

**KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

YÜKSEK LİSANS TEZİ

**SEZGİSEL YÖNTEMLER KULLANARAK AKILLI SORU
BANKASI SİSTEMİ TASARIMI**

SAMET DİRİ

KOCAELİ 2018

KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

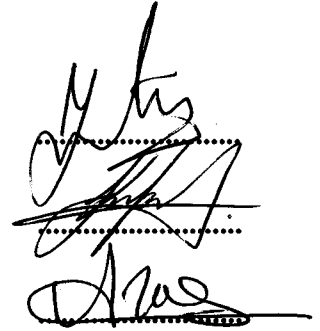
BİLGİSAYAR MÜHENDİSLİĞİ
ANABİLİM DALI

YÜKSEK LİSANS TEZİ

SEZGİSEL YÖNTEMLER KULLANARAK AKILLI SORU
BANKASI SİSTEMİ TASARIMI

SAMET DİRİ

Prof.Dr. Mehmet YILDIRIM
Danışman, Kocaeli Üniversitesi
Doç.Dr. Kerem KÜÇÜK
Jüri Üyesi, Kocaeli Üniversitesi
Dr.Öğr.Üyesi Adem TUNCER
Jüri Üyesi, Yalova Üniversitesi



Tezin Savunulduğu Tarih: 02.07.2018

ÖNSÖZ VE TEŞEKKÜR

Bu çalışmada genetik algoritma ile yerinde sınav uygulamalarında kullanılmak üzere akıllı soru bankası sistemi tasarlanmaya çalışılmıştır. Sonuç olarak akıllı soru bankası sistemi yardımıyla yerinde uygulanan çoktan seçmeli sınav süreçleri daha etkili yönetilecektir.

Yüksek lisans eğitimim boyunca değerli bilgilerini benimle paylaşmış, bana yol gösteren, yoğun çalışma temposunda zamanını her türlü problemimi çözmeye ayıran ve tez yazımında desteklerini esirgemeyen saygıdeğer hocam Prof. Dr. Mehmet Yıldırım'a çok teşekkür ederim. Çalışmam boyunca her ihtiyaç duyduğum anda desteğini hep hissettiğim hocam Arş.Gör.Dr. Selçuk Öğütçü'ye, tez sürecinde yardımlarını esirgemeyen değerli arkadaşım Özgür Barış Kayapınar'a, değerli çalışma arkadaşlarıma, annem Ayşe Diri, babam Nebi Diri, kardeşlerim Semih Diri ve Seda Diri'ye teşekkürlerimi sunarım.

Ayrıca tanıştığımız günden tezimi tamamladığım ana dek çalışmamın her aşamasında bana destek olan değerli eşim Elif Ayşe Diri'ye ve kızım Zeynep İdil'e teşekkür ederim.

Temmuz – 2018

Samet DİRİ

İÇİNDEKİLER

ÖNSÖZ VE TEŞEKKÜR	i
İÇİNDEKİLER	ii
ŞEKİLLER DİZİNİ.....	iv
TABLolar DİZİNİ	v
SİMGELER DİZİNİ VE KISALTMALAR	vi
ÖZET.....	vii
ABSTRACT.....	viii
GİRİŞ	1
1. EĞİTİM VE ÖLÇME.....	5
1.1. Eğitim	5
1.2. Ölçme ve Önemi.....	6
1.3. Ölçme Yöntemleri	6
1.3.1. Açık uçlu sorular	6
1.3.2. Kısa cevaplı sorular	7
1.3.3. Çoktan seçmeli sorular.....	7
1.4. Otomatik Sınav	9
2. SEZGİSEL YÖNTEMLER.....	11
2.1. Genetik Algoritmalar.....	11
2.1.1. Biyolojik arka planı	11
2.1.2. Genetik algoritma ve özellikleri	12
2.2. GA'yı Oluşturan Operatörler	14
2.2.1. Seçilim	14
2.2.1.1. Rulet tekerleği seçilimi	15
2.2.1.2. Rastgele olmayan örnekleme seçilimi	15
2.2.1.3. Doğrusal sıralama seçilimi.....	15
2.2.1.4. Seçincilik yöntemi	16
2.2.2. Çaprazlama	16
2.2.2.1. Parçalı eşleme çaprazlaması	16
2.2.2.2. Sıralı çaprazlama.....	17
2.2.2.3. Döngüsel çaprazlama.....	18
2.2.2.4. Kenar birleştirmeli çaprazlama.....	19
2.2.3. Mutasyon	20
3. AKILLI SORU BANKASI SİSTEMİ.....	22
3.1. Uygulama Geliştirme Ortamı	22
3.1.1. Adobe coldfusion	22
3.1.2. MSSQL	23
3.1.3. HTML	23
3.1.4. Javascript	24
3.1.5. JSON	25
3.2. Uygulama Alt Yapısı ve Ara Yüzü	26
3.2.1. Veri tabanı tasarımı.....	26
3.2.1.1. “Unvan” tablosu.....	28
3.2.1.2. “Kullanıcı” tablosu	28

3.2.1.3. “Konular” tablosu	29
3.2.1.4. “Soru” tablosu	29
3.2.1.5. “Cevaplar” tablosu	30
3.2.1.6. “Sınav” tablosu	31
3.2.1.7. “SınavSorulari” tablosu	33
3.2.1.8. “OgrenciCevaplari” tablosu	34
3.2.2. Sınav öncesi yapılması gereken işlemler	34
3.2.2.1. Kullanıcı kaydı oluşturma	35
3.2.2.2. Oturum açma ve ana giriş ekranı	37
3.2.2.3. Konu tanımlama	38
3.2.2.4. Konu listeleme	39
3.2.2.5. Konu güncelleme	41
3.2.2.6. Konu silme	41
3.2.2.7. Soru tanımlama	41
3.2.2.8. Soru listeleme	45
3.2.2.9. Soru güncelleme	47
3.2.2.10.Soru silme	48
3.2.2.11.Sınav tanımlama	48
3.2.2.12.Sınav listeleme	58
3.2.2.13.Kitapçık listeleme	60
3.2.2.14.Kitapçık görüntüleme	61
3.2.2.15.Sınav değerlendirme	62
3.2.2.16.Sınav sonuç listeleme	64
3.2.2.17.Sınav sorularının analizi	64
3.2.2.18.Sınav kopyalama	65
3.2.2.19.Sınav silme	66
4. SONUÇLAR VE ÖNERİLER	67
KAYNAKLAR	69
EKLER	72
KİŞİSEL YAYIN VE ESERLER	103
ÖZGEÇMİŞ	104

ŞEKİLLER DİZİNİ

Şekil 2.1. Genetik algoritma genel akış şeması	13
Şekil 2.2. Parçalı eşleme çaprazlaması örneği	17
Şekil 2.3. Parçalı eşleme çaprazlaması sonrası yer değiştirme örneği	17
Şekil 2.4. Sıralı çaprazlaması için çaprazlama ve sıralama örneği	18
Şekil 2.5. Döngüsel çaprazlama örneği	18
Şekil 3.1. HTML kodu örneği	24
Şekil 3.2. Örnek HTML sayfası çıktısı	24
Şekil 3.3. XML veri formatı örneği	25
Şekil 3.4. JSON veri formatı örneği	26
Şekil 3.5. ASBS veri tabanı şablonu	27
Şekil 3.6. Kullanıcı oturum açma ekranı	35
Şekil 3.7. Yeni kullanıcı kaydı oluşturma ekranı	35
Şekil 3.8. Yeni kullanıcı kaydı tamamlandı ekranı	36
Şekil 3.9. E-posta adresi sistemde kayıtlı ekranı	36
Şekil 3.10. Kullanıcı kaydı hata mesajı	36
Şekil 3.11. Kullanıcı ana giriş ekranı	37
Şekil 3.12. Konu tanımlama ekranı	38
Şekil 3.13. Kullanıcının tanımlı konularını listeleme ekranı	40
Şekil 3.14. Tanımlı konu düzenleme ekranı	41
Şekil 3.15. Yeni soru giriş ekranı	42
Şekil 3.16. Tanımlı soruları listeleme ekranı	46
Şekil 3.17. Soru düzenleme ekranı	47
Şekil 3.18. Sınav oluşturma ekranı	48
Şekil 3.19. Genetik algoritma gen, kromozom, nüfus ve tekrarlama yapıları	51
Şekil 3.20. Uygulamada geliştirilen genetik algoritmanın akış diyagramı	52
Şekil 3.21. Rulet tekerleği yöntemi	56
Şekil 3.22. Tanımlı sınav bilgileri listeleme ekranı	60
Şekil 3.23. Sınav kitapçıklarının listelenmesi	61
Şekil 3.24. Kitapçık kapağı örneği	62
Şekil 3.25. Sınav değerlendirme ekranı	63
Şekil 3.26. Sınav sonuç listesi	64
Şekil 3.27. Sınav sorusu analizi	65
Şekil 3.28. Sınav kopyalama ekranı	66

