, yapay zeka alanında hızla büyüyen bir parçası olan GA, evrim teorisi ilham alınarak John Holland tarafından keşfedilmiş, sonrasında ise Holland ve arkadaşları tarafından geliştirilmiştir [33]. GA, çok boyutlu uzayda non-lineer birçok problemin çözümünün araştırılmasında

başarısını kanıtlamıştır [34–37].

Biyolojik bir teoriden esinlenilerek ortaya çıkarıldığı için genetik algoritmanın anlaşılmasına yardımcı olacak biyolojik altyapısından ve ilgili bazı terimlerden bahsetmek gerekir. Tüm canlı organizmalar hücrelerden oluşur. Bir organizmanın her bir hücresinde aynı kromozom seti bulunur. Kromozomlar, tüm organizmanın modelini oluşturan DNA dizileridir. Bir kromozom, genlerden, yani DNA parçalarından meydana gelir. Her bir gen belli bir proteini kodlar. Basitçe söylemek gerekirse, her bir gen, göz rengi gibi bir kişisel özelliği kodlar. Bir genin DNA'da kapladığı fiziksel alana lokus, bir lokusta bulunan olası özelliklerin her birine allel denir. Genetik materyalin tümüne ise genom adı verilir. Birkaç genin bir araya gelerek oluşturduğu gen kümelerine genotip, genotipin gelişerek canlıda meydana getirdiği göz rengi gibi fiziksel gelişimlere de fenotip adı verilir.

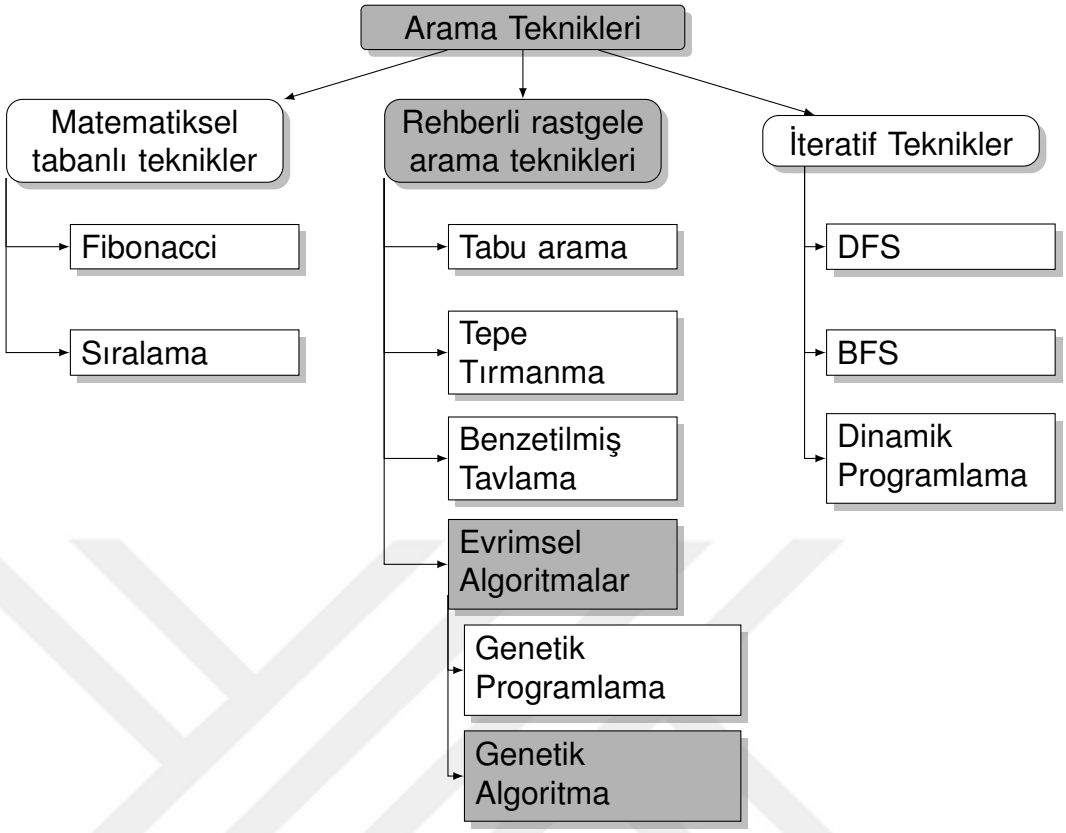
Canlıların çoğalma sürecinde, ilk olarak rekombinasyon veya çaprazlama işlemi gerçekleşir. Ebeveynlerin genleri bazı yollarla yeni kromozomu oluşturur. Yeni oluşan birey sonradan mutasyona uğrayabilir. Mutasyon, DNA elemanlarının değişmesidir. Canlı bir bireyin uygunluğu, o bireyin hayatta kalma başarısı ile ölçülür.

Yukarıda iki paragrafla tanımı yapılan her biyolojik materyalin veya durumun GA'da bir karşılığı vardır. Bu tanımlardan bazılarının GA'da işleniş biçimi aşağıda anlatılmıştır.

Bir problemin çözümünde, olası tüm çözümlerin kümesine arama uzayı denir. Arama uzayındaki her bir nokta, olası bir çözümü işaret eder. Problemin çözümü için, arama uzayında, en uygun çözüm veya çözümler araştırılır.

Genetik algoritmada, araştırma işlemine rastgele kromozomlardan oluşan bir popülasyon ile başlanır. Bir popülasyondaki çözümler, yeni bir popülasyon oluşturmak için kullanılır. Ancak bu işlem yapılırken tek motivasyon yeni popülasyonun önceki popülasyondan daha iyi olmasıdır. Yeni çözümler üretmek için eski popülasyondaki çözümler uygunluk değerlerine göre seçilirler. Dolayısıyla iyi olan çözümlerin yeni jenerasyona aktarılma şansı arttırılmış olur. Bu döngü bazı kriterler sağlanana kadar devam eder. Genetik algoritmanın temel adımları Şekil 1.6'da gösterilmiştir.

Geleneksel arama algoritmaları için Şekil 1.2'de verildiği gibi üç ana kategori



Şekil 1.2. Arama tekniklerinde Genetik Algoritma

tanımlanmıştır [34]: matematiksel tabanlı arama teknikleri, iteratif arama teknikleri, rastgele arama teknikleri. Bu kategoriler arasında yer alan diğer geleneksel arama algoritmalarından, GA'yı ayıran birçok özelliği vardır. Bu özellikler aşağıdaki listede verilmiştir.

1. Genetik algoritma, parametrelerin kendisi ile değil kodlanmış hali ile çalışır.
2. Araştırmaya arama uzayındaki tek çözümden değil bir popülasyondan başlar.
3. Türev gibi ara sonuç verisini değil, amaç fonksiyonundan dönen veriyi kullanır.
4. Olasılıksal kurallara dayanır.

Genetik algoritmada, kromozomlar 0 ve 1 'lerden oluşan bir dizi ile temsil edilir. Bu ikili gösterim Şekil 1.3'teki gibidir.

Kromozomdaki her bit çözümün bir özelliğini temsil eder. Problemin türüne göre kromozomların kodlanma şekli değişebilir. Bit dizisi bir numarayı kodlayacağı gibi

Kromozom X	1101100100110110
Kromozom Y	1101111000011110

Şekil 1.3. GA'da kromozom örneği

belli bir sıralamayı da kodlayabilir. Bu nedenle, genetik algoritmanın uygulanabileceği problem yelpazesi oldukça geniştir.

Problemin türüne göre kodlama işleminin biçimi belirlendikten sonra çaprazlama işlemine geçilir. Çaprazlama, ebeveyn kromozomlardan seçilen genlerden yeni bir birey üretme işlemidir. Çaprazlama için literatürde birçok yöntem geliştirilmiştir. Bunların en basiti, bir çaprazlama noktası seçerek ilk kromozomun o noktadan önceki bölümü ile ikinci kromozomun sonraki bölümünün birleştirilmesidir. Örnek bir çaprazlama işlemi Şekil 1.4'te gösterilmiştir.

Kromozom X	1101 - 100100110110
Kromozom Y	1101 - 111000011110
Yeni Birey X	1101 - 111000011110
Yeni Birey Y	1101 - 100100110110

Şekil 1.4. GA'da çaprazlama işlemi

Kromozomlardan oluşan bir popülasyonda hangi ikisinin çaprazlama işleminde ebeveyn olacağı da önemli bir problemdir. Çaprazlamaya girecek ebeveyn çözümlerinin seçimi için rulet tekerleği, turnuva yöntemi, elitizm vb. birçok yöntem geliştirilmiştir.

Yeni Birey X	1101111000011110
Yeni Birey Y	1101100100110110
Mutant X	1100111000011110
Mutant Y	1101100100010111

Şekil 1.5. GA'da mutasyon işlemi

Genetik algoritmanın diğer temel operatörü de mutasyondur. Mutasyon işlemi ile arama uzayındaki araştırmanın yerel optimuma takılmasının önüne geçilir. Mutasyon, çaprazlama işlemi ile oluşan yeni bireyde rastgele değişimler yapılarak gerçekleştirilir. İkili kodlamada, rastgele seçilen bir veya birden fazla bit değiştirilirek birey mutasyona

uğratılabilir. Basitçe değişim, bit dizisindeki 0 değerinin 1 yapılması veya 1 değerinin 0 yapılması işlemidir. Örnek mutasyon işlemi ikili bir dizi üzerinde Şekil 1.5'te gösterilmiştir.

Genetik algoritmada iki temel parametre vardır: çaprazlama ve mutasyon oranı. Çaprazlama oranı, çaprazlama işleminin gerçekleştirilme sıklığını belirtir. Çaprazlama yoksa, yeni birey ebeveynin aynısı olur. Çaprazlama oranının 100% olması durumunda, tüm bireyler çaprazlanıp üretilir. Çaprazlama oranının 0% olduğu durumda ise, yeni jenerasyon öncekinin aynısıdır.

Mutasyon oranı, bir kromozomun mutasyona uğrayan parçalarının sıklığını belirler. Mutasyon yoksa, yeni birey çaprazlamadan sonraki haliyle değişime uğramadan kalır. Mutasyon uygulanmışsa, değişim gerçekleşir. Mutasyon oranı 100% ise, tüm kromozom değişir, 0% ise hiçbir şey değişmez.

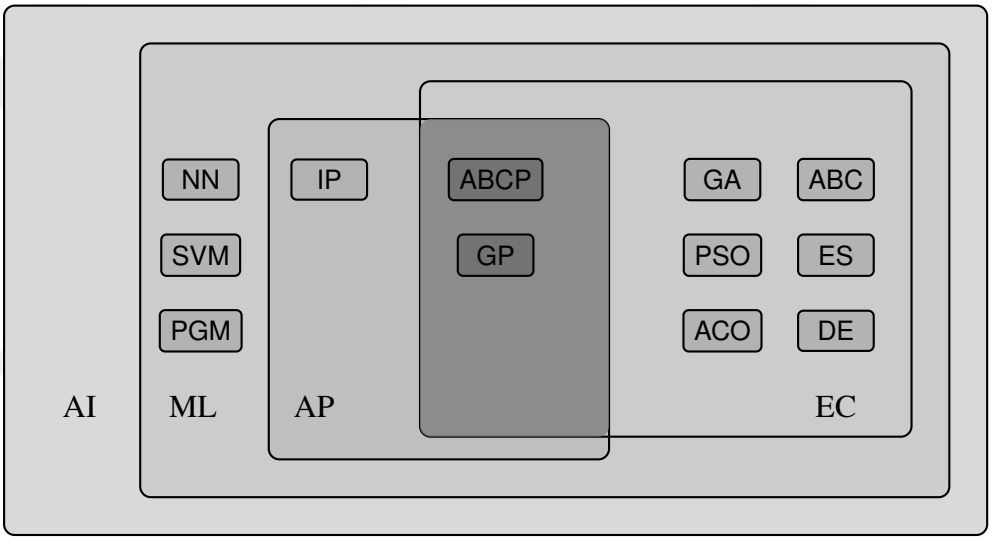
Genetik algoritmanın diğer parametrelerinden biri de popülasyon büyüklüğüdür. Popülasyon büyüklüğü, popülasyondaki kromozom sayısını belirtir.

1. **Başla** Rastgele N kromozomlu bir popülasyon oluştur.
2. **Uygunluk** Popülasyondaki her bir kromozomun (x) uygunluk değerini $f(x)$ hesapla.
3. **Yeni Popülasyon** Aşağıdaki adımları uygulayarak yeni popülasyonu üret.
    1. **Seçilim** Popülasyondan uygunluk değerlerine göre iki ebeveyn kromozom seç (iyi uygunluk, iyi seçilim şansı)
    2. **Çaprazlama** Bir çaprazlama oranına göre ebeveynleri çaprazlayarak yeni birey üret. Çaprazlama gerçekleşmezse, yeni birey ebeveynin kopyasıdır.
    3. **Mutasyon** Bir mutasyon oranına göre bireyleri mutasyona uğrat.
    4. **Kabul** Yeni bireyi yeni popülasyona dahil et.
4. **Yer değiştir** Eski popülasyonla yeni popülasyonu yer değiştir ve işleme yeni popülasyonla devam et.
5. **Test** Durdurma kriteri sağlandıysa, dur ve popülasyondaki en iyi çözümü döndür.
6. **Döngü** 2. adıma git.

Şekil 1.6. Genetik Algoritmanın Temel Adımları

1.4. Genetik Programlama

Genetik Programlama (GP), çözümünün formu veya yapısı hakkında herhangi bir bilgi olmayan problemlerin dışarıdan bir müdahale olmadan otomatik olarak çözülmesini sağlayan GA'ya dayalı bir evrimsel hesaplama yöntemidir. GP'nin amacı verilen problemi çözebilecek çalıştırılabilir programlar veya çözüm modelleri üretmektir. John Koza tarafından önerilmiş GA'nın özel bir formudur [38]. Genetik programlamanın, yapay zeka alanındaki yeri Şekil 1.7'de gösterilmektedir.



Şekil 1.7. GP ve ABCP 'nin yapay zeka alanındaki konumu

GP'de bilgisayar programlarından oluşan bir popülasyon, evrimin temel süreçlerinden geçirilerek geliştirilir. Bu gelişimde beklenti, GA'da olduğu gibi yeni oluşan jenerasyonların öncekilerden daha iyi olmasıdır. GP, bir problemin çözümünde yeni ve beklenmedik yollar bulabilir [39, 40].

Genetik programlamada başlangıç popülasyonunu oluşturmak için rastgele bilgisayar programları üretilir. Her bir program çalıştırılarak çıktılar edilir. Ardından elde edilen çıktı değerlerine göre programlara uygunluk değerleri atanır. Popülasyonun gelişimi için her programa bazı genetik operasyonlar uygulanır. Bu operasyonlar sonucunda evrimleşmiş yeni bireyler üretilir. Bu yeni bireyler yeni jenerasyonu yani yeni popülasyonu oluştururlar. Bu süreç durdurma kriteri sağlanana kadar veya kabul edilebilir çözüm/program bulunana kadar tekrar eder. Genetik Programlama algoritması

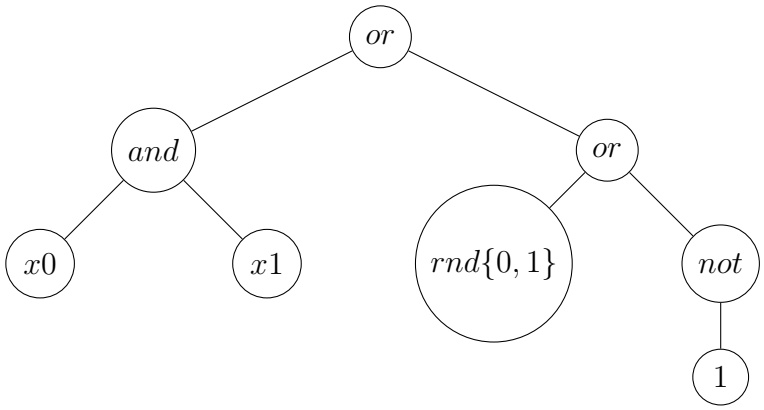
kabaca Şekil 1.8'de gösterilmiştir.