TABLolar DİZİNİ

Tablo 2.1. Kenar birleřtirmeli aprazlama komřuluk listesi	19
Tablo 3.1. Zorluk indeksinin kullanıcı dostu leęi	44
Tablo 3.2. Ayırt edicilik indeksinin kullanıcı dostu leęi	45



SİMGELER DİZİNİ VE KISALTMALAR

Kısaltmalar

AJAX	: Asynchronous JavaScript and XML (Eşzamansız JavaScript ve XML)
ASBS	: Akıllı Soru Bankası Sistemi
CF	: Adobe ColdFusion
CFML	: ColdFusion Markup Language (ColdFusion İşaretleme Dili)
DML	: Veri İşleme Dili
DNA	: Deoksiribo Nükleik Asit
GA	: Genetik Algoritma
HTML	: Hyper Text Markup Language (Köprü Biçimlendirme Dili)
JS	: JavaScript
JSON	: JavaScript Object Notation (JavaScript Nesne Gösterimi)
MSSQL	: Microsoft Structured Query Language (Microsoft Yapılandırılmış Sorgulama Dili)
SQL	: Structured Query Language (Yapılandırılmış Sorgulama Dili)
ÖYS	: Öğrenme Yönetim Sistemi
XML	: Extensible Markup Language (Genişletilebilir İşaretleme Dili)
VTYS	: Veri Tabanı Yönetim Sistemi

SEZGİSEL YÖNTEMLER KULLANARAK AKILLI SORU BANKASI SİSTEMİ TASARIMI

ÖZET

Gelişen bilişim teknolojileri sayesinde zaman kavramı, hesaplamaların hızı ve doğruluğu önem arz etmektedir. Günümüzde insanların eğitim alma ve aldıkları eğitimin hedeflere uygun olduğunu belgeleme gereksinimleri de artmaktadır. Eğitimin hedeflenen niteliklere uygunluğu ve başarı düzeyi ancak kaliteli sorulardan oluşan sınavlar ile ölçülebilmektedir. Kaliteli soruların geleneksel yöntemler ile oluşturulması ve belirlenmesi, teknolojinin geldiği nokta düşünüldüğünde, çok yavaş ve hatalara açık bir yöntemdir. Kaliteli sorulardan oluşan, hedeflere uygun ve yapılandırılmış sınavların süratli bir şekilde hazırlanması ve değerlendirilmesi amacıyla bilişim teknolojilerinin ve alt yapısının kullanılması kaçınılmazdır.

Bu tez çalışmasında, bir sınav oluşturmak amacıyla, belirlenen özelliklere göre, bir soru bankasının oluşturulması ve bu soru bankasından otomatik olarak soru seçilmesi için genetik algoritmayı kullanan sınav hazırlama sistemi önerilmektedir. Soru bankasının oluşturulması sırasında sorunun; gövdesi, cevap şıkları, ilgili olduğu konu, geçmişteki sınavlarda sorulma sıklığı, öngörülen zorluk ve ayırt edicilik değerleri gibi özellikleri belirlenmektedir. Soruların sorulma sıklıkları, zorluk seviyeleri ve ayırt edicilik indeksleri istatistiksel olarak güncellenmektedir. Sistem, genetik algoritmayı kullanarak istenen sayıda, zorluk ve ayırt edicilik seviyelerinde sorular seçmekte ve sınav kitapçığı şeklinde önceden belirlenmiş düzende çıktı vermektedir. Böylece, eğitmenin soru bankasından sorular seçerek sınav oluşturması işlemini basitleştiren, pratik ve zeki bir yöntem geliştirilmiştir.

Anahtar Kelimeler: Genetik Algoritma, Otomatik Sınav, Sezgisel Yöntemler, Soru Bankası.

DESIGNING OF INTELLIGENT QUESTION BANK BY USING HEURISTIC METHODS

ABSTRACT

Thanks to developing information technologies the concept of time, the speed and accuracy of the calculations have importance. Today, people's need of training and documenting that they meet the goals of the training they receive is increasing. The appropriateness of the training to the targeted qualifications and the level of achievement can only be measured by examinations consisting of quality questions. When the level which technology reach is considered, preparation of quality questions is very slow and error is probable. It is inevitable to use information technologies and its infrastructure to prepare and evaluate exams consisting of quality questions.

In this thesis study, an exam preparation system which uses a Genetic Algorithm to form a question bank and automatically select questions from this question bank is suggested. During the preparation of the question bank the body of the question, answer choices, interest with the subject, the frequency of asked in the past examinations, the difficulty indexes and the discrimination indexes are determined. The frequency of questions, difficulty levels and distinguishing indices are updated statistically. The system selects the desired number of questions of difficulty and discrimination indexes, outputs a predetermined set of test booklets by using the Genetic Algorithm. A practical and intelligent method that enables the instructor to simplify the process of exam preparation by selecting questions from the question bank has been developed.

Keywords: Autotest, Genetic Algorithm, Heuristic Methods, Question Bank.

GİRİŞ

Gelişen teknoloji ile birlikte bilgisayar kullanımı günlük yaşamın her alanında yer almaktadır. Dünya’da ve Türkiye’de 2000’li yılların başından itibaren hızla yaygınlaşan kullanımı ile İnternet (URL-1), sosyal ve ekonomik alanda büyük bir değişim ve gelişim sağlamıştır. İnternet toplum için vazgeçilmez bir bilgi ve veri havuzu haline gelmiştir (Yılmaz, 2015). Böylece, bu gelişmelerin ölçme ve değerlendirme ve çeşitli alanlarda stratejiler üretilmesi amacıyla sıkça tercih edilen sınavların da internet kullanılarak gerçekleştirilmesi düşüncesine yansımaları kaçınılmaz olmuştur.

Sınav, ön hazırlıklardan değerlendirmeye kadar ardışık bir dizi işlem gerektiren bir süreçtir. Sınavların oluşturulmasını, değerlendirilmesini, sonuçlarının elektronik olarak saklanmasını ve işlenmesini kolaylaştırması nedeniyle çeşitli hedeflere uygun yerel ağ (İntranet) ve/veya İnternet destekli bilgisayar uygulamaları geliştirilmiştir. Tek bir bilgisayar kullanarak bu işlemlerin yapılması mümkün olmayacağından birden fazla kullanıcıya aynı anda hizmet sunulması için ise İnternetin kullanılması elzemdir (Karakaya, 2001).

İnternet üzerinden gerçekleştirilen elektronik sınavların; sağlıklı bir ölçme ve değerlendirme için gerekli işlemleri kolaylaştırması, hızlandırması, hataların kolay tespiti ve düzeltmelerin tekrar gerektirmemesi, ihtiyaç duyulan zamanı kısaltması, istatistik ve raporlama işlemlerinin hızla gerçekleştirilebilmesi gibi birçok avantajı bulunmaktadır. Fakat, sistemin geliştirilmesi, sağlıklı bir şekilde işletilmesi, sınavın ön hazırlıklarının uzun bir süreç gerektirmesi ve kullanımının zor olabilmesi gibi birçok titizlikle ele alınması gereken olguyu da barındırmaktadır.

Eğitimde bilişim teknolojilerinin kullanımı konusunda ortaya çıkan hızlı gelişmeler sonucunda e-öğrenme ve Öğrenme Yönetim Sistemleri (ÖYS) geliştirilmiş ve yaygın olarak kullanılmaya başlanmıştır. ÖYS kullanımı, mekan-bağımsız bir eğitim yöntemi olarak uzaktan eğitimde de tercih edilmektedir. Bununla birlikte ÖYS’nin;

öğrencilerin mekan-bağımsız olmasından kaynaklanan kopya çekilmesi, öğrenci adına başkasının soruları cevaplama, eş zamanlı sınav uygulaması için bilgisayar laboratuvarı oluşturmanın yüksek maliyetleri gibi sorunları bulunmaktadır. Ayrıca farklı coğrafik bölgeler ve farklı koşullarda sınava giren öğrencilere yönelik eş zamanlı sınav uygulamalarına özgü sorunlar ve sistemin işletilmesi sırasındaki teknik aksaklıklar gibi çeşitli sorunları da bulunmaktadır. Bu sorunları ortadan kaldırmak ya da en aza indirmek için geleneksel sınav uygulamaları halen devam etmektedir. Geleneksel sınav uygulamalarında yazılı yoklamalar, kısa cevaplı testler, doğru-yanlış testleri, eşleştirmeli testler, çoktan seçmeli testler ve birden fazla test türünün aynı anda uygulanması gibi yöntemler kullanılmaktadır.