1. Rastgele programlar üreterek başlangıç popülasyonunu oluştur.
2. **Tekrarla** Kabul edilebilir bir sonuç buluncaya kadar veya durdurma kriteri sağlanana kadar
   1. Her bir programı çalıştır ve uygunluk değerini belirle.
   2. Genetik operasyonlar için popülasyondan uygunluk değerlerine programlar seç.
   3. Belli bir olasılığa göre genetik operasyonları uygulayarak yeni birey/bireyler oluştur.
3. En iyi çözümü/programı dönder.

Şekil 1.8. Genetik Programlama Adımları

GP 'nin temelleri her ne kadar GA 'yı andırsa da iki yöntemin ayrıldığı birkaç nokta vardır. Genetik programlamada bireyler GA'daki gibi bit dizisi olarak değil ağaç olarak gösterilir. Dolayısıyla, GA'daki genetik operatörler, bireylerin ağaç yapısına göre modifiye edilmiştir. Ayrıca uygunluk değerlendirme işlemi de genetik programlamada amaç fonksiyonuna göre değil programların çalıştırılmasına göre yapılmaktadır. Bu nedenle GA ve GP temelde evrimin operatörlerini kullanırlar ancak pratikte bahsedilen farklılıklar vardır.

GP'de bireyler ağaç olarak ifade edilir. Bu ağaçlar farklı boylarda ve yapılarda olabilirler. Popülasyondaki ağaç yapıları ve maksimum boyutları araştırmaya başlanmadan önce belirlenir. Örnek bir ağaç/program Şekil 1.9 'da gösterilmiştir.



Şekil 1.9. Genetik Programlama Ağaç Yapısı

Şekil 1.9'daki ağaçta olduğu gibi bir ağaçta fonksiyonlar ve terminaller olmak üzere iki düğüm tipi vardır. Bir terminal düğümünde olabilecek değerler aşağıda listelenmiştir.

- Sabit bir değer: probleme ilişkin sabitler sayı, dizi vb.
- Rastgele değer üreteci: çalışma esnasında ürettiği anlık değer
- Probleme ilişkin bağımsız değişken değeri: Şekil 1.9'daki $x0, x1$ düğümleri gibi.

Fonksiyonlar ise aldıkları girdilere göre çıktılar üretirler. Girdi, bir terminal olabileceği gibi bir başka fonksiyonun çıktısı da olabilir. Şekil 1.9'deki örnekte, fonksiyon olarak AND, OR ve NOT kapıları kullanılmıştır. İki fonksiyonun çıktısı başka bir fonksiyonun girdisi olmuştur.

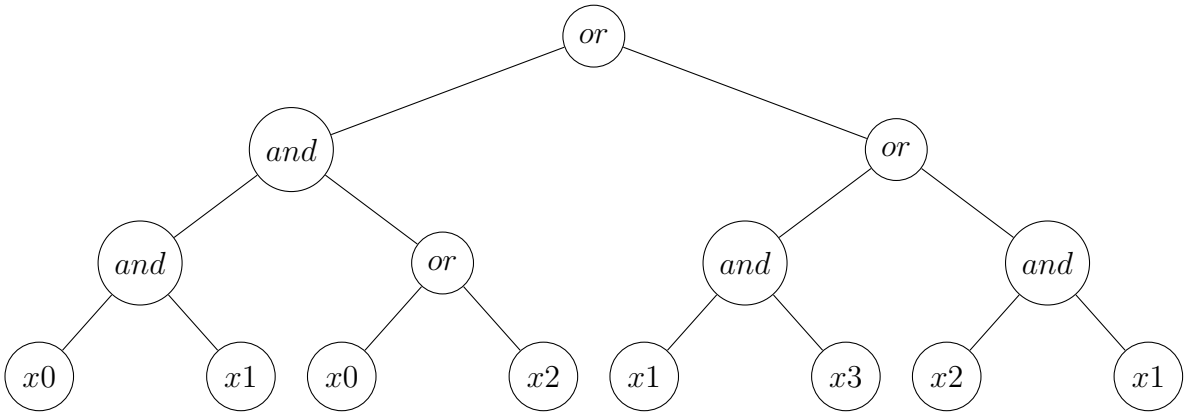
Bir ağaç, farklı yöntemler kullanılarak okunabilir. En sık bilinen okuma notasyonlardan biri içkök (infix) notasyonudur. Bu notasyona göre ağaç okunurken fonksiyonlar ifade içinde yer alır. Şekil 1.9'daki ağacın, içkök notasyonuna göre belirttiği ifade aşağıdaki gibidir:

$$(x0\ and\ x1)\ or\ (rnd\{0,1\}\ or\ not(1))$$

Bir düğümün derinliği, ağacın kök düğümü ve kendisi arasındaki kenar sayısına eşittir. Bir ağacın derinliği belirlenirken, ağacın en son düğümünün derinliği dikkate alınır. Yani ağaçtaki son düğümün derinliği, ağacın derinliğine eşittir. Örneğin, Şekil 1.9'daki ağacın derinliği 3'e eşittir çünkü en derindeki düğümün derinliği 3'tür.

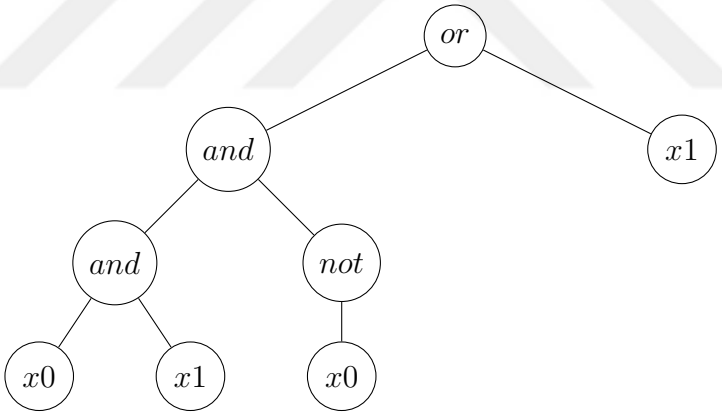
Diğer evrimsel algoritmalarda olduğu gibi, GP'de de başlangıç popülasyonu, rastgele bireylerin üretilmesi ile oluşturulur. Başlangıç popülasyonunu oluşturmak için grow, full ve ramped half-and-half olmak üzere üç farklı yöntem kullanılır [38]. Bu yöntemler aşağıda anlatılmıştır.

Full metodu ile oluşturulan bir ağacın tüm dalları aynı derinliğe sahiptir. Bu nedenle ağacın dengeli bir görünümü vardır. Ağacın tüm dalları belirtilen maksimum derinlik değerine sahip olmalıdır. Full metodu ile ağaç oluşturulurken son derinlikteki düğümler terminaller kümesinden, geri kalan tüm düğümler ise fonksiyonlar kümesinden seçilmelidir. Full metodu ile oluşturulmuş bir ağaç Şekil 1.10 'da gösterilmiştir.



Şekil 1.10. Full metodu ile oluşturulmuş ağaç örneği, $maksimum\ derinlik = 3$

Grow metodu ile oluşturulan bir ağacın dalları farklı derinliklerde olabilir. Bu nedenle bu metotla oluşturulan ağaçların boyut ve şekil olarak çeşitliliği, Full metodu ile oluşturulanlara nazaran daha fazladır. Ağaç oluşturulurken, düğümler, maksimum derinliği geçmeyecek şekilde istenilen kümeden (terminal veya fonksiyon) seçilebilir. Şekil 1.11'de örnek grow metodu ile oluşturulmuş örnek bir ağaç gösterilmektedir.



Şekil 1.11. Grow metodu ile oluşturulmuş ağaç örneği, $maksimum\ derinlik = 3$

Grow ve Full metotlarının öz yinelemeli algoritmalarının kaba kodu Şekil 1.12'de gösterilmektedir.

Ramped half-and-half metodu, başlangıç popülasyonunda kopya bireylerin oluşmasının önüne geçmek için Koza tarafından önerilmiştir. Bu metot ile çok çeşitli boyutlarda ve şekillerde ağaçlar oluşturulur. Full ve grow metotlarının kombinasyonudur. Ramped half-and-half metodu kullanıldığında, iki ile belirlenen maksimum derinlik değeri arasındaki her bir derinlik değeri için eşit sayıda ağaçlar oluşturulur. Örneğin, belirlenen maksimum derinlik 6 ise, popülasyondaki ağaçların %20 'sinin derinlik değeri 2, %20

fonksiyon: ifade_olustur(fonk_kumesi, term_kumesi, max_der, metot)

1: **if** $maks\_der = 0$ **or** $(metot = grow$ **and** $rand() < \frac{s(term\_kumesi)}{s(term\_kumesi)+s(fonk\_kumesi)})$ **then**
2: $agac = rastgele\_eleman\_sec(term\_kumesi)$
3: **else**
4: $fonk = rastgele\_eleman\_sec(fonk\_kumesi)$
5: **for** $i = 1$ **to** $arguman\_sayisi(fonk)$ **do**
6: $arg\_i = ifade\_olustur(fonk\_kumesi, term\_kumesi, max\_der - 1, metot)$
7: **end for**
8: $agac = (fonk, arg\_1, arg\_2, ...)$
9: **end if**
10: **return** $agac$

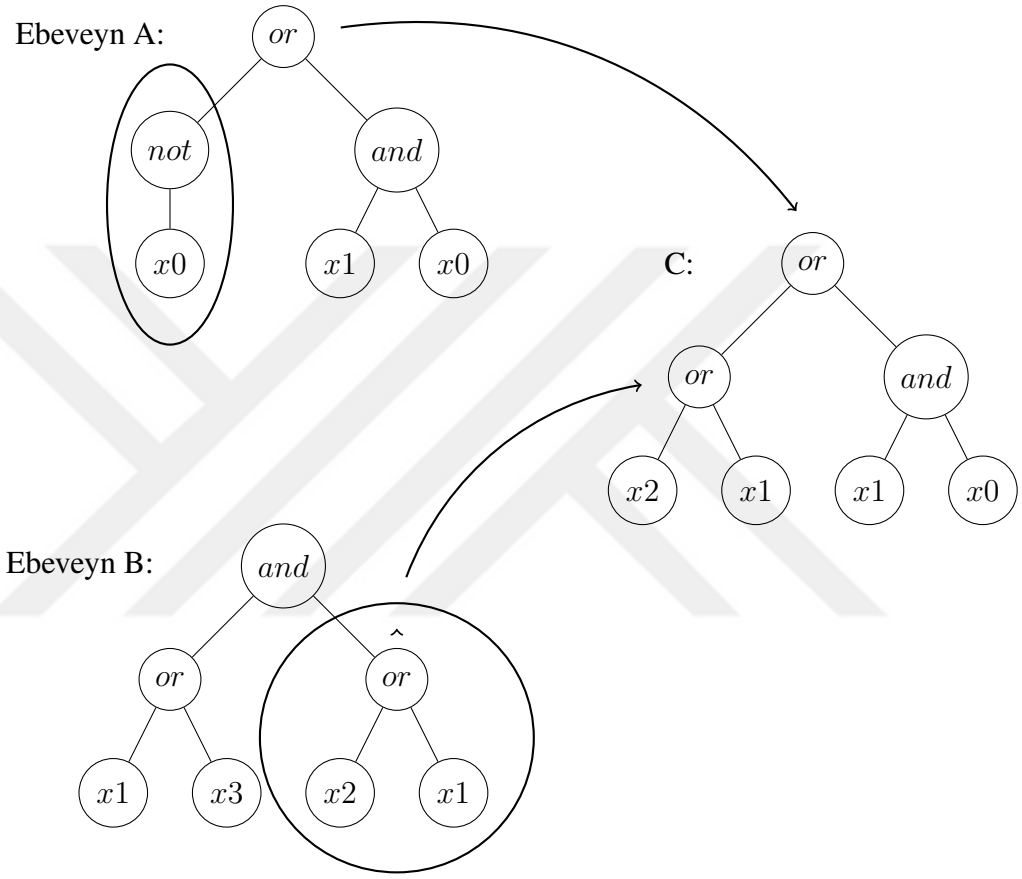
Şekil 1.12. Full ve Grow ağaç oluşturma öz yinelemeli fonksiyonunun kaba kodu

'sinin 3, ... (6 olana kadar) olur. Ayrıca, her bir derinlik değerindeki ağaçların yarısı full metodu ile diğer yarısı da grow metodu ile oluşturulur. Yani, popülasyonun %20 'si 2 derinlikli ise bu 2 derinlikli ağaçların yarısı grow metodu ile diğer yarısı ise full metodu ile oluşturulur [38].

GP'nin keşfinde önemli bir yere sahip olan çaprazlama operatörü ile yüksek uygunluk değerine sahip iki ebeveyn ağacı girdi olarak alınır, ardından bu iki ebeveynden yeni bir bireyin oluşması sağlanır [41–43]. Bu işlem için literatürde farklı yöntemler kullanılmaktadır. Yaygın olarak kullanılan yöntemlerden biri de alt-ağaç çaprazlamasıdır. Bu yönteme göre popülasyondan iki ebeveyn ağaç seçilir ve her biri için çaprazlama noktası (düğümü) belirlenir. Daha sonra, ilk ebeveynde belirlenen çaprazlama noktası, alt dalları ile birlikte kesilerek çıkarılır. Yerine ikinci ebeveynde belirlenen çaprazlama düğümü, alt dalları ile birlikte yerleştirilir. Böylece yeni birey elde edilmiş olur. Şekil 1.13'te iki ebeveyn ağacın (A,B) çaprazlanma işlemi ve sonucunda oluşturulan yeni ağaç (C) gösterilmiştir.

Çaprazlama noktaları, sürekli düzgün dağılıma (uniform) göre seçilmezler. Çünkü

düzgün dağılımla seçilim gerçekleştiğinde küçük miktarda genetik materyal (alt-ağaç) değişimi gerçekleşir. Yani çaprazlanacak alt ağaç boyutu küçük olur. Bu nedenle Koza, çaprazlama noktaları seçilirken %90 olasılıkla fonksiyonlardan %10 olasılıkla da terminallerden seçilmesini önermiştir [42].

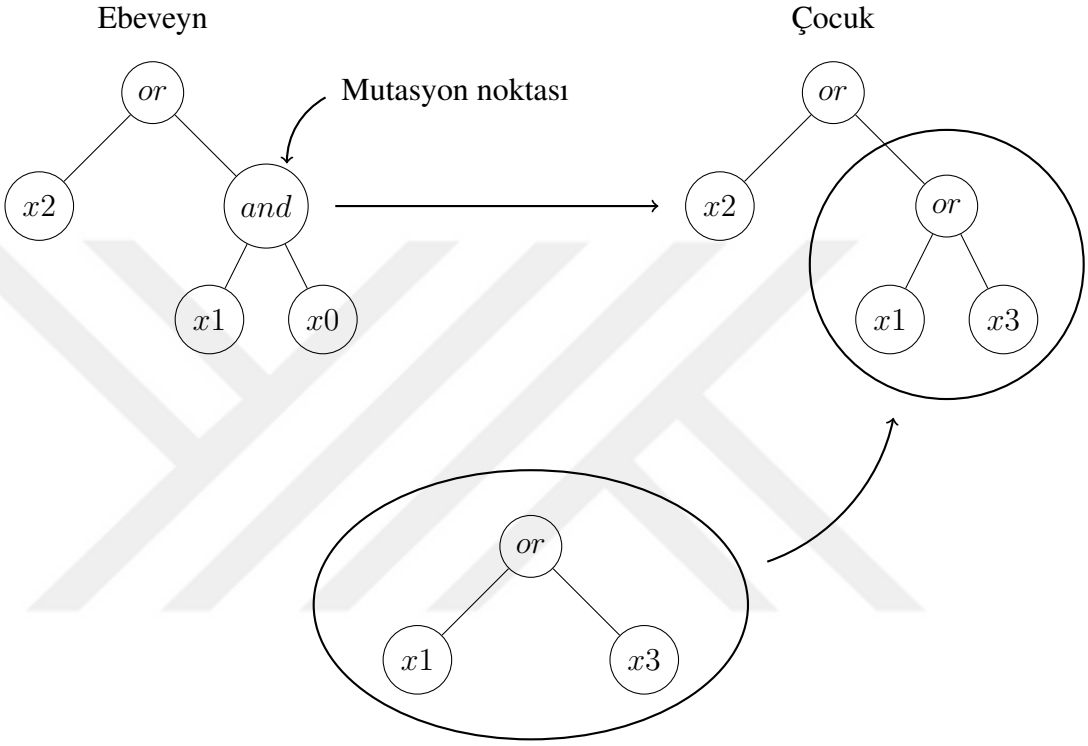


Şekil 1.13. Genetik programlamada çaprazlama işlemi

GP'de en bilinen mutasyon biçimi alt-ağaç mutasyonudur. Bu yöntemde bir ağaçta mutasyona uğrayacak düğüm seçilir ve bu seçilen düğümün yerine rastgele oluşturulmuş bir ağaç yerleştirilir. Şekil 1.14'te alt-ağaç mutasyon örneği gösterilmiştir. Mutasyon işlemi bazen bir program ile, rastgele oluşturulmuş yeni bir program arasında çaprazlama yapılarak da gerçekleştirilir. Bu yönteme de "başsız tavuk" çaprazlaması denir.

Çok bilinen diğer bir mutasyon yöntemi ise nokta mutasyondur. Genetik algoritmadaki bitlerde gerçekleştirilen mutasyon mantığına az da olsa benzemektedir. Nokta mutasyonda, rastgele bir düğüm seçilir ve sadece o düğüm (yaprakları aynen kalır) argüman sayısı aynı olan terminal veya fonksiyon kümesinden başka bir elemanla

değiştirilir. Eğer aynı argüman sayısına sahip bir eleman yoksa, bu durumda seçilen düğüm aynen kalır, değiştirilmez. Alt-ağaç mutasyonunda seçilen düğüm, kök kabul edilerek bir ağaç olarak mutasyona dahil edilir, nokta mutasyonda ise tek bir düğüm üzerinde işlem yapılır.



Şekil 1.14. Genetik programlamada alt-ağaç mutasyonu

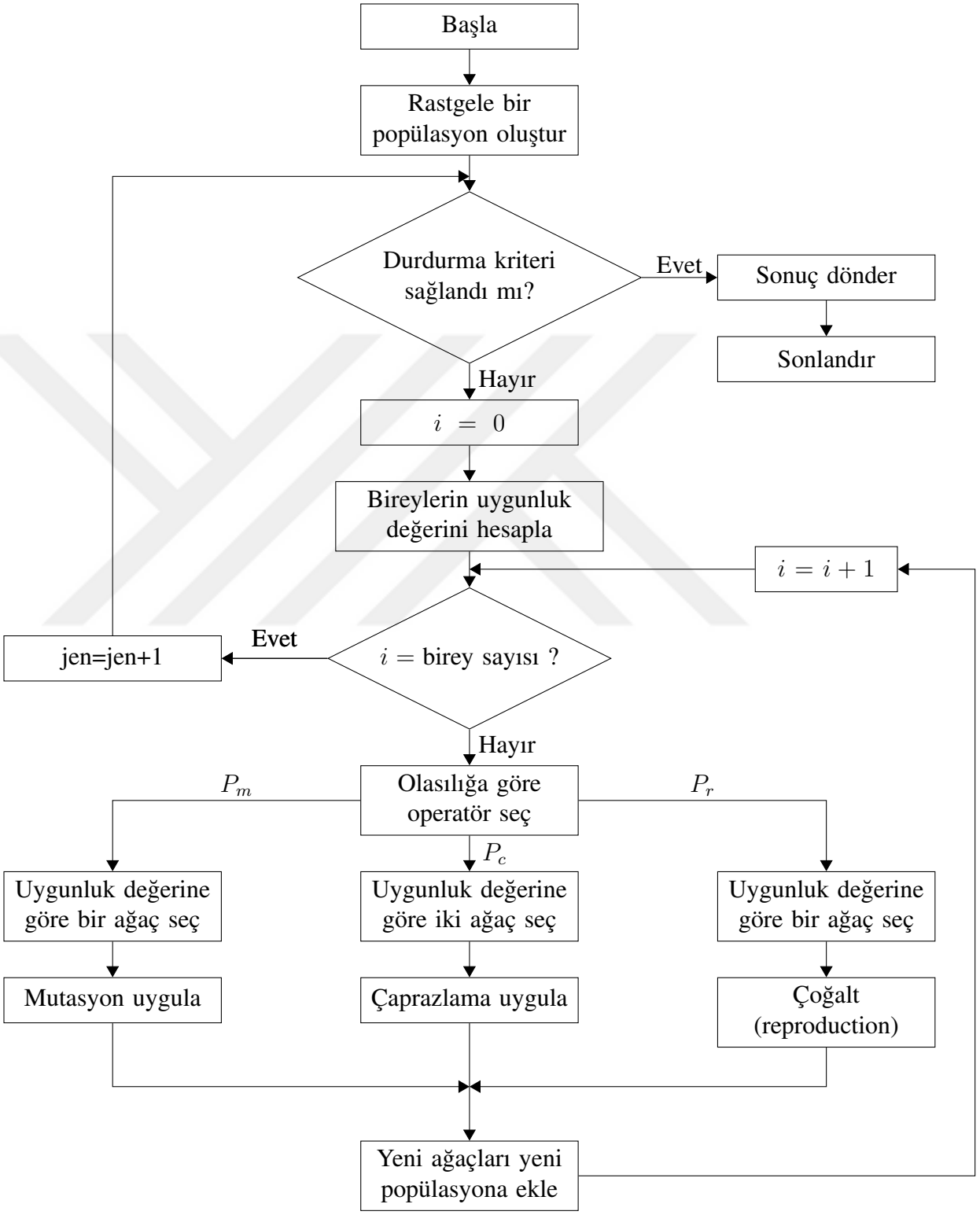
Yeni birey oluştururken kullanılacak operatörün seçimi, olasılığa dayalıdır. Diğer evrimsel algoritmalarda her bir çevrimde, birkaç operatör uygulanarak yeni birey elde edilirken, GP 'de belli bir olasılığa göre tek bir operatör seçilip uygulanır. Operatörlerin seçilim olasılıklarına operatör olasılık oranları denir. Genellikle, çaprazlama operatörü sık kullanılır, bu nedenle seçilim olasılığına (P_c) %90 gibi yüksek değerler verilir. Mutasyon operatörünün seçilim olasılık değerine (P_m) ise %10 'un altında küçük değerler verilir.

Kullanılan bir diğer operatör de çoğaltma (reproduction) operatörüdür. Mutasyon ve çaprazlama olasılık değerlerinin toplamının p olduğu durumda, çoğaltma operatörünün seçilim olasılığı (P_r), $1 - p$ kadar olur. Çoğaltma operatörü uygulanan bir ağaç, sonraki jenerasyona değiştirilmeden aynen aktarılır. Bu durum GP'nin akış şemasının yer aldığı

Şekil 1.15'te açıkça gösterilmektedir.

Bireylerin uygunluk değerlendirmesi için kullanılan birçok yöntem vardır. Örneğin, olması gereken değer ile programın çıktısı arasındaki hatanın miktarı bu yöntemlerden biridir. Bu yöntemde bir bireyin uygunluk değeri, o bireyin uygunluk durumlarında (fitness cases) verdiği çıktı değerleri ile olması gereken değerlerinin karşılaştırılması ile belirlenir. Bu metriğe, mutlak hata metriği denir.

GP'nin çalıştırılması için kontrol parametre değerlerinin belirlenmesi gerekir. Bu parametreler şunlardır; popülasyon büyüklüğü, başlangıç maksimum ağaç derinliği, fonksiyon kümesi, terminal kümesi, genetik operatörlerin kontrol parametreleri, ağaç oluşturma yöntemi (full, grow veya ramped half and half).



Şekil 1.15. GP akış diyagramı

2. BÖLÜM

YAPAY ARI KOLONİ PROGRAMLAMA

2.1. Giriş

Bu bölümünde önce, ABCPWeb'in temelini oluşturan, arıların yiyecek arama davranışından esinlenilerek geliştirilmiş Yapay Arı Koloni (ABC) algoritması, ardından ABC 'yi temel alan ve evrimsel hesaplama yöntemlerinden biri olan Yapay Arı Koloni Programlama (ABCP) anlatılmıştır.

2.2. Yapay Arı Koloni Algoritması

ABC algoritması, bal arılarının besin arama davranışları model alınarak 2005 yılında Karaboğa tarafından geliştirilmiş [44] sürü zekasına dayalı bir optimizasyon algoritmasıdır. Çok boyutlu ve çok modlu optimizasyon problemlerinin çözümü için geliştirilen ABC algoritması, genetik algoritma (GA), parçacık sürü optimizasyonu (PSO), diferansiyel gelişim (DE) gibi bilinen diğer algoritmalar ile karşılaştırılarak başarılı olduğu çeşitli çalışmalarla ortaya konmuştur [45].

ABC algoritmasında, bal arıları, görevlerine göre üç farklı gruba ayrılır: görevli arılar, gözcü arılar ve kaşif arılar. Bu türleri birbirinden ayırt eden özellikleri, üstlendikleri görevleridir. Yani her bir türün farklı görevi vardır.

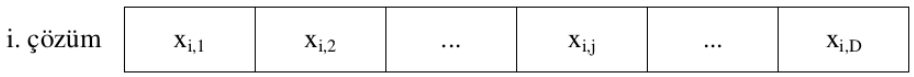
Görevli arı, belirli bir besin kaynağını tüketen ve bu kaynak ile ilgili bilgileri koloninin bulunduğu yuvaya taşıyan arıdır. Tükettiği besin kaynağı ile ilgili bilgileri (yönü, yuvaya uzaklığı, kalitesi vb.) dans ederek yuvada bekleyen arılara iletir. Yuvada, görevli arıların dansını izleyerek kaynakları hakkında bilgi edinen arılar da gözcü arılardır. Gözcü arılar, besin kaynaklarının kalite değerlerine göre her bir kaynağa bir olasılık değeri verirler. Bu

olasılık değerlerine göre tüketecekleri besin kaynaklarını seçerler. Kaşif arılar, tükenen kaynakların yerine, yeni kaynakların keşfi için yuva etrafında arama gerçekleştirirler.

Arılar, duruma göre görev değiştirerek farklı arı grubuna dahil olabilirler. Örneğin, bir gözcü arı, kovana gelen görevli arılardan aldığı bilgilerle kaynak seçimi yaptıktan sonra görevli arı grubuna dahil olarak, seçtiği kaynağı tüketmeye başlar [44].

Kaynak arayışının başında, yapay arı kolonisinin yarısı görevli, diğer yarısı gözcü arı olarak belirlenir. En başta kovandaki hiçbir arının hafızasında belirli bir besin kaynağı olmadığı için görevli arılar, kovan çevresinde yeni kaynak aramaya başlayarak kaşif arı olurlar. Kaşif arılar besin kaynağı bulduklarında tekrar görevli arı konumuna geçerek buldukları kaynakları tüketmeye ve kovandaki gözcü arılara bilgi getirmeye başlarlar. Görevli arılar, tükettikleri kaynaktan daha iyisini bulabilmek için komşuluk araştırması yaparlar. Bu işleme iyileştirme denemesi denebilir. Her bir kaynak için yapılan iyileştirme denemelerinin sayısı tutulur. Algoritmanın her bir çevriminde, bir kaynağın tükenme durumunun kontrolü için daha önce belirlenen "limit" değeri, en fazla deneme sayısına sahip kaynağın deneme sayısı ile karşılaştırılır. Deneme sayısı limit değerinden büyük veya eşit ise o kaynak terk edilir ve yerini rastgele belirlenmiş başka bir kaynak alır.

ABC algoritmasında çözümler (besin kaynakları) bir boyutlu dizi olarak gösterilir. Bu dizinin eleman sayısı (D) problemdeki karar değişkeni kadardır ve araştırma boyunca sabittir. Çözümler, Şekil 2.1'deki gösterilmektedir.



Şekil 2.1. ABC algoritmasında bir çözüm.

ABC modelini oluştururken bazı kontrol parametreleri belirlenmelidir. Bu parametreler aşağıdaki listede belirtilmiştir:

- **Koloni büyüklüğü** ($2 \times SN$), kolonideki toplam arı sayısını belirtir. Besin kaynağı sayısı (SN), görevli arı sayısına eşit ve koloni büyüklüğünün yarısına eşittir.
- **Maksimum çevrim sayısı** (MCN), besin kaynağı araştırma sürecinin tekrarlanma

sayısıdır.

- **Limit**, bir besin kaynağının tükenme durumunu kontrol için kullanılır.

Başlangıç aşamasında, görevli arılar kaşif arı gibi davranarak rastgele çözümler üretilir. Çözümler üretilirken Eşitlik 2.1 'den yararlanılır. Ayrıca, kaşif arı fazında rastgele yeni bir çözüm üretileceği zaman da aynı şekilde Eşitlik 2.1 kullanılır.

$$x_{i,j} = l_j + rand(0,1) * (u_j - l_j) \tag{2.1}$$

Eşitlik 2.1'deki i, i'inci çözümü, j ise çözümün j'inci elemanını diğer bir deyişle problemin j'in boyutunu belirtmektedir. Yine eşitlikte yer alan l_j ve u_j, problemin j'inci boyutunun sırasıyla alt ve üst sınırlarını göstermektedir. Bu eşitlik ile bir çözümün bir boyutundaki değer hesaplanmış olur. Yeni bir çözüm oluşturmak için problemin boyutu D kadar rastgele değer Eşitlik 2.1 kullanılarak hesaplanır.

Görevli arı, daha iyi besin kaynağı bulmak için popülasyonda bulunan bir kaynaktan faydalanarak Eşitlik 2.2 ile komşuluk araştırması yapar. Araştırma sonucunda yeni bir aday çözüm v_i üretilir.

$$v_{i,j} = x_{i,j} + \varphi_{i,j} * (x_{i,j} - x_{k,j}) \tag{2.2}$$

Komşuluk araştırmasında, tüketilen kaynak ($x_{i,j}$) ile faydalanılan kaynağın ($x_{k,j}$) farklı olması gerekir. Bu yüzden $k \neq i$ şartı sağlanmalıdır. Eşitlikte belirtilen j, rastgele seçilmiş bir boyutu, $\varphi_{i,j}$ ise $[-1, 1]$ aralığında rastgele belirlenmiş bir değeri göstermektedir.