Bu tez çalışmasında, bir sınavın elektronik ortamda oluşturulduğu ve geleneksel yöntemle sınav uygulamasının gerçekleştirildiği bir ölçme-değerlendirme sistemi ele alınmıştır. Bilgisayar ortamında sınav yapmanın zorlukları göz önüne alınarak geliştirilmiş olan örnek uygulamada, öğretim elemanlarına uygulama kolaylığı sağlayan bir ara yüz sunularak, sınav uygulaması ile ilgili zorlukları en aza indirmek hedeflenmiştir.

Akıllı soru bankası ve çevrimiçi sınavlarla ilgili olarak literatürde gerçekleştirilmiş olan yedi çalışma irdelenmiş ve aşağıda yer alan bulgulara ulaşılmıştır.

İçten (2006) yüksek lisans tezinde, yönetici rolüne sahip kullanıcıların ders ve öğrencilerle ilgili çeşitli işlemler yapabildiği; öğretim elemanı rolüne sahip kullanıcıların sınav oluşturma, değerlendirme, sınav istatistikleri ve rapor alma görevlerini yürütme, mesaj gönderme, duyuru ekleme gibi işlemleri yapabildiği bir sistem tasarlamıştır. Öğrenci rolüne sahip kullanıcılar sınava girebilme, sınav sonuçlarını ve duyuruları görebilme gibi işlemleri yapabilmektedir. Vize, final ve bütünleme türlerinde sınavlar oluşturulabilmektedir. Öğretim elemanı, çoktan seçmeli sorular hazırlayarak sisteme ekleyebilmektedir.

Çinici (2006) yüksek lisans tezinde, Madde Tepki Kuramı modeli ile öğrencilerin bilgi seviyelerine uygun çoktan seçmeli sorularının, ÖYS - Soru ve Test Birlikte Çalışabilirliği standardında oluşturulmuş olan soru bankasından otomatik olarak

seçilmesini sağlayan sınav ve değerlendirme sistemi tasarlamıştır. Öğrenciler, başarısının daha düşük olan konulardan daha fazla soru ile oluşturulan sınavlara girebilirler ve sınav sonucunda elde edilen geri bildirimlerle konulardaki eksikliklerini giderme şansı bulabilirler.

Gezgin (2006) yüksek lisans tezinde, yönetici rolünün kullanıcı listesi oluşturma, etkinleştirme işlemleri, şifre ve güncelleme işlemleri yapabildiği; öğretim elemanı rolünün not listesi, soru ekleme, not güncelleme, etkinleştirme ve soru bankası işlemleri yapabildiği; öğrenci rolünün sınavlar, alıştırma sınavları ve not bilgileri etkinliklerini yapabildiği bir sistem tasarlamıştır. Yapılan çalışmada, açık uçlu, çoktan seçmeli ve boşluk doldurmalı olmak üzere üç tipte soru oluşturulabilmektedir.

Akın (2007) yüksek lisans tezinde, sistem tarafından ya da kullanıcı tarafından soru seçimi yapılabilen, güvenli, açık uçlu veya çoktan seçmeli sorulardan oluşan sınav uygulamalarına olanak sağlayan bir sistem tasarlamıştır. Tasarlanan sistemde; çoktan seçmeli veya açık uçlu soru, sorulara resim, animasyon ve/veya dosya ekleme, soruların seviyelendirilmesi, açık uçlu veya çoktan seçmeli sınav ekleme, sınav seviyelendirme, sınav değerlendirme, istatistiki rapor oluşturma, sınava erişim kısıtları ekleme, silme, güncelleme ve listeleme işlemleri yapılabilmektedir. Ayrıca, yönetici modülü üzerinden bölüm, dönem, ders, öğretim elemanı ve öğrenci bilgileri sisteme eklenebilmekte, güncellenebilmekte ve silinebilmektedir.

Ata (2008) yüksek lisans tezinde, bant genişliğini verimli kullanan ve soru bankasının çoktan seçmeli sorulardan oluşmasını sağlayan bir sistem tasarlamıştır. Sorular sırayla; ders, modül ve konu seçilerek eklenebilmektedir. Sınav oluşturmak için sırayla; ders ve modül seçilip sınav oluşturulabilmektedir. Gerçekleştirilecek sınav için konu seçilip soru adedi belirtilerek sınava istenilen sayıda ve rastgele sorular eklenebilmektedir.

Günoğlu (2008) yüksek lisans tezinde, veri tabanındaki soru bankasından rastgele sorular seçerek çevrimiçi sınav yapılmasına olanak sağlayan devimsel bir sistem geliştirmiştir. Bu sistemde eğitici; haber, makale, örün adresi, ders ve konulara göre grup oluşturulabilmekte, çoktan seçmeli soru, kullanıcı, sınav ekleme, düzenleme ve

silme işlemlerinin yanı sıra sınav sonuçlarının istatistiğini görüntüleme işlemini de gerçekleştirebilmektedir. Öğrenciler; duyuru, haber ve makalelere ulaşabilme, e-posta gönderebilme, aldığı derslere, derslerin sınavlarına ve sınav sonuçlarına ulaşabilmenin yanında genel ortalamayı görme gibi işlemleri de yapabilmektedir.

Sarıkaya (2011) yüksek lisans tezinde, uygulamalı sınavların yapılabileceği bir sistem tasarlamıştır. Soru bankasındaki soruların zorluk derecesini ve konusunu öğretim elemanı belirlemektedir. Öğretim elemanı yeni bir soru oluşturup sınav tasarlayabilmekte, zorluk derecesine ve konuya göre soruları seçerek sınava ekleyebilmektedir. Öğrenci sınavı uygulayıp sınav sonuçlarıyla ilgili detayları ve bilgileri görebilmektedir. Değerlendirme bağlantısı sayesinde, sınava giren tüm öğrencilerin sınav bilgilerine ulaşıp grafiksel olarak gösterilebilmektedir.

Bu tez çalışması 4 bölümden oluşmaktadır.

Birinci bölümde; eğitim kavramı, ölçmenin önemi, ölçme yöntemleri, çoktan seçmeli sorular ve otomatik sınavlar hakkında genel bilgiler verilmektedir.

İkinci bölümde; sezgisel yöntemler hakkında genel bilgilerin yanı sıra genetik algoritmalar ile ilgili detaylı bilgiler, genetik algoritmaların biyolojik arka planı, genetik algoritmalarda kullanılan seçim, çaprazlama ve mutasyon operatörleri hakkında bilgiler sunulmaktadır.

Üçüncü bölümde; tez çalışması için geliştirilmiş olan akıllı soru bankası sistemi ele alınmaktadır. Geliştirilmiş olan akıllı soru bankası sisteminin geliştirme ortamı, uygulamanın alt yapısı, ara yüzü, konu ve soru ekleme, silme, güncelleme, genetik algoritma kullanarak sınav oluşturma süreçleri detaylı bir şekilde anlatılmaktadır.

Dördüncü bölümde; geliştirilen akıllı soru bankası sisteminin eğitime ve ölçme-değerlendirme alanına sağladığı katkılar, avantajlar ve gelecekte bu alanda yapılabilecekler ile ilgili önerilerden bahsedilmektedir.

1. EĞİTİM VE ÖLÇME

1.1. Eğitim

Ertürk'e göre eğitim; bireylerin davranışlarında kendi yaşantısı yoluyla kasıtlı olarak ve istendik değişme meydana getirme sürecidir (Ertürk, 1994). Yine başka bir görüşe göre eğitim; çevre ayarlaması yoluyla bireylerin davranışlarını istendik biçimde değiştirmesi ve değerlendirmesi sürecidir (Sönmez, 1993). Bireylerin davranış ve tutumlarında istendik bir şekilde değişiklik meydana getirilmesi için eğitim ve öğretime ihtiyaç vardır.

Öğrenme ve öğretmenin de uzun yıllar boyunca pek çok değişik tanımı yapılmıştır. Öğrenmenin en yaygın ve kabul görmüş ifadesi, "yaşantı ürünü ve nispeten kalıcı izli davranışlar" şeklinde yapılmıştır (Ertürk, 1994). Birbirini izleyen öğrenme süreçlerinin eğitim olarak kabul edilebilmesi için bu öğrenme süreçlerinin önceden belirlenmiş eğitsel hedef veya hedefler doğrultusunda birbirini destekleyen, tamamlayan ve geliştiren yapıda gerçekleştirilmelerine gereksinim duyulmaktadır. Bu bakımdan gerçekleşen öğrenmenin eğitim olarak sayılabilmesi, eğitim sürecinin başlıca basamakları olan amaç, öğretme-öğrenme etkinlikleri ve değerlendirme öğelerini sağlamasına bağlıdır (Akıncı, 1998).

Amaçlar, kişilere kazandırılması istenen bilgi, beceri ve tutumlar ile bunların hangi konular vasıtasıyla edindirileceğini gösterir. Öğretme ve öğrenme faaliyetleri esnasında ise eğitilenin söz konusu bilgi, beceri ve tutumları öğrenmeleri sağlanır. Öğretim süreci sonunda elde edilen çıktıların amaçlara uygunluğu ve yeterliliği gerçekleştirilecek olan değerlendirme etkinliği ile anlaşılır (Fidan ve Erden, 1987).

Özel hedefler ise öğrencinin sahip olması uygun görülen özelliklerdir. Özel hedefler, genellikle bir bilim dalı veya çalışma alanı için hazırlanmaktadır. Belli bir disiplinde öğrencinin yetiştirilebilmesi için gerekli eğitim durumlarının tespit edilmesinde ve değerlendirilmesinde özel hedeflerin kullanılması amacın gerçekleştirilmesinde faydalı olacaktır (Ertürk; 1994).

1.2. Ölçme ve Önemi

Ölçme, genel anlamda herhangi bir niteliğin gözlemlenerek sonuçlarının sayılar ya da başka sembollerle ifade edilmesidir. Değerlendirme ise ölçme işleminin sonuçlarının ölçülen alana ait bir ölçütler ile kıyaslanarak bir yargıya veya bir karara ulaşmasıdır (Baştürk ve diğ., 2014).

Eğitim sisteminde ölçme çok önemli bir yer tutmaktadır. Ölçülen bireyin özellikleri ile ilgili bilgi sahibi olunmasını sağlamaktadır. Bu nedenle, ölçmenin hatasız olarak, yani güvenilir bir şekilde yapılabilmesi gerekmektedir. Çünkü, hatasız ya da hatası en aza indirgenmiş ölçmeler doğru sonuca yaklaşılmasını sağlar. Bu sebeple, eğitimde yapılacak ölçme işlemlerinin güvenilir olması gerekmektedir. Hatalar yalnızca bireyden değil ölçme aracından veya çevresel koşullarından da kaynaklanabilmektedir. Güvenirliğin yüksek olması ölçme sonuçlarının tesadüfi hatalardan arındırılmış olduğu manasına gelmektedir (Turgut, 1984).

1.3. Ölçme Yöntemleri

Ölçme ve değerlendirme süreçlerinde kullanılan; öğrenci ürün dosyası, performans ödevi, proje, öz değerlendirme, akran değerlendirme, grup değerlendirme, gözlem, görüşme, poster, sunum, dereceli puanlama anahtarları, kavram haritası, kontrol listeleri, açık uçlu sorular, doğru-yanlış soruları, eşleştirmeli sorular ve çoktan seçmeli sorular gibi birçok ölçme ve değerlendirme yöntemi ve aracı bulunmaktadır. Bu ölçme ve değerlendirme yöntemlerinin özünde, öğrencilerin özelliklerine uygun olarak hem öğrenme sürecinin hem de ürünün ölçülüp değerlendirildiği çoklu değerlendirme yöntemi bulunmaktadır (Özdemir, 2010).

1.3.1. Açık Uçlu sorular

Açık uçlu sorular, cevabın öğrenci tarafından düşünülüp planlanarak, dilediği gibi dilsel becerilerini kullanarak, yazılı olarak ifade edildiği sorulardır. Düşünme ve planlama işleminin yanında yazılı olarak ifade gerektirdiğinden sınav hızı düşüktür. Cevaplama işlemi öğrencinin çok zamanını alır. Açık uçlu sorulardan oluşan sınav süresinin çok büyük bir kısmı cevapların düzenlenip yazılmasına harcanmaktadır. Cevapların herhangi bir sınırı yoktur ve bu sebeple verilen cevaplar için tamamen

dođru veya tamamen yanlış çıkarımı yapmak genellikle mümkün değildir. Bu özelliklerinden ötürü, açık uçlu herhangi bir soruya verilen cevabın dođruluđu ve deđerlendirici tarafından uygun görülen puanın dođruluđu tartışmalara yol açmaktadır. Açık uçlu soruların güçlük derecesinin istatistiksel yöntemlerle tayin edilmesi güçtür. Genellikle, kısa cevaplı veya çoktan seçmeli sorular ile ölçülemeyecek veya ölçülmesi güç olan özelliklerin yoklanması gerektiğinde kullanılması beklenir. Açık uçlu soruların ve açık uçlu sorulardan oluşan sınavların hazırlanması kısa sürmesine rağmen, deđerlendirmesi uzun zaman alabilir ve deđerlendirme işlemi öznelidir (Baştürk ve diđ., 2014; Baş ve Beyhan, 2016; Bektaş ve Kudubeş, 2014).

1.3.2. Kısa Cevaplı Sorular

Kısa cevaplı sorular, aynı açık uçlu sorularda olduđu gibi cevabın öğrenci tarafından düşünülüp yazılı olarak ifade edildiđi sorulardır. Öğrencinin sınanan bilgiyi hatırlamasını veya bulmasını yoklar. Bilgiyi hatırlamak, verilen bilgiler arasında o bilgiyi tanımak veya yanlış olan bilgilerden ayırt etmekten farklıdır. Kısa cevaplı sorularda öğrenci, dilediđi cevabı verme özgürlüğüne sahiptir. Bu nedenle beklenenden çok farklı cevaplar alınabilir. Özellikle soruların ifadesi net bir biçimde yapılamamışsa ve öğrencinin bilgi düzeyi de düşükse istenen cevap ile alakasız cevaplarla karşılaşılır. Kısa cevaplı sorularda cevaplar kısa olduğundan, bir sınav süresinde çok sayıda soru sorulabilir ve daha geniş bilgi seviyesi sınanabilir. Böylece, müfredatı daha kapsayıcı bir sınav yapılabileceğinden, açık uçlu sorulardan oluşan bir sınava kıyasla, daha güvenilir ve geçerliđi daha yüksek bir sınav yapılabilir. Kısa cevaplı sorulardan oluşan sınavlar eğitimin her düzeyi için uygundur. Deđerlendirmenin açık uçlu sorulara göre daha kolay olması, cevapların açık uçlu sorulara nazaran sınırlı ve kesin olmasından kaynaklanmaktadır. Öğrencinin dilediđi cevabı verme özgürlüğü deđerlendirmenin öznel olmasına neden olabilmektedir (Baştürk ve diđ., 2014).

1.3.3. Çoktan Seçmeli Sorular

Çoktan seçmeli sorular, sorunun dođru cevabı kendi içerisinde verilmiş olan sorulardır. Öğrenciler, açık uçlu ve kısa cevaplı sorularda olduđu gibi, dilsel becerilerini kullanarak dilediđi şekilde cevap veremezler; bunun yerine cevabı, verilen

seçenekler arasından seçmeleri gerekmektedir. Tüm cevaplı sorularda olduğu gibi, çoktan seçmeli sorularda da sorulan sorunun cevabı seçenekler arasında verilmiş olduğundan, öğrencinin doğru cevabı tanınması ve çeldiriciler arasından seçmesi gerekmektedir. Daha ileri bilgi düzeyleri ölçülmek isteniyorsa, çağrışım ve hatırlatmalar madde yapısı güçlendirilerek önlenbilirse, çoktan seçmeli sorulardan oluşan sınavlar yalnızca basit ve önemli bilgilerin sınanabileceği bir sınav türü olmaktan çıkarlar. Sınav için harcanan sürenin büyük bir kısmı soru ve cevapları okumak ve doğru cevabı bulmak için harcanmaktadır. Tercih edilen cevabın işaretlenmesi az zaman alır. Bu nedenle, sınırlı bir sınav süresinde açık uçlu sorularla veya kısa cevaplı sorularla oluşturulan sınavlara kıyasla daha çok sayıda çoktan seçmeli soru ile sınav hazırlanabilir. Çoktan seçmeli sorular ile hazırlanan bir sınavda, öğrencinin sınavla sınanmak istenen bilgi ve tutumunun yanında, okuma ve anlama yeteneği de sınanmaktadır.