Görevli arı, komşuluk araştırması yaparak ürettiği aday çözüm v_i ile tükettiği çözüm x_i arasında aç gözlü seleksiyon uygulayarak, uygunluk değeri yüksek olanı seçer. Böylece iyi olan çözümler popülasyonda kalmış olur. Uygunluk değerinin $fit(x_i)$'nin hesaplanması için Eşitlik 2.3 kullanılır.

$$fit(x_i) = \begin{cases} \frac{1}{1+f(x_i)}, & \text{eğer } f(x_i) \geq 0 \\ 1 + abs(f(x_i)), & \text{eğer } f(x_i) < 0 \end{cases} \tag{2.3}$$

Eşitlik 2.3'teki $f(x_i)$, x_i çözümünün maliyet değerini göstermektedir.

Gözcü arı, izlediği görevli arıların danslarına birer olasılık değeri verir. Bu değeri kullanarak gideceği besin kaynağını seçer. ABC modelinde her bir çözümün seçilim olasılık değeri Eşitlik 2.4 'e göre hesaplanır.

$$P_i = \frac{fit(x_i)}{\sum_{i=1}^{SN} fit(x_i)} \tag{2.4}$$

Yukarıdaki eşitlikte, SN, toplam besin kaynağı sayısını, P_i ise i'inci çözümün olasılık değerini belirtir.

ABC algoritması Şekil 2.2'de adım adım gösterilmiştir.

Kontrol parametrelerini belirle.
Başlangıç çözümlerini (x_i) rastgele üret ve değerlendir.
En iyi çözümü hafızaya al.
Çevrim sayacını başlat (çevrim = 0)
TEKRARLA
  Her bir görevli arı için;
    Komşuluk araştırması yaparak aday bir çözüm (v_i) üret.
    x_i ve v_i arasında uygunluk değerine göre aç gözlü seleksiyon uygula. v_i seçilirse
      i. çözüme ait deneme sayacını sıfırla, seçilmezse bir arttır.
  Tüm çözümlerin seçilim olasılıklarını (P_i) hesapla.
  Her bir gözcü arı için;
    P_i 'ye göre bir çözüm seç.
    Komşuluk araştırması yaparak aday bir çözüm (v_i) üret.
    x_i ve v_i arasında uygunluk değerine göre aç gözlü seleksiyon uygula. v_i seçilirse
      i. çözüme ait deneme sayacını sıfırla, seçilmezse bir arttır.
  Hafızadaki en iyi çözümü güncelle.
  Kaşif arı için, tükenmiş bir çözüm varsa;
    Yeni bir çözüm üret ve tükenmiş çözümle değiştir.
    Yeni çözümün deneme sayacını sıfırla.
  çevrim = çevrim + 1
KADAR (çevrim = maksimum çevrim sayısı)

Şekil 2.2. ABC algoritması

2.3. Yapay Arı Koloni Programlama

Yapay Arı Koloni Programlama (ABCP), ABC algoritmasında olduğu gibi arıların yiyecek arayışını temel alarak bilgisayar programları üreten bir otomatik programlama yöntemidir. Giriş ve çıkış değerleri bulunan, ve çözüm yöntemi bilinmeyen bazı problemlerin çözümünde kullanılır. ABC algoritması ile arasındaki temel fark, çözümlerin gösteriminde ve bazı evrimsel operatörlerin çözümlerin gösterimine uyarlanmasındadır. [1].

ABCP'deki yiyecek kaynakları, Şekil 1.9'daki gibi ağaçlarla temsil edilen rastgele

oluşturulmuş bilgisayar programlarıdır. Birey, ağaç, program ve yiyecek kaynağı ifadeleri aynı anlamı taşırlar. Bu terimlerin hepsi çözülmesi beklenen bir problemin olası bir çözümünü ifade eder.

Bir ağaç önceden belirlenen terminal ve fonksiyon düğümlerinden oluşur. Problemin türüne göre belirlenen terminal ve fonksiyonlar değişebilir. Örneğin, sembolik regresyon problemlerinde fonksiyonlar kümesinde, çarpma, çıkarma, toplama vb. aritmetik operatörler, terminaller kümesinde ise, sabit sayılar ve herhangi bir veri setindeki değerleri barındıran değişkenler (x_1, x_2, y_1) yer alabilir.

Bölüm 1.4 'te, GP'de ağaç oluşturma tekniklerinden Grow, Full ve Ramped half-and-half metotları anlatılmıştır. ABCP'de de aynı şekilde başlangıç popülasyonundaki yiyecek kaynakları bu metotlarla oluşturulur. Popülasyondaki birey tekrarını önlemek için, kaşif arı fazında oluşturulan yeni bireyin diğer bireylerle aynı olmamasına dikkat edilir. Bu fazda yeni birey Grow metodu ile oluşturulur.

Bireylerin değerlendirilmesi için birçok metrik kullanılır. Bunlardan biri Eşitlik 2.5'te gösterilen toplam mutlak hata fonksiyonudur.

$$f(x_i) = \sum_{j=1}^{N} |(c_j - o_j)| \tag{2.5}$$

Yukarıdaki eşitlikte, N toplam uygunluk durumu sayısını, c_j ve o_j sırasıyla değerlendirilen i. bireyin çıktı değerini ve olması gereken değeri belirtir. Bu fonksiyonla toplam mutlak hata değeri ölçülmüş olur. Hata değeri ne kadar büyük olursa birey o kadar kötüdür.

$$P_i = \frac{1}{1 + f(x_i)} \tag{2.6}$$

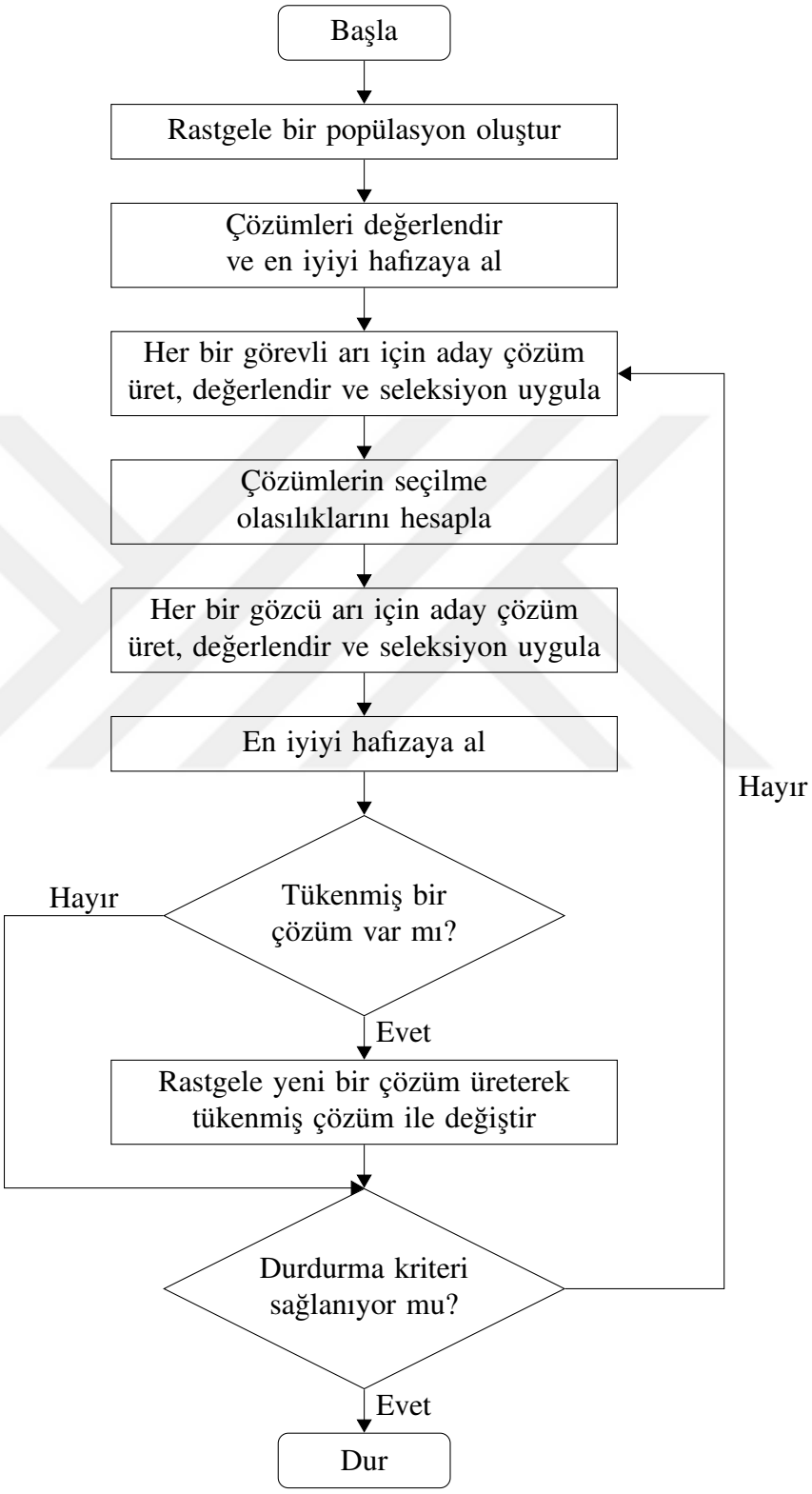
ABC algoritmasında olduğu gibi, gözcü arıların tüketecekleri kaynakları seçebilmesi için tüm kaynaklara kalite değerlerine göre olasılık değeri verilmesi gerekir. Kalite değerleri Eşitlik 2.6 ile belirlenen kaynaklara Eşitlik 2.7 kullanılarak birer olasılık değeri atanır.

$$P_i = \frac{\alpha \times fit(x_i)}{fit(x_{best})} + (1 - \alpha) \tag{2.7}$$

Eşitlik 2.7'deki, α parametresi, 0 ile 1 arasında değer alır.

ABC algoritmasında gerçekleştirilen komşuluk araştırması, ABCP algoritmasında yerini paylaşım mekanizmasına bırakır [1]. Paylaşım mekanizması, Şekil 1.13'te gösterilen GP'de kullanılan çaprazlama işlemi ile aynıdır. Bu işlemde tüketilen kaynaktan rastgele bir düğüm seçilir. Bu düğüm %90 olasılıkla fonksiyonlardan veya %10 olasılıkla terminallerden seçilebilir ve alt dalları ile birlikte kesilerek kaldırılır. Yerine, popülasyondan rastgele seçilen başka bir kaynağın aynı şekilde seçilen düğümü yerleştirilir. Böylece bireyler arasında paylaşım gerçekleşmiş olur.

Şekil 2.3'te ABCP'nin akış diyagramı gösterilmektedir. Diyagramda yer alan aday çözüm üretme işleminde, paylaşım mekanizması kullanılır. Kaşif arı fazında ise Grow metodu ile yeni bir ağaç oluşturularak tükenmiş olan kaynakla değiştirilir.



Şekil 2.3. ABCP akış diyagramı

3. BÖLÜM

ABCP WEB YAZILIMI

3.1. Giriş

ABCPWeb yazılımı, bu tez çalışması kapsamında, ABCP algoritmasının bulut tabanlı web ortamına taşınması ile geliştirilmiştir. Bu yazılımla birlikte kullanıcı, sisteme yüklediği verileri ABCP algoritması ile kolayca işleyerek sonuçlar elde edebilecektir. Aynı zamanda internet erişimi olan her yerden, yüklediği verilere ve çalıştırdığı deney bilgilerine kolayca erişebilecektir. ABCPWeb yazılımı mobil uyumlu olduğu için, kullanıcı, herhangi bir akıllı cihaz ile yazılımı kullanabilecektir.

Yazılımın çekirdeğinde yer alan ABCP modülü, Bölüm 2'de anlatılan ABCP algoritmasına göre kodlanmıştır. Kullanıcının, ABCPWeb'i kullanması için herhangi bir yazılım dili bilmesine gerek yoktur. Bu kolaylıkla birlikte kullanıcı, ABCP algoritmasını kodlamakla zaman kaybetmeyecektir.

ABCPWeb yazılımı Python programlama dili kullanılarak geliştirilmiştir. Web modülü için DJango frameworkü ile PostgreSQL veritabanı kullanılmıştır. Kullanıcı dostu bir tasarım için Twitter Bootstrap kütüphanesinden yararlanılmıştır.

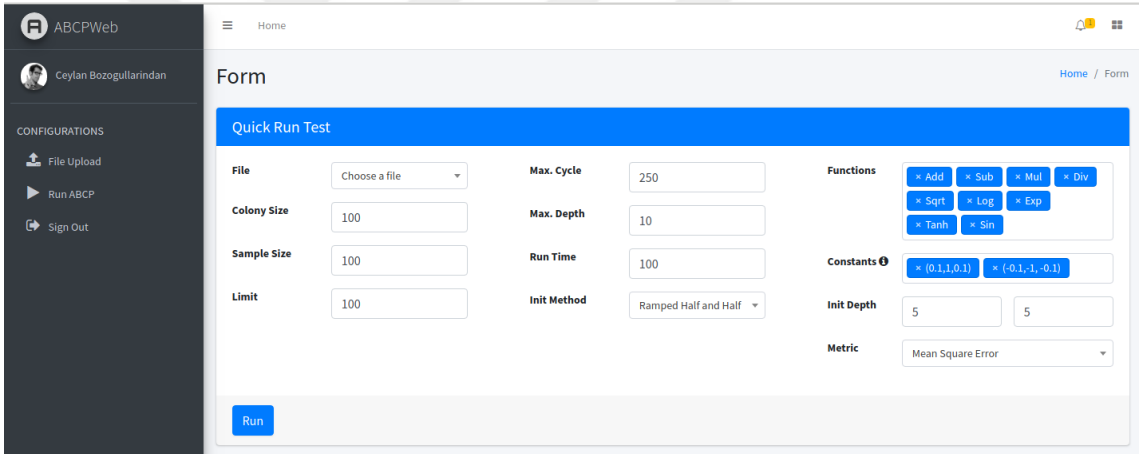
Yazılımı araştırmacıların kullanımına açmak amacıyla **"www.abcpweb.com"** alan adı alınarak, sistem hayata geçirilmiştir. İlk versiyonda kullanıcı kaydı olmadığı için, üyelikler yönetici tarafından verilmektedir. Sistemi kullanmak isteyen kullanıcıların sistem yöneticiyle iletişime geçmesi gerekmektedir.

Sistemde, büyük işlemler, asenkron olarak arka planda çalıştığından, kullanıcının diğer işlemleri kesintiye uğramaz. Sonuçların dökümü işlem bitmeden, sonuç geldikçe verilir.

Bu sayede işlemde meydana gelebilecek herhangi bir hata nedeniyle veri kaybı yaşanmaz. Sonuçlar hem görsel olarak web sayfasında gösterilir hem de dosyaya kaydedilerek kullanıcıya sunulur.

Kullanıcı kendi veri seti kütüphanesini uygulama içinde oluşturabilir. Bir defa yüklediği veri setini, ikinci defa yüklemeksizin çeşitli işlemler için kullanabilir. Güvenlik amacı ile sisteme yüklenen dosyaların adları belli formatlara göre değiştirilir.

Şekil 3.1'de ABCPWeb yazılımının genel arayüzü gösterilmiştir. Bu arayüzde sol tarafta bir menü yer alır. Bu menüde kullanıcının yapmak istediği işlemler bulunur. İlk versiyon için üç seçenek vardır; veri seti yükleme, yüklenen veri setini ABCP algoritmasını kullanarak çalıştırma ve çıkış yapma seçenekleri.