Çoktan seçmeli sorulara verilen cevaplar kesinlikle doğru, kesinlikle yanlış ya da en doğru şeklinde sınıflandırılabilir. Bu sebeple, çoktan seçmeli sorular nesnel olarak değerlendirilebilir. Çoktan seçmeli sorulardan oluşan sınavlarda kullanılacak çeşitli madde formları geliştirilebilir ve bu sayede çeşitli beceri ve tutumların ölçülmesi de sağlanabilir. Çoktan seçmeli sorular, farklı madde formları kullanılmak koşuluyla, tüm eğitim basamaklarına uygulanabilir. Çoktan seçmeli sorularda doğru cevabın tesadüfen bulunabilmesi ihtimali bulunmaktadır. Sorulardaki seçenek sayısının artırılması şans başarısının azalmasını sağlamaktadır. Sınavdaki soru sayısı arttıkça, sınavın tümünden yalnızca tesadüfen önemli bir puan kazanma olasılığı da azalmaktadır. Çoktan seçmeli sorulardan oluşan sınavlarda maddeler yazılırken, sınav veya soru gücünü belirlemek ve sınav uygulandıktan sonra istenilen güçlükte soru seçmek mümkündür. Her bir sorunun nesnel olarak değerlendirilebilmesi için sorunun güçlük indeksinin istatistiksel olarak hesaplanmasına olanak vermektedir. Ayrıca soru zorluğunu çeldiricileri doğru cevaba yakınlaştırarak yapmak da mümkündür. Çoktan seçmeli soruların oluşturulması açık uçlu sorulara göre daha uzun zaman alabilir fakat sınav değerlendirme işlemi daha kısa sürede gerçekleştirilebilir (Baştürk ve diğ., 2014).

1.4. Otomatik Sınav

Ölçme ve değerlendirme süreçleri de bilgisayar destekli veya otomatik olarak gerçekleştirilebilmektedir. Ölçme ve değerlendirme faaliyetinin bilgisayar destekli veya otomatik olarak gerçekleştirilmesinde; soru bankası oluşturma, sınav oluşturma ve öğrenciyi değerlendirme başlıkları altında ele alınabilecek temel süreçler bulunmaktadır.

Soru bankası oluşturma sürecinde, çok çeşitli yeterliliklerin ölçülmesine olanak sağlayacak nitelik ve çeşitlilikte soruların tanımlanması gerekmektedir. Soruların; ölçülmek istenen davranışın konusu, soru türü, zorluk ve ayırt edicilik değerleri gibi özelliklerinin belirlenmesi gerekir. Soru bankası soruların özelliklerini saklamaktadır ve otomatik sınav sisteminin en önemli bileşeni konumundadır. Soru bankası mümkün olabildiğince fazla sayıda soru barındırmalıdır. Böylece öğretmen, hedef davranışı ölçmeye yönelik yeterli sayıda ve seviyede sorudan rastgele veya seçerek ölçme aracını oluşturabilir.

Sınav oluşturma sürecinde, belirlenen özel ölçütlere dayalı olarak hedeflenen yeterliliklerin ölçülmesine olanak sağlayacak nitelikte sorulardan oluşan bir sınavın tanımlanması gerekir. Ölçülecek konunun hedeflenen öğrenme düzeyine ulaşmada keskin bir biçimde ölçülmesi önemlidir. Oluşturulacak olan sınavın hedeflenen düzeyde zorluk ve ayırt edicilik değerlerine sahip olması ölçme ve değerlendirmenin etkinliğini arttıracaktır.

Öğrenciyi değerlendirme sürecinde, ölçüm sonuçlarının belirli bir ölçüt ile karşılaştırılması ve hedefe ulaşma derecesinin yorumlanması gerekmektedir. Mutlak ve bağıl değerlendirme olmak üzere iki çeşit değerlendirme yöntemi bulunmaktadır. Mutlak değerlendirmede ölçüt sabittir. Örneğin bir sınavda 50 puan ölçüt olarak belirlenmişse, öğrencinin puanı 49 ise başarısız, 51 ise başarılı olarak nitelendirilmektedir. Bağıl değerlendirmede ise, ölçüm sonuçlarının birbirine göre değerlendirilmesi söz konusudur. Bağıl ve mutlak değerlendirmede öğretmen ölçüm sonuçlarını grup değerlendirmesi ile elde eder. Geleneksel yöntemler ile ölçüm sonuçlarının değerlendirilmesi zor ve zaman almaktadır. Bilgisayar desteği olmadan tüm öğrencilerin ölçüm sonuçlarının aritmetik ortalaması, standart sapması, ranjı,

maksimum ve minimum deęerlerinin bulunması eęitmen aısından uzun zaman ve aba gerektiren bir iřtir. Öğrencinin ölçüm sonuçları bilgisayar yardımıyla işlendięi takdirde, deęerlendirme sonuçlarına ulaşmak çok hızlı ve kolay bir şekilde gerçekleşir. Bilgisayar desteęiyle elde edilen deęerlendirme raporu, öğrencinin öğrenme güçlüğü ektięi noktaları belirlemede, öğretim programının eksiklerini tespit etmekte, öğrencinin öğrenme düzeyini tespit etmede ve verilen eęitimin kalitesini artırmada eęitmene yardımcı olur (elik, 2006).



2. SEZGİSEL YÖNTEMLER

Var olan bir problemi çözmek amacıyla, farklı çözümlerden en iyi olanını tercih etmek için kullanılan yöntemler sezgisel yöntemler olarak isimlendirilmektedir. Çözüm süresi bir ömür kadar uzun sürebilecek problemlerin çok kısa sürede çözülebilmesi ve farklı türden problemlere uygulanabilmesi sezgisel yöntemlerin tercih edilme sebeplerindedir. Çözümün garanti edilememesi, parametre sayısının çok olması ve uygun şekilde tasarlanmasının gerekliliği sezgisel yöntemlerin dezavantajı olarak gösterilebilir. Sezgisel yöntemlerin uygulanan metodun doğruluğunun ispat etmesi gerekmez. Yöntemden asıl istenen karmaşık bir problemi daha basit hale getirmesi veya algoritmanın tatmin edici bir sonuç bulabilmesidir.

2.1. Genetik Algoritmalar

2.1.1. Biyolojik Arka Planı

Evrende var olan tüm canlı organizmalar hücrelerden meydana gelmektedir. Her bir hücrede aynı kromozom kümesi bulunur. DNA, organizmaların canlılık faaliyetleri ve biyolojik gelişmeleri için genetik talimatları taşıyan, adenin, timin, guanin ve sitozin aminoasitlerinin çeşitli bileşimlerinden oluşan bir nükleik asittir. Kromozomlar ise birer DNA dizeleridir ve tüm organizma için bir model olarak hizmet ederler. Bir kromozom genlerden, DNA bloklarından oluşur; her gen belirli bir proteini kodlar. Temel olarak her bir genin göz rengi, saç rengi gibi özellikleri kodladığı söylenebilir. Her genin kromozomda mevki (locus) olarak isimlendirilen kendine ait bir konumu vardır.

Genetik materyaldeki tüm kromozomlar genom olarak adlandırılmaktadır. Genomdaki belirli gen kümelerine ise genotip adı verilmektedir.

Üreme sırasında, ilk olarak çaprazlama işlemi gerçekleşir. Ebeveynlerden gelen genler, tamamen yeni bir kromozom oluşturur. Yeni oluşan yavru daha sonra mutasyona uğrayabilir. Mutasyon, DNA'daki elementlerin bazılarının değişmesi

anlamına gelir. Bu deęişiklikler temel olarak ebeveynlerden gelen genlerin kopyalanmasındaki hatalardan kaynaklanmaktadır. Bir canlının güçlülüęü veya uygunluęu, yaşamındaki başarısı ile ölçülür.

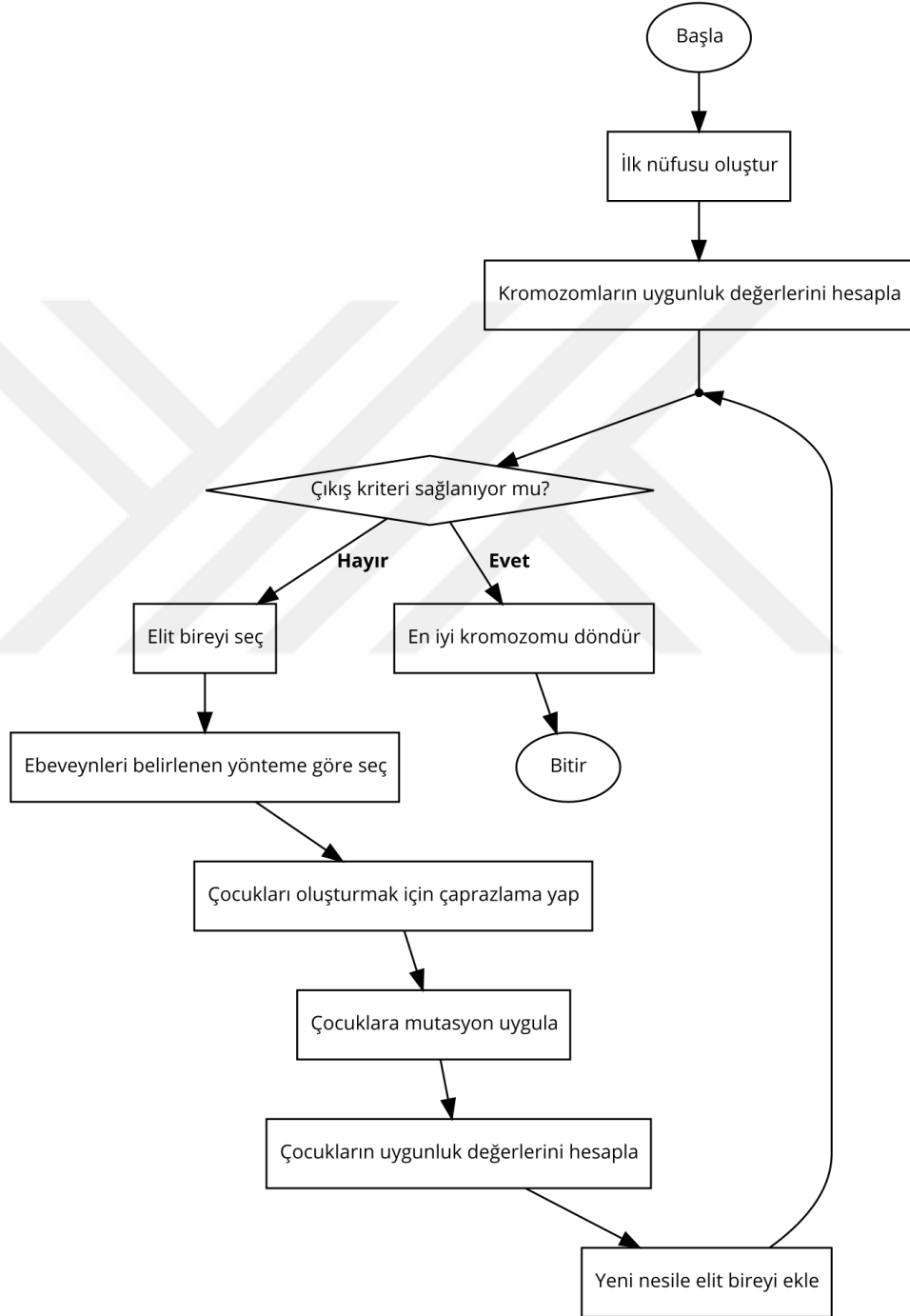
2.1.2. Genetik Algoritma ve Özellikleri

Genetik Algoritma (GA), bilgisayar bilimcilerin doğadan öğrendięi ve kendi problemlerini çözmek için kullandığı, genetik ve doğal seçim prensibine dayanan, arama temelli bir eniyileme (optimizasyon) tekniğidir. GA, Darwin'in Evrim Teorisi'nden esinlenmiştir. Genellikle, çözümleri bilgisayar hızıyla dahi olsa bir ömür veya dünyanın yaşından daha fazla sürebilecek derecede zor olan problemlerin en uygun ya da en uyguna yakın sonuçlarını çok kısa bir sürede bulma amacıyla kullanılır. GA'lar eğitimden uzay araştırmalarına kadar çözümlerinin normal yöntemler ile bulunmasının güç olduęu çeşitli problemlere çok farklı şekillerde uygulanabilmektedir. Temel ilkeleri John Holland tarafından 1970'li yıllarda ortaya atılmıştır. 1989 yılında David E. Goldberg, "Genetik Algoritma Uygulamaları" üzerine klasik eser olarak kabul edilen kitabını yayınlamıştır.

Nüfus tabanlı bir algoritma olan GA'nın çözüm metodu temelde çok geneldir. GA, iyi kromozomlar seçerek yeni nesillerin ebeveynlerinden daha iyi olacağı fikrine dayanmaktadır. Her kromozom problemin çözümü olmaya adaydır. Olasılık kurallarına göre çalışan genetik algoritmalar, yalnızca iyi tanımlanmış bir amaç fonksiyonuna gereksinim duymaktadır. Çözüm uzayının tümünü deęil, çözüme yakın olabilecek olan belirli bir kısmını taramaktadır. Böylece, etkin arama yaparak eđer çözümü varsa çok uzun zaman alabilecek problemlerin çözümlerine çok daha kısa bir sürede ulaşmaktadır (Goldberg, 1989). GA'nın önemli bir avantajı ise, çözümlerden oluşan nüfusları eş zamanlı incelemesi ve bu sayede yerel en iyi çözümlere baęımlı olmamasıdır.

Algoritmanın çalışması Şekil 2.1'de gösterilmektedir. Algoritma, nüfus denilen ve kromozomlar ile temsil edilen bir dizi çözümle başlatılır. Burada nüfusu oluşturan kromozomlar ve kromozomları oluşturan genler rastgele üretilmektedirler. Her bir kromozom bir aday çözümü ve her bir gen çözümün parametrelerini temsil etmektedir. Nüfus içerisindeki çözümler genetik işlemlere tabi tutulmakta ve bir sonraki

jenerasyondaki nüfusu oluşturmak için kullanılmaktadır. Oluşan yeni nüfusun eskisinden daha iyi olacağı umulmaktadır. Yeni jenerasyon oluşturma süreci belirli bir jenerasyon sayısı veya önceden tanımlı bir iyileştirme eşik seviyesi sağlanana kadar tekrarlanmaktadır.



Şekil 2.1. Genetik algoritma genel akış şeması

Genetik biliminde kullanılan üç temel işlem olan çaprazlama (crossover), mutasyon (mutation) ve seçilim (selection) GA'da da kullanılmaktadır. Çaprazlama ve mutasyon, bir genin değişiminde rol oynayan iki temel işlemdir. Bu işlemlere tabi tutulacak genlerin (ebeveyn genler) belirlenmesinde seçilim kullanılmaktadır. Seçilim, Darwin'in güçlü olan yaşar prensibini gerçekleştirmektedir ve genlerini sonraki kuşaklara aktaracak kadar güçlü olan ebeveynlerin seçilmesini sağlar.

2.2. GA'yı Oluşturan Operatörler

2.2.1. Seçilim

Seçilim, bir nüfustaki uygunluk değerleri yüksek kromozomların çaprazlanarak gelecekteki nesillere genomlarını aktarması amacıyla seçildiği aşamadır.

Burada en önemli gösterge uygunluk değeridir. Uygunluk fonksiyonu, her kromozom için tek tek işletilir ve tüm kromozomlara ait uygunluk değerleri belirlenir. Bazı seçilim yöntemleri normalleştirme işlemine ihtiyaç duymaktadır. Normalleştirme işlemi ise her bir kromozomun uygunluk değerinin tüm kromozomların uygunluk değerlerinin toplamına bölünmesiyle gerçekleştirilir. Sonuç olarak ortaya çıkan tüm uygunluk değerlerinin toplamı 1'e eşittir.

Büyük problemler için seçilim işlemleri hesaplanma açısından oldukça zorlayıcı olabilir.

Seçilim işlemi için tüm kromozomları dikkate almayan bazı seçilim algoritmaları da vardır. Bu algoritmalar, sadece keyfi olarak belirlenmiş bir sabitten daha yüksek uygunluk değerine sahip olan kromozomlarla seçilim işlemi gerçekleştirir. Diğer algoritmalar ise, uygunluk değerine bağlı olarak, kromozomların yalnızca belirli bir yüzdesinin seçilimine izin verilen kısıtlı bir havuzdan seçim yapar.

Seçilim ile ilgili çözümü kolaylaştıran diğer bir yöntem ise elitizmdir. GA'daki amaç çözüme hızlı bir şekilde ulaşmayı sağlar. Bu da en uygun birkaç kromozomun sonraki nesillere aktarılması ile gerçekleşebilir. Seçilim aşamasında seçilmesi gerçekleşmeyen ancak en iyi uygunluk değerine sahip olan belirli sayıdaki kromozom, çaprazlama ve mutasyon sonucunda üretilen kromozomların arasında en kötü uygunluk değerlerine

sahip olan kromozomlar ile yer deđiştirilir. Bu sayede, en uygun kromozomlar seçkin kromozomlar olarak gelecek nesillerde eşleşme işlemlerine girebilirler.