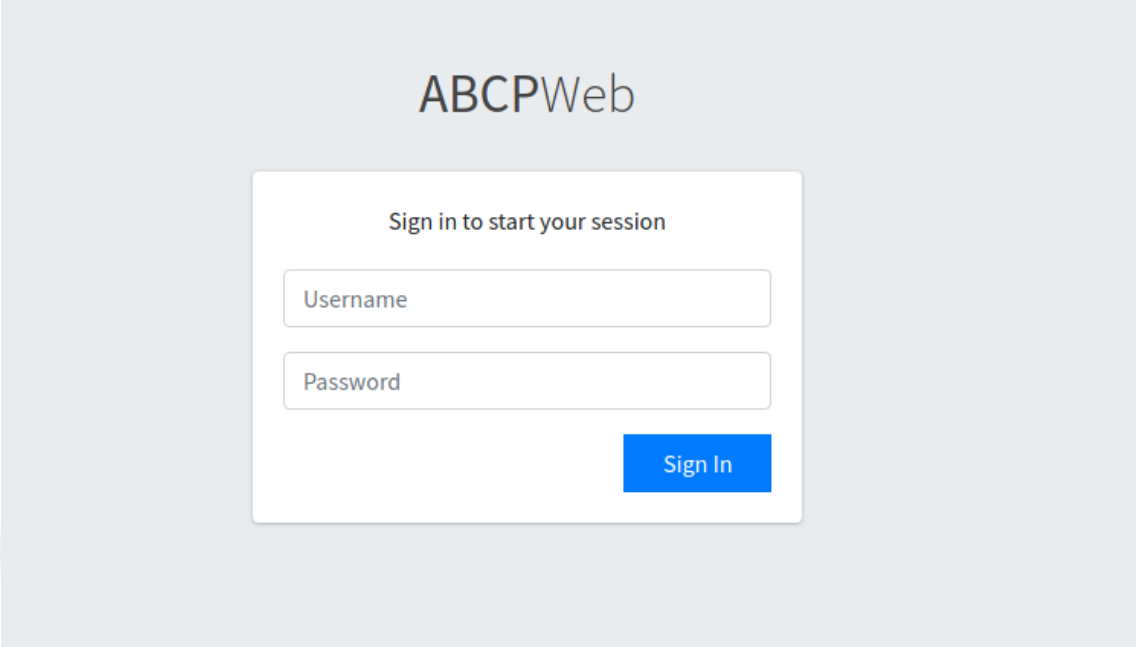
Şekil 3.1. ABCPWeb Genel Arayüz

Bu bölümde sırasıyla ABCPWeb yazılımındaki, üyelik sistemi, verinin yüklenmesi, ABCP'nin çalıştırılması ve sonuçların alınması işlemleri anlatılmıştır.

3.2. ABCPWeb: Üyelik Sistemi

Yazılımı kullanabilmek için sisteme üye olmak gerekmektedir. Üyelik bilgileri ilk versiyon için yönetim tarafından sağlanır. Kullanıcı, sisteme, verilen bilgilerle giriş yapar. Şekil 3.2'de ABCPWeb yazılımının üye giriş ekranının arayüzü gösterilmektedir.

Üyelik sisteminde yönetici ve üye dışında herhangi bir rol tanımı yapılmamıştır. Bu nedenle tüm üyeler aynı haklara sahiptir. Sonraki versiyonlar için, üyelik sistemi paketlere



Şekil 3.2. ABCPWeb Üye Giriş Ekranı

ayrılarak farklı rol tanımlarının yapılması planlanmaktadır.

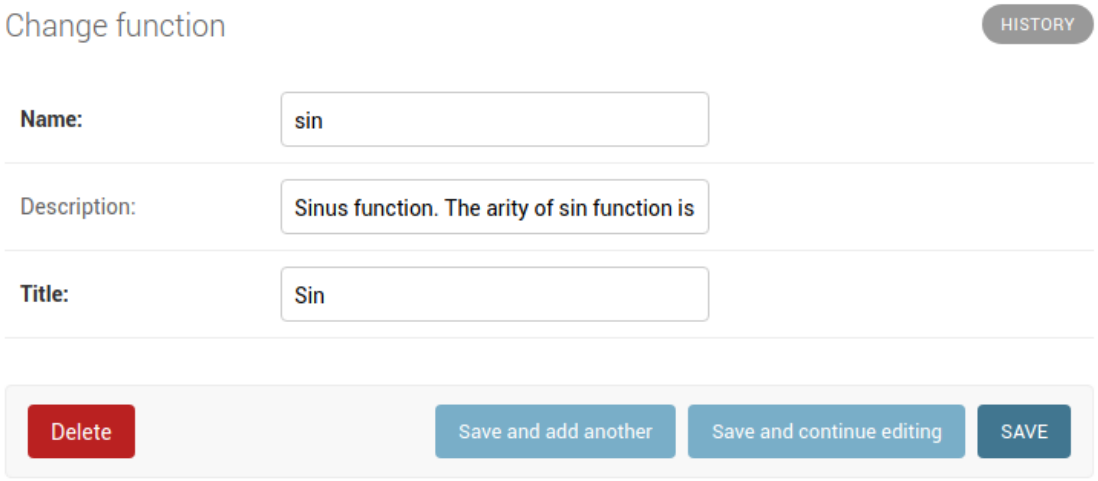
Yöneticinin hakları aşağıdaki listede gösterildiği şekildedir:

1. Üye yönetimi - Üye ekleme, silme ve düzenleme yetkisi
2. Metrik yönetimi - Metrik ekleme, silme ve düzenleme yetkisi
3. Fonksiyon kümesi yönetimi - Fonksiyon ekleme, silme ve düzenleme yetkisi

Listede yer alan, yöneticinin metrik ve fonksiyon kümesi yönetimi maddeleri ABCP ile ilgilidir. ABCP algoritmasını çalıştırmak için kontrol parametresi olarak metrik ve fonksiyon kümeleri tanımlanmalıdır. Kullanıcı, yazılımın bu versiyonunda kendi fonksiyon kümesini ve metriğini oluşturamadığı için bu değerler yönetici tarafından ön tanımlıdır. Yani kullanıcı sadece ön tanımlı fonksiyonları ve metriği seçebilir. Sonraki versiyonlarda kullanıcılara yeni metrik ve fonksiyon ekleme yetisinin kazandırılması planlanmaktadır.

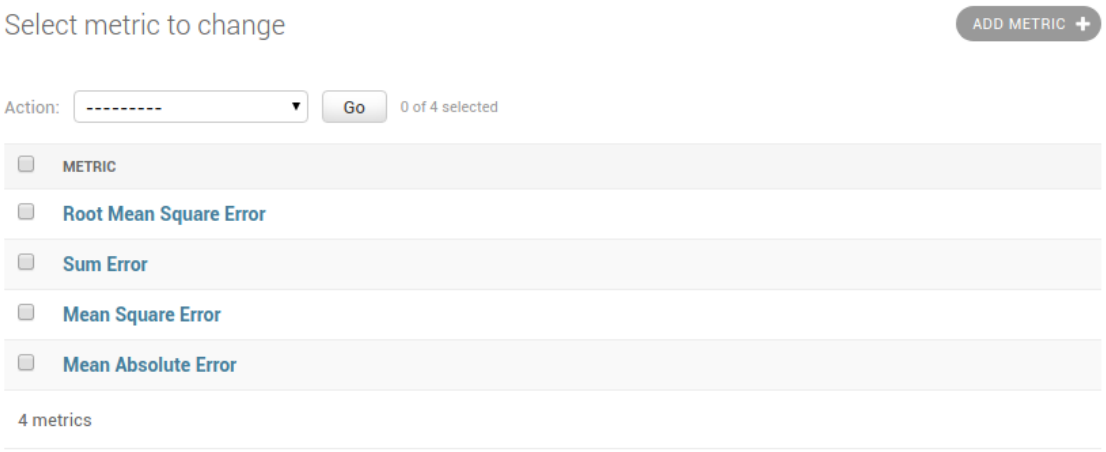
Yönetici fonksiyonun, kodda yer alan ismini ve son kullanıcıya gösterilecek ismini yönetim panelinden ekler. Şekil 3.3'te bu işlemi gerçekleştirdiği arayüz gösterilmiştir.

Şekil 3.4'te ise metrik yönetimi ile ilgili panelin arayüzü gösterilmektedir. Şekilde de



Şekil 3.3. ABCPWeb Yönetim Paneli - Fonksiyon Düzenleme

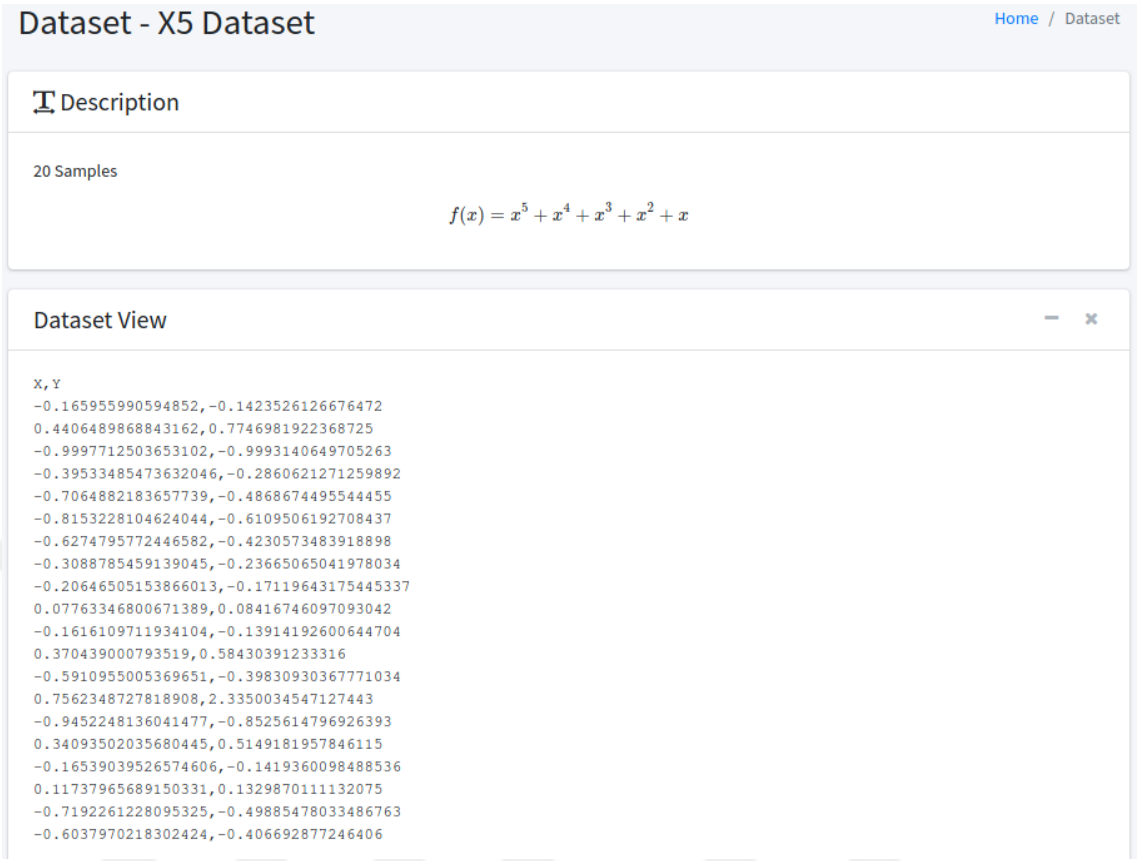
görüldüğü gibi şuanda sistemde 4 adet metrik tanımlanmıştır. Bu nedenle kullanıcı ancak bunlardan birini seçebilir.



Şekil 3.4. ABCPWeb Yönetim Paneli - Metrik Yönetimi

3.3. ABCPWeb: Veri Yükleme

Kullanıcı üye girişi yaptıktan sonra veri yükleyebilir. Veri, her bir satırın formu $X1, X2, X3...Y$ arada virgül olacak şekilde metin belgesi (.txt) formatında olmalıdır. İlk versiyon, tek bir bağımlı değişkeni (Y) desteklemektedir. Yüklenen veri setleri yönetici tarafından belirlenen limitten küçük olduğu durumda yazılım üzerinden görüntülenebilir. Şekil 3.5'te, ABCPWeb veri seti görüntüleme arayüzü gösterilmiştir.



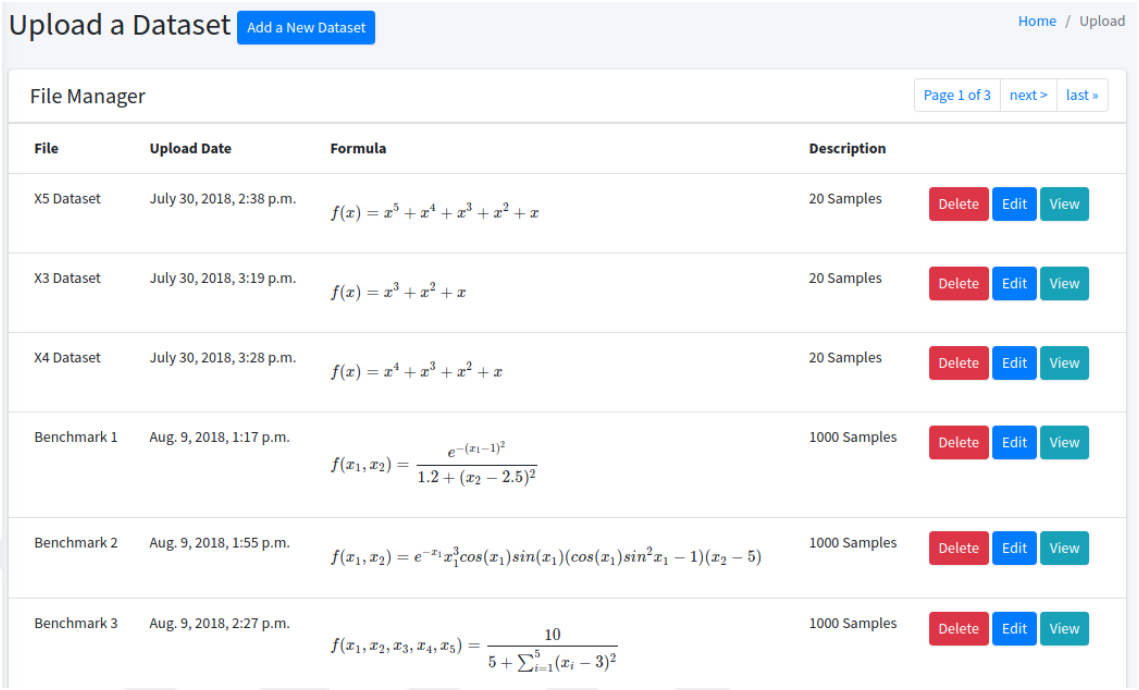
Şekil 3.5. ABCPWeb - Veri Seti Gösterim Arayüzü

Şekil 3.5'te veri setinin temsil ettiği eşitlik de veri setinin açıklama kısmında görülmektedir. Burada ve Şekil 3.6'da gösterilen veri seti listesinde, kullanıcıya matematiksel eşitliklerin kolayca yazabilmesi için LaTeX kullanım imkanı sağlanmıştır.