Genetik algoritmalarda kullanılan bazı seçim yöntemleri aşağıdaki bölümlerde verilmektedir.

2.2.1.1. Rulet tekerleđi Seçilimi

Rulet tekerleđi seçim yönteminde, ilk adım, tüm kromozomların uygunluk değerlerinin toplanması yoluyla tüm nüfusun toplam uygunluk değerini hesaplamaktır. İkinci adım, her bir kromozom için $P_{seçilimi} = \frac{Uygunluk_i}{Toplam\ Uygunluk}$ denklemi ile seçim olasılığı hesaplanır. Üçüncü adımda, kromozomların seçim olasılıklarına göre büyükten küçüğe bir dizi üzerinde kümülatif olarak sıraya konulur. 0 ile 1 aralığında, çaprazlamaya girecek kromozom sayısı kadar rastgele sayı üretilir ve her bir rastgele sayı seçim olasılığı dizisi üzerinde hangi kromozom üzerine düşüyor ise o kromozom seçilmiş olur. Bu nedenle, kromozomlar seçim olasılıklarına göre belirlenir ve seçim olasılığı yüksek yani uygunluk değeri yüksek olan kromozom daha büyük olasılıkla seçilmiş olur.

2.2.1.2. Rastgele Olmayan Örnekleme Seçilimi

Rastgele olmayan örneklemede, her bir kromozomun seçim olasılığı ($P_{seçilimi}$), her bir kromozoma ait uygunluk, ortalama uygunluđa bölünerek hesaplanır. Bu işlemin tamsayı kısmı, çaprazlamaya girecek kromozomları seçme, tüm kesir ise kromozomları sıralama amacıyla kullanılır. Yeni nüfusta kalan serbest yerler var ise, sıralanan listede en yüksek kesir değeri sahip olan kromozomlarla doldurulur.

2.2.1.3. Doğrusal Sıralama Seçilimi

Doğrusal sıralama seçiminde kromozomlar önce uygunluk değerlerine göre sıralanır. Yüksek uygunluk değerlerine sahip olanlar yüksek sıralarda, düşük uygunluk değerlerine sahip olanlar ise düşük sıralarda olacaktır. Daha sonra, kromozomlar sıralamadaki sırası ile doğru orantılı bir olasılıkla seçileceklerdir (Baker, 1985).

2.2.1.4. Seçkincilik Yöntemi

Elitizm işlemi, kromozomların iyi genlerini çaprazlama ve mutasyon sürecinde bir çocuğa aktaramaması olasılığına karşın, bir veya birkaç kromozomun tüm genlerini olduğu gibi yeni nüfusa kopyalanmasıdır. Elitizm, olası en iyi çözümü kaybetmeyi önlediği için GA'nın performansını ciddi bir şekilde arttırmaktadır.

2.2.2. Çaprazlama

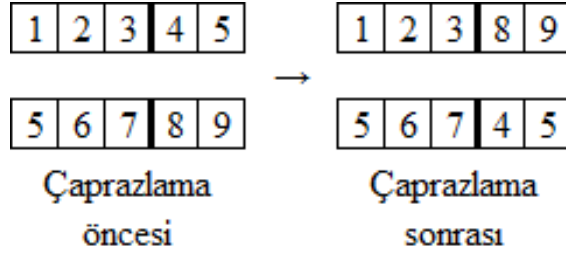
Çaprazlama işlemi ebeveynlerin özelliklerini kullanarak sonraki nesiller için yeni bireyler üretmek amacıyla gerçekleştirilir. Çaprazlama işlemi ile her iki ebeveynin belirli genleri çocuklara geçer. Bu sayede sonraki nesillerde bazı çocukların iyileşmesi veya bazı çocukların kötüleşmesi mümkündür.

Çaprazlama işlemi için ise;

1. Parçalı Eşleme Çaprazlaması (Partially Match Crossover),
2. Sıralı Çaprazlama (Order Crossover),
3. Döngüsel Çaprazlama (Cycle Crossover),
4. Kenar Birleştirmeli Çaprazlama (Edge Recombination Crossover), yöntemleri kullanılabilir.

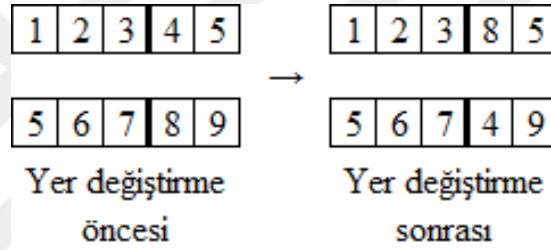
2.2.2.1. Parçalı Eşleme Çaprazlaması

Parçalı eşleme çaprazlaması, iki kromozomun genlerinin bir veya daha çok noktadan parçalanması işlemidir. Parçalanma işlemi sonucunda, çaprazlanacak olan kromozomların genleri yer değiştirir. Örneğin: [1 2 3 4 5] ve [5 6 7 8 9] kromozomları tek noktadan çaprazlanmak istenirse Şekil 2.2.'deki gibi olacaktır.



Şekil 2.2. Parçalı eşleme çaprazlaması örneği

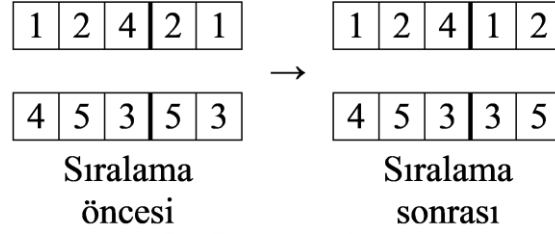
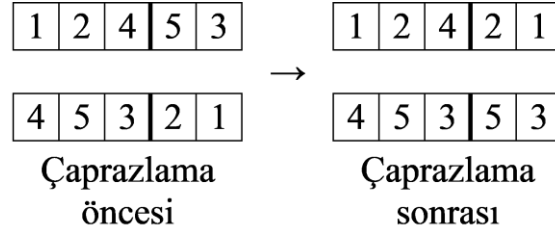
Kromozomlarda tekrarlı genler bulunması, bazı problem türleri için (kombinasyonel ve permutasyonel problemler) yukarıdaki çaprazlama işleminin bitmemesine neden olmaktadır. Normal şartlarda sorun yaratmayan bu durum, parçalı eşleme çaprazlaması için bir problemdir; Şekil 2.3'te gösterildiği gibi tekrarlı olan 5 geni, yeni yerinde daha önceden var olan 9 geni ile yer değiştirilerek mevcut sorun çözülür.



Şekil 2.3. Parçalı eşleme çaprazlaması sonrası yer değiştirme örneği

2.2.2.2. Sıralı Çaprazlama

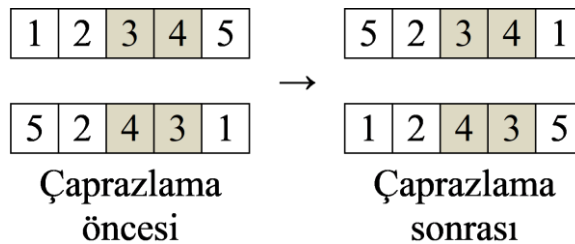
Sıralı çaprazlama işleminde, parça eşleştirmeli çaprazlamada olduğu gibi belirli bir noktadan çaprazlama işlemi gerçekleştirilir. Parça eşleştirmeli çaprazlama yönteminden farklı olarak, çaprazlama işleminin ardından kırılım noktasının sonrasındaki kısım kendi içerisinde sıralanır. Örneğin: [1 2 4 5 3] ve [4 5 3 2 1] ebeveynleri 3 numaralı genden itibaren çaprazlanmak istenirse işlem Şekil 2.4'teki gibi gerçekleşecektir.



Şekil 2.4. Sıralı çaprazlaması için çaprazlama ve sıralama örneği

2.2.2.3. Döngüsel Çaprazlama

Döngüsel çaprazlama işleminde, çaprazlama işlemi için seçilmiş olan kromozomlar arasında çevrimi sağlayacak döngüler bulunur. Çaprazlama döngülerinden bir kısmı sabit kalırken bir kısmı çaprazlanır. Örneğin: [1 2 3 4 5] ve [5 2 3 4 1] kromozomları çaprazlamak istenirse; [1 2 3 4 5] ve [5 2 3 4 1] genleri arasında var olan döngüler bulunmaya çalışılır. Döngüler, ardışık elemanların her iki kromozomda ortak olan genlerin bilgisini tutan ve başlangıç noktası yönüne doğru dönen yollardır. Belirtilen koşullara uyan çaprazlama işlemi Şekil 2.5'te gösterildiği gibi yapılmaktadır.