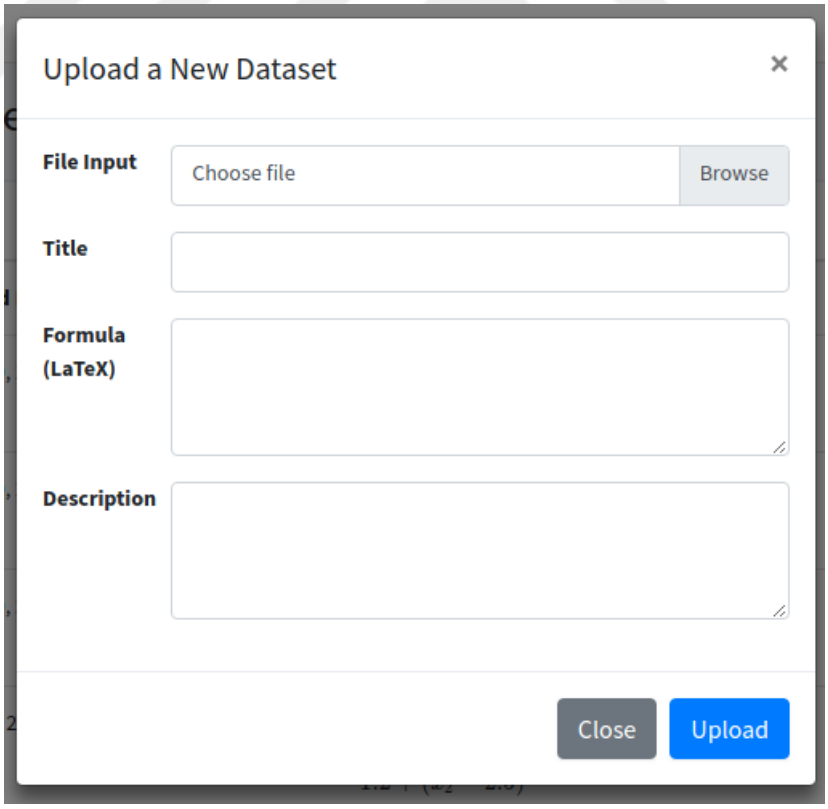
Veri yükleme işlemi için kullanıcıdan veri dosyası, veri seti başlığı, verinin temsil ettiği eşitlik varsa LaTeX formatında formülü ve açıklama bilgilerini girmesi istenir. Alınan bilgilerle dosya sunucuya taşınır ve veritabanına bilgiler kaydedilir. Şekil 3.7'de veri seti yükleme arayüzü gösterilmiştir.

Kullanıcının yüklediği veri setleri zamanla çoğaldığında, bunların yönetimi zorlaşacağından ABCPWeb uygulamasında veri setlerinin düzgün biçimde görüntülenmesi sağlanmıştır. Şekil 3.6'da gösterilen veri seti listesi arayüzünde, kullanıcı, yüklediği veri setlere dair bilgileri listede görüntüleyebilir, silebilir veya veri seti ile ilgili bilgileri düzenleyebilir.

Veri listesi arayüzünde, formül sütunun olmasının nedeni, amaç fonksiyonu bilinen veriler yüklendiğinde ilgili veri setinin hangi amaç fonksiyonunu temsil ettiğini görebilmektir.



Şekil 3.6. ABCPWeb - Veri Seti Listesi Gösterim Arayüzü



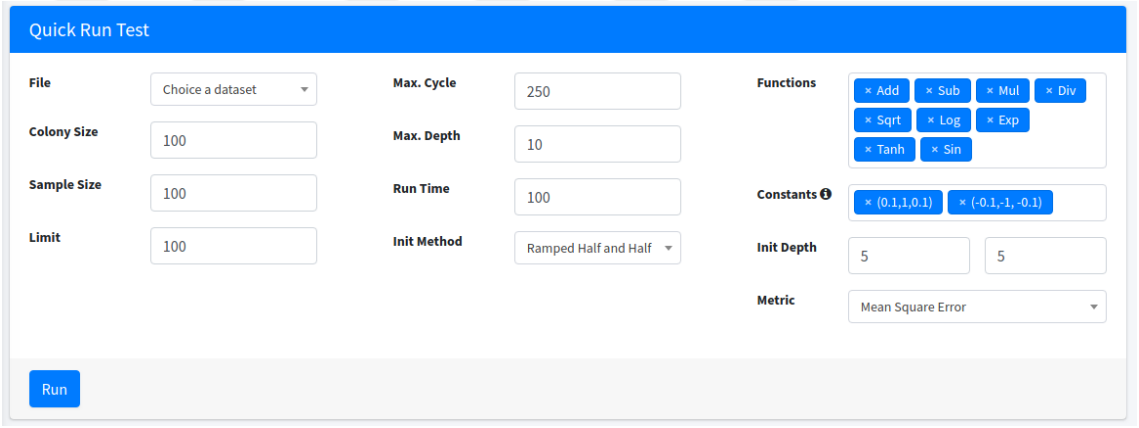
Şekil 3.7. ABCPWeb - Veri Seti ekleme arayüzü

Bunun dışında kullanıcı istediği takdirde açıklama kısmına veri setinde kaç kayıt/satır olduğunu da açıklama kısmına girebilir. Bu sayede kullanıcı, ABCP algoritmasını

koşmadan önce, uygunluk durumu sayısını, veri setindeki kayıt sayısına göre seçebilir. Yazılımın sonraki versiyonlarında, kullanıcının belirttiği uygunluk durumu sayısı ile kullandığı veri setindeki örnek sayısının kontrolünün otomatik olarak gerçekleştirilmesi planlanmaktadır.

3.4. ABCPWeb'in Çalıştırılması

Kullanıcı veri setini Bölüm 3.3 'te anlatıldığı gibi sisteme yükledikten sonra Şekil 3.1'deki arayüzde yer alan menüden "Run ABCP" seçeneğini seçerek Şekil 3.8'de gösterilen çalıştırma arayüzüne girer. Bu arayüz iki kısımdan oluşur: kontrol parametrelerinin girildiği alan ve daha önceki çalıştırmaların durumlarını gösteren listenin yer aldığı alan.



Şekil 3.8. ABCPWeb - Çalıştırma formu

Kontrol parametrelerinin girildiği alanda Şekil 3.8'de görüldüğü gibi bir form yer alır. Bu formda kullanıcıdan aşağıdaki bilgiler istenir:

- Daha önce yüklediği veri seti dosyası
- Yüklenen veri setinden kullanılacak örnek sayısı
- ABCP kontrol parametreleri (koloni büyüklüğü, limit, maksimum çevrim sayısı)
- Maksimum ağaç derinliği, başlangıç derinlik değerleri, popülasyon başlangıç metodu
- Bağımsız koşum sayısı, Metrik seçimi
- Fonksiyon kümesi ve sabitler

Veri seti seçimi yapılırken, listede kullanıcının veri setine verdiği isim görünür. Böylece kullanıcı veri setinin seçimini kolaylıkla yapar. Koloni büyüklüğü, örnek sayısı, limit, maksimum çevrim sayısı, maksimum derinlik, bağımsız koşma sayısı, başlangıç minimum ve maksimum derinlik değerleri numerik değerlerdir ve kullanıcı istediği değerleri verebilir. Ancak sonraki versiyonlarda özellikle bağımsız koşma sayısı ve çevrim sayısı gibi parametrelere, sistemin yoğunluğuna göre kontrollü bazı kısıtlamalar getirilecektir.

Kullanıcı, fonksiyon kümesi, popülasyon başlangıç metodu ve metrik parametrelerinin değerlerini, ancak sisteme yönetici tarafından girilen ön tanımlı seçeneklerden seçebilir. Metrik ve fonksiyon kümesinin sistem yöneticisi tarafından nasıl tanımlandığı ile ilgili bilgi Bölüm 3.2'de verilmiştir.

Yazılımın şimdiki versiyonunda bulunan ön tanımlı parametreler ve değerleri aşağıdaki listede gösterilmiştir.

Başlangıç metotları: Ramped half-and-half, Grow ve Full.

Metrikler: Mean Absolute, Sum Absolute Error, Mean Square ve Root Mean Square Error.

Fonksiyonlar: Tanh, Pow, Exp, Tan, Cos, Sin, Min, Max, Neg, Abs, Log, Sqrt, Div, Mul, Sub ve Add. Fonksiyonların açık yazılımları Tablo 3.1'de verilmiştir.

Formda yukarıda sayılan parametreler dışında, sabitlerin girilmesi için de bir parametre vardır. Bu parametre farklı kombinasyonlarda birden çok değer alabilir. Bu kombinasyonlar Tablo 3.2 'de belirtilmiştir.

Tablo 3.2'de görüldüğü gibi ABCPWeb, ağaçlar oluşturulurken, terminal yapraklarına üç farklı şekilde sabitleri yerleştirebilir. Tablonun ilk satırındaki notasyon, kullanıcının başlangıç ve bitiş noktası belli olan ve sabit aralıklarla artan/azalan sayıların girilmesini kolaylaştırır. Dördüncü satırdaki notasyon, ağaç oluştururken terminal yaprağına belirtilen aralıklarda rastgele bir sayı üreten operatörü yerleştirir. Bu operatörün olduğu her yaprak farklı rastsal sayılar içerir. Son olarak kullanıcı tek tek sabit sayılar girebilir. Bu durum da Tablo 3.2'nin 3. ve 5. satırlarında gösterilmiştir.

Tablo 3.1. Fonksiyonlar Tablosu

No.	Fonksiyon Kısaltması	Açık Yazılımı	Argüman sayısı
1	Tanh	Hiperbolik Tanjant	1
2	Pow	Üst alma	2
3	Exp	Eksponansiyel	1
4	Tan	Tanjant	1
5	Cos	Kosinüs	1
6	Sin	Sinüs	1
7	Min	Minimum	2
8	Max	Maksimum	2
9	Neg	Negatif	1
10	Abs	Mutlak	1
11	Log	Logaritma	1
12	Sqrt	Kare Kök	1
13	Div	Bölme	2
14	Mul	Çarpma	2
15	Sub	Çıkarma	2
16	Add	Toplama	2

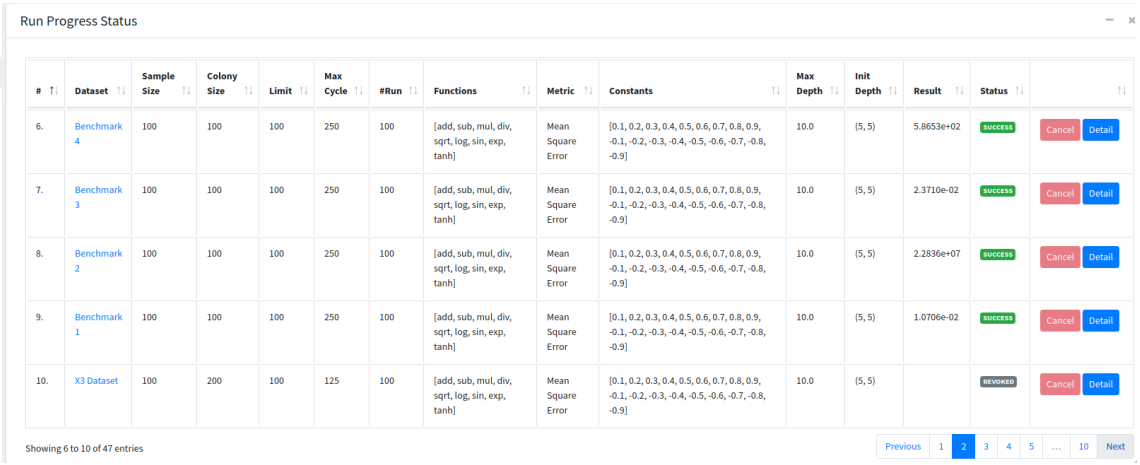
Kullanıcı parametre değerlerini girdikten sonra yine aynı formda yer alan Çalıştır butonuna tıklayarak çalıştırma işlemini gerçekleştirebilir. Bu işlem asenkron olarak başlar. Yani kullanıcının başka bir işlem yapmasını engellemez veya yazılım içindeki gezintisini kesintiye uğratmaz. Başlatılan işlem sayfanın alt bölmesinde yer alan çalıştırılmış işlemler listesine "Başladı" durum bilgisi ile eklenir.

Bilgilerin girildiği alanın altında ise Şekil 3.9'da gösterilen daha önce çalıştırılmış işlemlerin listesi yer alır. Bu listede bir işleme ait parametre bilgileri, durum bilgisi ve bitmiş işlemler için sonuç bilgisi yer alır. Ayrıca devam eden işlemler için iptal butonu, bitmiş işlemler için de Detay butonu bulunur.

Çalıştırılmış işlemlerin bulunduğu listede kontrol parametrelerin olmasının sebebi, kullanıcının işlemleri ayırt etmesi içindir. Bir kullanıcı bazen sadece bir parametreyi

Tablo 3.2. Sabitler parametre örnekleri

Örnek	Anlam	Çıktı
(0.1,0.5,0.1)	0.1 ’den 0.5 ’e kadar 0.1 aralıklarla	0.1, 0.2, 0.3, 0.4, 0.5
(-0.1, -0.5, -0.1)	-0.1’den -1 ’e kadar 0.1 aralıklarla	-0.1, -0.2, -0.3, -0.4, -0.5
(0.1, 0.5, 0.1), 1, 2, 3	1. satır + 1,2 ve 3 değerleri	0.1, 0.2, 0.3, 0.4, 0.5, 1, 2, 3
(0,1)	0 ile 1 arasında rastgele bir değer operatörü	0.24
(0,1),1,2,3	Rastgele bir sayı ve 1,2,3	0.14, 1, 2, 3



Şekil 3.9. ABCPWeb - Çalıştırılan İşlemler Listesi

değiştirerek algoritmayı çalıştırabilir. Bu işlemi diğerlerinden ayırt eden özelliği değiştirdiği parametredir. Kullanıcının bunu görebilmesi için tüm parametreler listede görünmektedir.

Bir işlemin durum bilgisi sistemde dört şekilde tanımlanmıştır:

- **Başladı:** İşlem başlatıldı ve devam ediyor.
- **İptal edildi:** İşlem kullanıcı tarafından iptal edildi.
- **Başarılı:** İşlem başarıyla, hatasız bir şekilde sonlandı.
- **Hatalı:** İşlem hata nedeniyle durduruldu.

Başlatılan bir işlemin iptal edilmesi için, bulunduğu satırda yer alan "İptal Et" butonu kullanılır. Kullanıcının bu butona yanlışlıkla tıklama ihtimaline karşın, tıklandıktan sonra, kullanıcının tekrar onayı bir kutucuk içinde sorulmaktadır. Kullanıcının iptal işlemini

onayladığı durumda, işlem iptal edilerek, işlem listesinde bulunduğu satırdaki durum bilgisi "İptal Edildi" olarak değiştirilir.

Bir işlemin detayını görüntüleyebilmek için ise "Detay" butonu kullanılır. Bu butona tıklandığında Şekil 3.10'daki gibi bir ekran gösterilir.

Başarıyla sonuçlanmış bir işlemin, listedeki ilgili satırında yer alan "Sonuç" alanında, bağımsız koşmalar sonucu elde edilmiş en iyi bireyin, seçilen metriğe göre hesaplanmış maliyet değeri yer alır.

Sayfanın kullanıcı dostu olması adına çalıştırılmış işlemler listesinde sayfalama yapılmıştır. Her bir sayfada 10 adet işlem listelenmektedir.

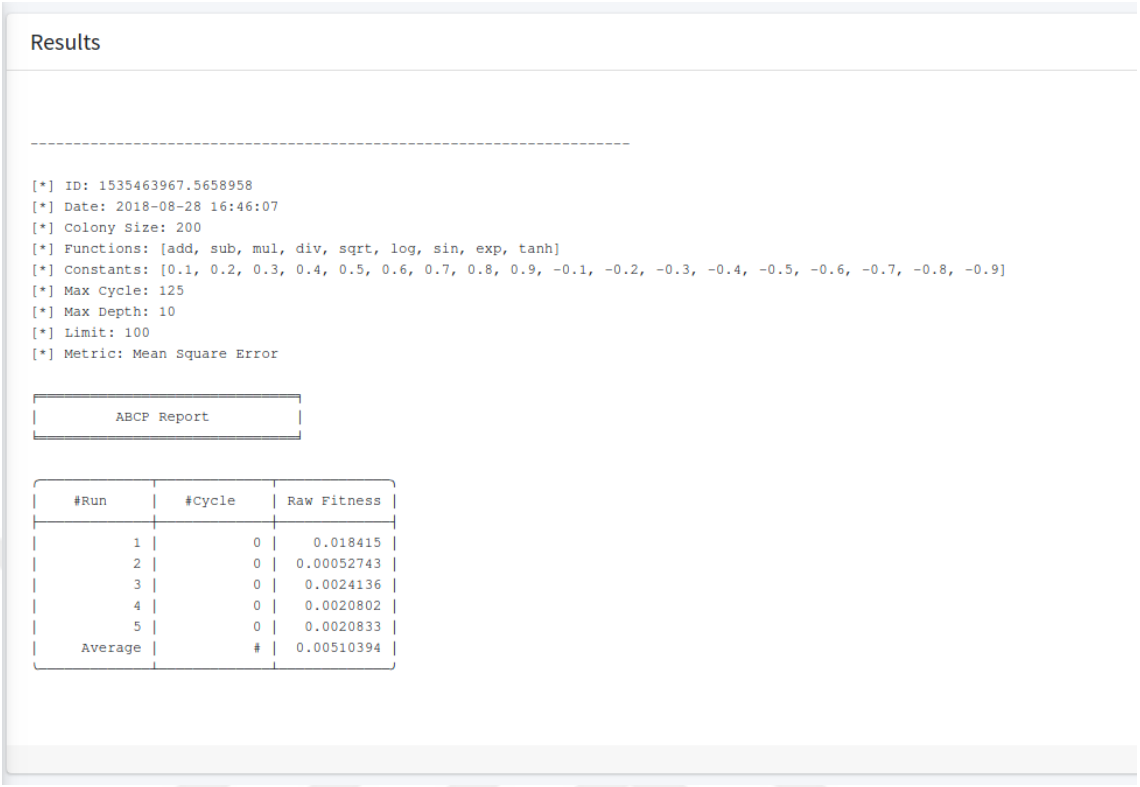
3.5. Sonuçların Alınması

Kullanıcı ABCP algoritmasını Bölüm 3.4'te anlatıldığı gibi çalıştırdıktan sonra, arka planda algoritma asenkron olarak çalışmaya başlar. Herhangi bir işlemin detayını görmek için listeden ilgili işlem satırında yer alan "Detay" butonuna tıkladığında Şekil 3.10'da gösterilen "İşlem Detay" sayfası açılır.

"İşlem Detay" sayfası iki kısımdan oluşur. İlk kısımda sonuçlar düz metin olarak yazdırılmaktadır. İkinci kısımda ise popülasyondaki en iyi birey ağaç olarak çizdirilmektedir. Şekil 3.10 'da ilk kısım gösterilmiştir.

İşlem başlatıldıktan sonra her bir bağımsız koşma sonunda koşma numarası, en iyi bireyin bulunduğu çevrim numarası ve en iyi bireyin uygunluk değeri kaydedilerek arayüzde tablo şeklinde gösterilir. Tablonun en altında tüm koşmaların ortalama uygunluk değeri gösterilir. Bunun dışında sonuç raporunda işlem numarası (ID), işlem zamanı ve işlem parametreleri de yer alır. Kullanıcıların her işlemi için sistemde o kullanıcıya özel oluşturulan dizinde kayıt dosyası oluşturulur. İşlemler asenkron çalıştığı için, ilk çevrim bittiği anda sonuçlar dosyaya kaydedildiğinden, detay sayfasında sonuçlar anlık olarak görüntülenir.

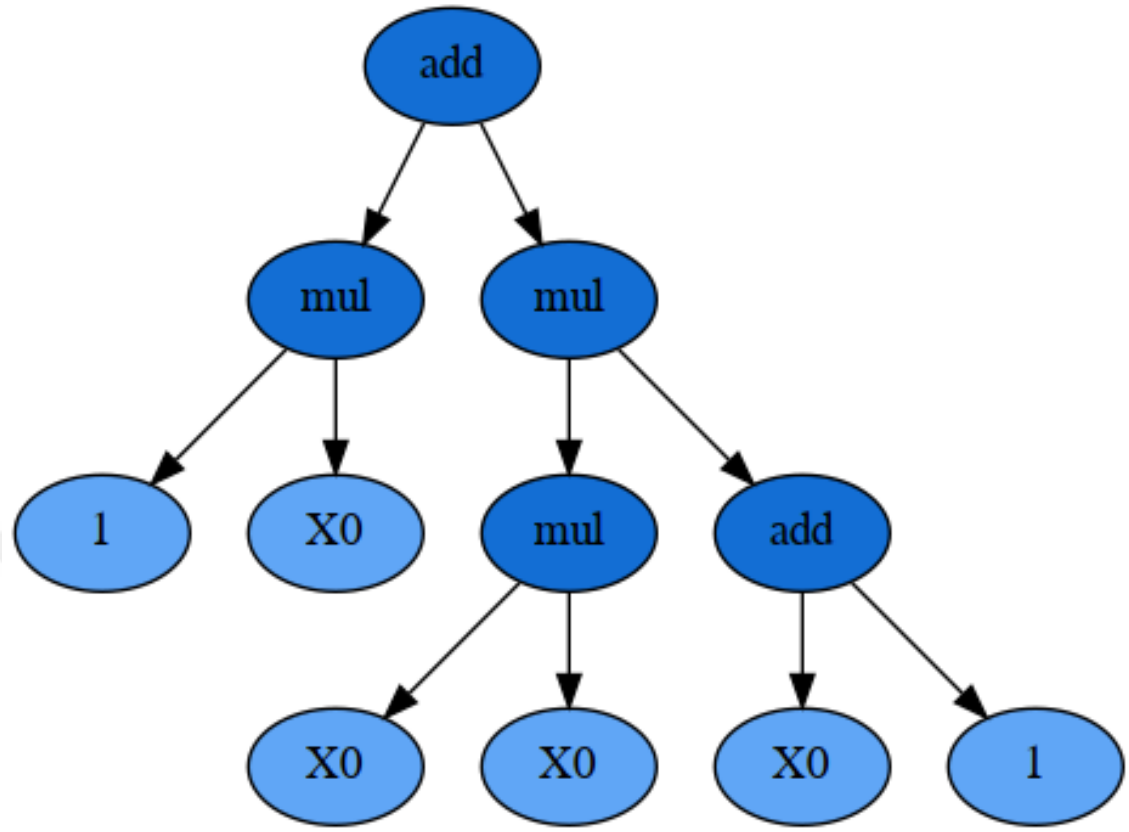
"İşlem detay" sayfasının ikinci kısmında ise bağımsız koşmalar sonucunda elde edilmiş en iyi birey Şekil 3.11'deki gibi ağaç olarak çizdirilir. Çizim, .png formatında resim dosyası



Şekil 3.10. ABCPWeb - İşlem Detay Örneği

olarak sistemde tutulur. Kullanıcı bu dosyayı sistemden indirip dilediği gibi kullanabilir.

Yazılımın sonraki versiyonları için "İşlem Detay" sayfasına sonuçların yorumlanmasını kolaylaştıracak çeşitli grafiklerin de yerleştirilmesi planlanmaktadır.



Şekil 3.11. ABCPWeb - En iyi birey ağaç örneği

3.6. Deneysel Çalışmalar

Bu tez kapsamında iki deney çalışması gerçekleştirilmiştir. Bunlardan ilkinde, ABCPWeb uygulamasının doğru çalışıp çalışmadığını test edilmiştir. İkinci deneyde ise, doğruluğu ilk deneyle sabitlenen ABCPWeb uygulaması kullanılarak, daha önce ABCP ile çözülmemiş problemler üzerine çalışılmıştır. Bu deneyler sonucunda ABCPWeb uygulamasının başarılı sonuçlar ürettiği gözlemlenmiştir. Aşağıda deneylere ilişkin detaylı açıklamalar yer almaktadır.

3.6.1. ABCPWeb'in Doğruluğunun Test Edilmesi

Bu deneyde, ABCPWeb'in performans ve başarım durumunu ölçmek için ABCP 'nin daha önce uygulandığı 10 adet sembolik regresyon problemi kullanılmıştır [1]. Problemler ve uygunluk değerlendirme durumları Tablo 3.3'te gösterilmektedir.

Problemlerin ABCPWeb uygulamasında çalıştırılması ile elde edilen sonuçlar, [1] 'de elde edilen sonuçlarla karşılaştırılmıştır. Deney sonucunda ABCPWeb uygulamasının başarılı sonuçlar ürettiği doğrulanmıştır.

Bu işlem yapılırken, referans alınan çalışmanın parametre değerleri ile aynı değerler kullanılmıştır. Kullanılan parametreler ve değerleri Tablo 3.4'te gösterilmektedir.

Tablo 3.3. Deneyde kullanılan problemler ve uygunluk değerlendirme durumları

Problem	Uygunluk Değerlendirme Durumu
$F_1 = x^3 + x^2 + x$	20 rastsal nokta $\subseteq [-1, 1]$
$F_2 = x^4 + x^3 + x^2 + x$	20 rastsal nokta $\subseteq [-1, 1]$
$F_3 = x^5 + x^4 + x^3 + x^2 + x$	20 rastsal nokta $\subseteq [-1, 1]$
$F_4 = x^6 + x^5 + x^4 + x^3 + x^2 + x$	20 rastsal nokta $\subseteq [-1, 1]$
$F_5 = sin(x^2)cos(x) - 1$	20 rastsal nokta $\subseteq [-1, 1]$
$F_6 = sin(x) + sin(x + x^2)$	20 rastsal nokta $\subseteq [-1, 1]$
$F_7 = log(x + 1) + log(x^2 + 1)$	20 rastsal nokta $\subseteq [0, 2]$
$F_8 = \sqrt{x}$	20 rastsal nokta $\subseteq [0, 4]$
$F_9 = sin(x) + sin(y^2)$	100 rastsal nokta $\subseteq [-1, 1] \times [-1, 1]$
$F_{10} = 2sin(x)cos(y)$	100 rastsal nokta $\subseteq [-1, 1] \times [-1, 1]$

Fonksiyon kümesinde kullanılan bölme ve logaritma işlemleri korunmuş olarak kullanılmıştır. Yani bu fonksiyonlarda, sonucun sonsuz veya tanımsız olduğu durumlar elemine edilmiştir. Mesela bölme işleminde bölen, sıfıra eşitse işlem sonucu "1" olarak değerlendirilmiştir. Benzer durum logaritma fonksiyonu için de geçerlidir.

Deneyde diğer parametrelerde olduğu gibi kontrol parametreleri de [1]'de kullanılan parametrelerle aynıdır. Yani, koloni büyüklüğü ve limit parametrelerinin değerleri 500 olacak şekilde 100 bağımsız çalıştırma yapılmıştır. Bu çalıştırmaların sonuçları Tablo 3.5'te, referans alınan çalışma çıktıları ile birlikte gösterilmektedir.

Tablo 3.5'te gösterilen sonuçlara göre ABCPWeb uygulamasının, çalıştırılan 10 fonksiyon için başarım durumu, referans alınan çalışmayla yaklaşık olarak aynıdır. Hatta bazı fonksiyonlarda daha iyi sonuçların elde edildiği görülmektedir. Tabloda, referans

Tablo 3.4. Deney parametreleri

Parametre	Değer
Başlangıç maksimum derinlik	6
Maksimum derinlik	15
Fonksiyon kümesi	$+, -, \times, \div, sin, cos, exp, rlog$
Terminal kümesi	$X, 1$
Değerlendirme	Eşitlik 2.5 - Mutlak hata toplamı
Koşma sayısı	100

deneyden daha iyi olan değerler kalın puntolarla gösterilmiştir.

Tablo 3.5. Deney Sonuçları - Ortalama Maliyet Değerleri

Yöntem	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
ABCP [1]	0.01	0.05	0.07	0.10	0.05	0.02	0.06	0.10	0.47	1.06
ABCPWeb	0.02	**0.03**	0.10	**0.06**	**0.02**	0.05	**0.04**	**0.06**	0.5	**0.3**

Tablo 3.6'da test edilen problemler için ABCPWeb ile elde edilen eşitlikler gösterilmektedir. Bu tabloya göre ABCPWeb istenilen sonuçları başarıyla elde etmiştir. Referans alınan çalışma ile kıyaslandığında, diğerleri aynı olmakla birlikte F_4 ve F_7 fonksiyonları için daha net sonuçlar alınmıştır.

Tablo 3.6. Test fonksiyonları ve ABCPWeb 'in ürettiği fonksiyonlar

Problem	Hedef Eşitlik	Üretilen Eşitlik
F_1	$x^3 + x^2 + x$	$x^3 + x^2 + x$
F_2	$x^4 + x^3 + x^2 + x$	$x^4 + x^3 + x^2 + x$
F_3	$x^5 + x^4 + x^3 + x^2 + x$	$x^5 + x^4 + x^3 + x^2 + x$
F_4	$x^6 + x^5 + x^4 + x^3 + x^2 + x$	$x^6 + x^5 + x^4 + x^3 + x^2 + x$
F_5	$sin(x^2)cos(x) - 1$	$sin(x^2)cos(x) - 1$
F_6	$sin(x) + sin(x + x^2)$	$sin(x) + sin(x + x^2)$
F_7	$log(x + 1) + log(x^2 + 1)$	$log(x^3 + x^2 + x + 1)$
F_8	$\sqrt{x}$	$(x/\sqrt{x}) - ((ln(ln2) + (x/e^{\sqrt{x}}))/(2e^2 + x + e^{e^3}))$
F_9	$sin(x) + sin(y^2)$	$sin(x) + sin(y^2)$
F_{10}	$2sin(x)cos(y)$	$2sin(x)cos(y)$

3.6.2. ABCPWeb Kullanarak ABCP'nin Yeni Problemlere Uygulanması

Bu deneyde, ABCP'nin daha önce hiç uygulanmadığı 10 adet yeni problem üzerinde, ABCPWeb kullanılarak, ABCP algoritması çalıştırılmıştır. Tablo 3.7'de gösterilen problemler, daha önce GP ile çalıştırılan [46]'dan seçilmiştir.

Deneyde referans alınan çalışmadaki parametre değerleri kullanılmıştır. Bu parametreler ve değerleri Tablo 3.8'de gösterilmiştir. Maksimum çevrim sayısı ve koloni büyüklüğü parametrelerine, referans alınan çalışmadaki toplam değerlendirme sayısına yakın olabilecek değerler verilmiştir.

Deney sonucunda, ABCPWeb üzerinde çalıştırılan ABCP algoritmasının 6 problemde ($F_{14}, F_{15}, F_{17}, F_{18}, F_{19}, F_{20}$) GP'den daha başarılı, diğer problemlerde ise GP'ye yakın sonuçlar ürettiği gözlemlenmiştir. Deney sonuçları Tablo 3.9'da gösterilmiştir.

Bu deneyle birlikte ABCP, 10 yeni probleme daha uygulanarak, sembolik regresyon problemlerinin çözümündeki başarısını ortaya koymuştur.

Tablo 3.7. Deneyde kullanılan problemler ve uygunluk değerlendirme durumları

Fonksiyon	Eşitlik
F_{11}	$f(x_1, x_2) = \frac{e^{-(x_1-1)^2}}{1.2+(x_2-2.5)^2}$
F_{12}	$f(x_1, x_2) = e^{-x_1}x_1^3 cos(x_1)sin(x_1)(cos(x_1)sin^2 x_1 - 1)(x_2 - 5)$
F_{13}	$f(x_1, x_2, x_3, x_4, x_5) = \frac{10}{5+\sum_{i=1}^{5}(x_i-3)^2}$
F_{14}	$f(x_1, x_2) = \frac{(x_1-3)^4+(x_2-3)^3-(x_2-3)}{(x_2-2)^4+10}$
F_{15}	$f(x_1, x_2) = \frac{1}{1+x_1^{-4}} + \frac{1}{1+x_2^{-4}}$
F_{16}	$f(x_1, x_2) = x_1^4 - x_1^3 + x_2^2/2 - x_2$
F_{17}	$f(x_1, x_2) = \frac{8}{2+x_1^2+x_2^2}$
F_{18}	$f(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}) = x_1x_2 + x_3x_4 + x_5x_6 + x_1x_7x_9 + x_3x_6x_{10}$
F_{19}	$f(x_1, x_2, x_3, x_4, x_5) = x_1x_2x_3x_4x_5$
F_{20}	$f(x_1, x_2, x_3, x_4, x_5) = 2 - 2.1cos(9.8x_1)sin(1.3x_5)$

Tablo 3.8. Deney parametreleri

Parametre	Değer
Başlangıç maks. derinlik	5
Maksimum derinlik	10
Fonksiyon kümesi	$+, -, \times, \div, sin, tanh, exp, rlog, sqrt$
Terminal kümesi	0.1 adımlarla $[-0.9..0.9] \cup [0.1..0.9]$
Değerlendirme	Ortalama karesel hata (MSE)
Bağımsız koşma sayısı	100
Popülasyon başlangıç metodu	Ramped half-and-half
Tüm fonksiyonlar için giriş değer aralığı	100 rastsal nokta $\subseteq [-5, 5]$
Limit	100
Koloni Büyüklüğü	250
Maksimum çevrim sayısı	100

Tablo 3.9. Deney Sonuçları - Ortalama Karesel Hata Değerleri (MSE)

Yöntem	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}	F_{16}	F_{17}	F_{18}	F_{19}	F_{20}
GP [46]	0.007	$2.16E7$	0.012	$2.20E3$	0.078	$4.21E2$	2.372	$4.470E3$	$6.71E4$	1.204
ABCPWeb	0.01	$2.48E7$	0.024	**1.71E3**	**0.049**	$4.97E2$	**0.137**	**0.826E3**	**2.67E4**	**0.982**

4. BÖLÜM

TARTIŞMA, SONUÇ ve ÖNERİLER

4.1. Tartışma, Sonuç ve Öneriler

Bu çalışmada, ABCP 'nin kolayca kullanımına imkan tanıyan ABCPWeb yazılımı geliştirilip, doğruluğu test edilmiştir. Ayrıca ABCP'nin daha önce uygulanmadığı yeni problemlere ABCPWeb kullanılarak ABCP uygulanmıştır.

Deneyler sonucunda, ABCPWeb yazılımı, "**www.abcpweb.com**" alan adı ile kullanıcıların hizmetine açılmıştır. Üyelik sistemi olduğu için, ilk versiyon için hesap açma işlemi sadece yönetici tarafından gerçekleştirilebilmektedir. Bu nedenle, ABCPWeb kullanmak isteyen kullanıcıların yönetici ile iletişime geçmesi gerekmektedir.

Bu tezde bahsedilen ABCPWeb yazılımı, yazılımın ilk versiyonudur. Dolayısıyla yazılımın en ilkel hali anlatılmıştır. Sonraki versiyonlarda ABCPWeb yazılımında yapılacak geliştirmeler listesi aşağıda verilmiştir.

- Kullanıcılara yeni metrik ve fonksiyon ekleme yetisini kazandırmak. Bu özellikle birlikte kullanıcılar fonksiyon kümesine diledikleri fonksiyonları oluşturarak ekleyebilirler. Aynı durum metrik için de geçerlidir.
- Sistem kaynaklarını verimli kullanmak için kısıtlamalar getirilmesi. İlk versiyonda kısıtlamalar olmadığından kullanıcılar, diledikleri verileri kullanarak diledikleri kadar çalıştırma yapabilirler. Bu durumda, kaynaklar yetersiz kalarak sistem yanıt vermeyi durdurabilir. Bunu önlemek için kullanıcıların yüklediği veri setleri için boyut kısıtlamasının yanında bağımsız çalıştırma gibi iteratif parametreler için de kısıtlamalar getirilmelidir. Kaynaklar arttıkça bu kısıtlar esnetilebilecektir.

- Bağımsız çalışma alanları (work space) oluşturabilme imkanı sağlanması. Araştırmacılar genelde aynı parametrelerle, farklı veri setleri üzerinde birden çok çalıştırma işlemi yapabilirler. Bu durumda kullanıcının her işlemin başında parametreleri yeniden ayarlamak yerine bir kere ayarlayıp çalışma alanına kaydetmesi ideal olandır. Bu nedenle, kullanıcının, parametre değerlerini bir kere ayarlayıp kaydedebileceği çalışma alanlarının olması sağlanacak ve istediği zaman bu çalışma alanlarını seçerek işlemlerini gerçekleştirebilecektir.

- Tanıtım arayüzünün geliştirilmesi. Yazılım işlevsel olarak amacına uygun çalışabilecek durumdadır ancak kullanıcılara bu yazılımı tanıtacak arayüzü bulunmamaktadır. Sonraki versiyonlarda, kullanıcıları bir giriş sayfasından ziyade yazılımı tanıtan bir sayfanın karşılaması için gerekli geliştirmelerin yapılması planlanmaktadır.

- İşlemlerin sonuçlarını yorumlamayı kolaylaştıracak, gerekli grafiklerin eklenmesi. Çalıştırmalar sonucunda üretilen ağaç çiziminin yanında, yakınsama grafikleri gibi yorumlamayı kolaylaştıracak grafiklerin de yazılıma eklenmesi planlanmaktadır.

KAYNAKLAR

1. Karaboga, D., Ozturk, C., Karaboga, N., and Gorkemli, B., 2012. Artificial bee colony programming for symbolic regression, **Information Sciences**, **209**:1–15.

2. Weise, T., 2009. Global Optimization Algorithms - Theory and Application, Self-Published, second edition, online available at http://www.it-weise.de/.

3. Atmar, W., 1994. Notes on the simulation of evolution, **IEEE Transactions on Neural Networks**, **5**(1):130–147.

4. Back, T., 1996. Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms, Oxford university press.

5. Goldberg, D.E., Lingle, R., et al., 1985. Alleles, loci, and the traveling salesman problem, *Proceedings of an international conference on genetic algorithms and their applications*, volume 154, 154–159, Lawrence Erlbaum, Hillsdale, NJ.

6. Fogel, D.B., 1988. An evolutionary approach to the traveling salesman problem, **Biological Cybernetics**, **60**(2):139–144.

7. Oliver, I.M., Smith, D.J., and Holland, J.R.C., 1987. A Study of Permutation Crossover Operators on the Traveling Salesman Problem, *Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application*, 224–230, L. Erlbaum Associates Inc., Hillsdale, NJ, USA.

8. Fogel, D.B., 1990. A parallel processing approach to a multiple travelling salesman problem using evolutionary programming, *Proceedings of the fourth annual symposium on parallel processing*, 318–326, Fullerton, CA.

9. Blaton, J., 1993. Multiple vehicle routing with time and capacity constraints using genetic algorithms, *Proc. of the 5th. International Conference On Genetic, Algorithms*, 452–459.

10. Thamgiah, S., Vinayagamoorthy, R., and Gubbi, A., 1993. Vehicle routing with time deadlines using genetic and local algorithm, *Proc. of The Fifth Int. Conf. on Genetic Algorithms*, 506–513.

11. Michalewicz, Z., 1993. A hierarchy of evolution programs: An experimental study, **Evolutionary Computation**, **1**(1):51–76.

12. Davis, L., 1985. Job shop scheduling with genetic algorithms, *J.J. Grefenstette, Ed:, Proceedings of an international conference on genetic algorithms and their applications*, volume 140, 136–150, Carnegie-Mellon University Pittsburgh, Pennsylvania.

13. Biegel, J.E. and Davern, J.J., 1990. Genetic algorithms and job shop scheduling, **Computers & Industrial Engineering**, **19**(1-4):81–91.

14. Syswerda, G., 1991. Scheduling optimization using genetic algorithms, L. Davis, Ed:, In: Handbook of genetic algorithms, 332–349, Van Nostrand Reinhold, New York.

15. Yamada, T. and Nakano, R., 1992. A Genetic Algorithm Applicable to Large-Scale Job-Shop Problems., *PPSN*, volume 2, 281–290.

16. Corne, D., Ross, P., and Fang, H.L., 1994. Fast practical evolutionary timetabling, *AISB Workshop on Evolutionary Computing*, 250–263, Springer.

17. Ling, S.E., 1992. Integrating genetic algorithms with a Prolog assignment problems as a hybrid solution for a polytechnic timetable problem, **Parallel Problem Solving from Nature**, **2**:321–329.

18. Smith, D., 1985. Bin packing with adaptive search, *Proc. of the 1st International Conference on Genetic Algorithms and Their Applications, Pittsburgh PA*, 202–207.

19. Fujita, K., Akagi, S., and Hirokawa, N., 1993. Hybrid approach for optimal nesting using a genetic algorithm and a local minimization algorithm, *Proceedings of the 19th annual ASME design automation conference*, volume 1, 477–484, ASME.

20. Juliff, K., 1993. A multi-chromosome genetic algorithm for pallet loading, *Proceedings of the 5th international conference on genetic algorithms*, 467–473, Morgan Kaufmann Publishers Inc.

21. Etter, D., Hicks, M., and Cho, K., 1982. Recursive adaptive filter design using an adaptive genetic algorithm, *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'82.*, volume 7, 635–638, IEEE.

22. Martin, R.S., 1993. Genetic algorithms for optimization of integrated circuits synthesis, *Proc. of the Fifth International Conference on Genetic Algorithms, Urbana-Champaign, USA*, 432–438.

23. Louis, S.J. and Rawlins, G.J., 1991. Designer Genetic Algorithms: Genetic Algorithms in Structure Design., *ICGA*, 53–60.

24. Smith, A.E., 1993. Genetic optimization using a penalty function, *Proc. Fifth Int. Conf. on Genetic Algorithms*, 499–505.

25. Miller, G.F., Todd, P.M., and Hegde, S.U., 1989. Designing Neural Networks using Genetic Algorithms., *ICGA*, volume 89, 379–384.

26. Polani, D. and Uthmann, T., 1992. Adaptation of Kohonen feature map topologies by genetic algorithms, *In: Parallel Problem Solving-Nature Conf procs (PPSN 2)*, Elsevier.

27. Lucasius, C.B., Blommers, M.J., Buydens, L.M., and Kateman, G., 1991. A genetic algorithm for conformational analysis of dna, L. Davis, Ed:, In: Handbook of genetic algorithms, 251–281, Van Nostrand Reinhold, New York.

28. Janikow, C.Z. and Cai, H., 1992. A Genetic Algorithm Application in Nonparametric Functional Estimation., *PPSN*, 251–260.

29. Fonseca, C.M., Fleming, P.J., et al., 1993. Genetic Algorithms for Multiobjective Optimization: FormulationDiscussion and Generalization., *Icga*, volume 93, 416–423.

30. Krishnakumar, K., 1990. Genetic algorithms in control system optimization., *AIAA Guidance, Navigation, Control Conf.*, 1568–1577.

31. Handley, S., 1993. Automated learning of a detector for alpha-helices in protein sequences via genetic programming, *Proceedings of the 5th International Conference on Genetic Algorithms*, 271–278, Morgan Kaufmann Publishers Inc.

32. Bala, J. and Wechsler, H., 1993. Learning to detect targets using scale-space and genetic search, *Proceedings of the 5th International Conference on Genetic Algorithms*, 516–522, Morgan Kaufmann Publishers Inc.

33. Holland, J.H., 1975, Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.

34. Goldberg, D.E. and Holland, J.H., 1988. Genetic algorithms and machine learning, **Machine Learning**, **3**(2/3):95–99.

35. Davis, L., 1987. Genetic Algorithms and Simulated Annealing, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

36. Davis, L., 1991. Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York.

37. Davidor, Y., 1991. Genetic Algorithms and Robotics: A heuristic strategy for optimization, volume 1, World Scientific Publishing Company.

38. Koza, J.R., 1994. Genetic programming as a means for programming computers by natural selection, **Statistics and computing**, **4**(2):87–112.

39. Spector, L., Barnum, H., and Bernstein, H.J., 1998. Genetic Programming for Quantum Computers, *J.R. Koza, W. Banzhaf, K. Chellapilla, K. Deb, M. Dorigo, D.B. Fogel, M.H. Garzon, D.E. Goldberg, H. Iba, and R. Riolo, Eds:, Genetic Programming 1998: Proceedings of the Third Annual Conference*, 365–373, Morgan Kaufmann, University of Wisconsin, Madison, Wisconsin, USA.

40. Lohn, J., Hornby, G., and Linden, D., 2004. Evolutionary Antenna Design for a NASA Spacecraft, U.M. O'Reilly, T. Yu, R.L. Riolo, and B. Worzel, Eds:, Genetic Programming Theory and Practice II, chapter 18, 301–315, Springer, Ann Arbor.

41. Cramer, N.L., 1985. A representation for the adaptive generation of simple sequential programs, *Proceedings of the first international conference on genetic algorithms*, 183–187.

42. Koza, J.R., 1992. Genetic programming II, automatic discovery of reusable subprograms, MIT Press, Cambridge, MA.

43. Koza, J.R., 1989. Hierarchical Genetic Algorithms Operating on Populations of Computer Programs., *IJCAI*, volume 89, 768–774.

44. Karaboga, D., 2005. An idea based on honey bee swarm for numerical optimization, Technical report, Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.

45. Karaboga, D. and Basturk, B., 2007. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, **Journal of global optimization**, **39**(3):459–471.

46. Nicolau, M., Agapitos, A., O'Neill, M., and Brabazon, A., 2015. Guidelines for defining benchmark problems in genetic programming, *Evolutionary Computation (CEC), 2015 IEEE Congress on*, 1152–1159, IEEE.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı, Soyadı	:	Ceylan BOZOĞULLARINDAN
Uyruğu	:	Türkiye (T.C.)
Doğum Tarihi ve Yeri	:	
Telefon	:	0 352 207 66 66
Belgegeçer	:	0 352 437 57 84
E-posta	:	cboz@erciyes.edu.tr
Adres	:	Erciyes Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölüm Başkanlığı 38039, Melikgazi KAYSERİ TÜRKİYE

EĞİTİM

Derece	**Kurum**	**Mez.Yılı**
Lisans	ERÜ Müh. Fakültesi Bilgisayar Müh., KAYSERİ	2016

İŞ DENEYİMİ

Yıl	**Kurum**	**Görev**
2018-Halen	ERÜ Mühendislik Fakültesi Bilgisayar Müh. Bölümü, KAYSERİ	Araştırma Görevlisi
2017-2018	Kayseri Büyükşehir Belediyesi Coğrafi Bilgi Sistemleri Müdürlüğü	Yazılım Geliştirici
2016-2017	İndis Yazılım, KAYSERİ	Yazılım Geliştirici