Şekil 2.5. Döngüsel çaprazlama örneği

Söz konusu çaprazlamada birden çok döngü olduğundan çaprazlama şekildeki gibi yapılabilir. Çaprazlamada farklı döngüler de seçilebilirdi; fakat şartlara uygun döngüler arasından rastgele seçilmektedir. Döngülerin tümünün tespit edilmesinin ardından, bir döngü çaprazlama için kullanılır.

2.2.2.4. Kenar Birleştirmeli Çaprazlama

Kenar Birleştirmeli Çaprazlama, sıralı kromozomlar için kullanılan çaprazlama tekniğidir. Amacı mümkün olan en az yolu taramaya çalışmaktır. Gezgin satıcı problemi gibi problemlerde, iki düğüm arasına bir uçurumun girmesi uygunluk değerinin hesaplanması açısından olumsuz bir durumdur. Ana fikir ise, çocuk üretmek için ebeveynlerin kromozomları kullanarak kenar veya düğüm arasında var olan olabildiğince çok sayıda bağlantı kurmaktır. En olumsuz yönlerinden birisi genellikle hesaplanmasının diğer yöntemlere göre daha uzun sürmesidir. Bu yöntemde ilk düğüm rastgele seçilir. Komşuluk listelerinde daima en az elemanı olan listeyi bularak tüm elemanları tüketme yöntemi izlenir.

Örneğin: [1 2 6 5 4 7 3] ve [7 6 1 2 3 4 5] ebeveynleri için komşuluk listesini tablo 2.1'deki gibi varsayalım.

Tablo 2.1. Kenar birleştirmeli çaprazlama komşuluk listesi

Düğüm	Komşuluk Listesi
1	2, 3, 6
2	1, 6, 3
3	1, 7, 2, 4
4	5, 7, 3, 5
5	6, 4, 7
6	2, 5, 7, 1
7	4, 3, 5, 7

Rastgele seçim sonucunda elde ettiğimiz değer olan 1 düğümü ile başlanır ise çocuk = [1] şeklinde olacaktır. Ardından, tüm komşuluk listelerinden 1 çıkartılır. 1'in komşulukları arasında en az komşuluğu bulunanın 2 olduğu görülmektedir. Bu durumda çocuk = [1, 2] şeklinde olacaktır. Aynı 1 düğümünde yapıldığı gibi 2 düğümü de tüm komşuluk listelerinden çıkartılır. 2 düğümünün komşularından 6 ve 3'ün her ikisi de sadece iki komşuya sahip olduğundan sadece ikisi arasındaki rastgele seçim ile 6 düğümü seçilirse; çocuk = [1, 2, 6] şeklinde olacaktır. 6 düğümü için de 1 ve 2 düğümlerine yapıldığı gibi komşuluk listelerinden çıkartma işlemi uygulanır. 6

düğümünün komşuluklarından en az elemanı olan 5 düğümü seçilir. Bu durumda çocuk = [1, 2, 6, 5] şeklinde olacaktır. Aynı işlemler sırasıyla 7, 3 ve 4 düğümlerinde uygulanmaktadır. En son aşamada çocuk = [1, 2, 6, 5, 7, 3, 4] şeklinde olacaktır (URL-2).

2.2.3. Mutasyon

Mutasyon, bir kromozomu oluşturan genlerin çok az bir miktarının değişmesidir. Genetik algoritmalarda mutasyon, bir nüfusun genetik çeşitliliğini korumak için kullanılır. Mutasyonun rolü, arama algoritmasının bir yerel en iyiye takılmamasını garanti etmektir. Seçilim ve çaprazlama işlemleri homojen bir çözüm kümesinde durgunlaşabilir. Bu durumda nüfusun ortalama uygunluğu geliştirilemeyebilir. Bununla birlikte, çözüm sadece yerel en iyi olarak görünebilir ve algoritma daha fazla ilerleyemeyebilir. Mutasyon, rastgele bir aramaya eşdeğerdir ve genetik benzerlik sorununu çözmektedir.

Mutasyon işlemi için;

1. Tersleme (Inversion),
2. Ekleme (Insertion),
3. Çıkartma (Displacement),
4. Yer değiştirme (Swap), yöntemleri kullanılabilir.

Tersleme, seçilen bir kromozomun rastgele bir değerinin tersine çevrilmesidir. Örneğin: [11010101] → [10010101] örneğindeki 2. kromozomun 1 olan değerinin tersi alınarak 0 yapılmıştır.

Ekleme, mevcut gen dizilimine yeni bir kromozom eklenmesi işlemidir. Örneğin: [11010101] → [110010101] dizisine 3. kromozom olarak 0 eklenmiştir.

Çıkartma, mevcut gen diziliminden bir kromozomun azaltılması işlemidir. Örneğin: [11010101] → [1010101] dizisindeki 2. kromozom olan 1 çıkarılmıştır.

Yer deęiřtirme, mevcut gen diziliminden rastgele seilen iki genin birbiriyle yerinin deęiřmesidir. rneęin: [11010101] → [10011101] dizisindeki 2. kromozom olan 1 ile 5. kromozom olan 0'ın yerleri deęiřtirilmiřtir.



3. AKILLI SORU BANKASI SİSTEMİ

Kullanıcıların sınav hazırlama sürecini yönetmesi ve harcadığı zamanı daha verimli kullanabilmesi amacıyla Akıllı Soru Bankası Sistemi geliştirilmiştir. Akıllı Soru Bankası Sistemi üzerinden kullanıcılar konu tanımlayabilir, tanımladıkları konulara soru ekleyebilir, konuları ve parametreleri belirleyerek GA ile sınav hazırlayabilir, hazırladıkları sınavı değerlendirebilir ve sınava ait raporlama yapabilir.

3.1. Uygulama Geliştirme Ortamı

Geliştirilen Akıllı Soru Bankası Sisteminde birden fazla ve farklı teknolojiler kullanılmıştır. Kullanıcı tarafından görüntülenecek olan ara yüz HTML, ColdFusion, JavaScript ve CSS ile geliştirilmiştir. Veri tabanı işlemleri için MSSQL veri tabanı yönetim sistemi kullanılmıştır. GA'ya soruların bilgilerinin gönderilmesi için JSON biçimi kullanılmıştır

3.1.1. Adobe ColdFusion

Adobe ColdFusion (CF), gelişmiş web uygulamaları geliştirmek amacıyla ColdFusion Markup Language (CFML) yazılım dilini kullanan ve Java Sanal Makinesi (JVM) üzerinde çalışan web sunucusudur. İlk olarak Allaire firması tarafından geliştirilmeye başlanmış olan CF, Macromedia ile iki şirketin birleşmesi sonucunda Macromedia'nın ürünü haline gelmiştir. Daha sonra Adobe firması ile Macromedia'nın da birleşmesi sonucunda "Adobe ColdFusion" olarak ismi güncellenmiştir. CF'da uygulama geliştirmek için kullanılan dile CFML ColdFusion Markup Language ismi verilmektedir. CFML kod geliştirmesi HTML etiketlerine benzer CFML etiketleri ve CFScript adı verilen, sözdizimi Java diline benzer yüksek seviyeli bir dil ile yapılmaktadır. CF'un yapısı ve geliştirmesi oldukça kolaydır. Dil için sağlanan destek, teknolojinin verimliliği ve Adobe Flash gibi ürünler ile direk bağlantılı olarak çalışabilmesi CF'u diğer diller kadar esnek ve güçlü yapmaktadır. Adobe firmasının verdiği gelişmiş destek sayesinde MSSQL, MySQL, Oracle, PostgreSQL, MS Access, FoxPro ve Paradox gibi birçok veri tabanı yönetim sistemi ile sorunsuz olarak

kullanılabilmektedir. CF gelişmiş bir ağ betiği olması nedeniyle pek çok profesyonel tarafından birçok yazılım projesinde tercih edilmektedir. CF ile dinamik web siteleri, e-ticaret siteleri, ağ uygulamaları ve bu uygulamalara ait gelişmiş raporlama araçları geliştirilebilmektedir (URL-3).

3.1.2. MSSQL

Microsoft Structured Query Language (MSSQL), Microsoft tarafından geliştirilen bir ilişkisel veri tabanı yönetim sistemidir. İlk geliştiricisi olan Sybase, Microsoft tarafından satın alınmıştır. Sybase 3.0 versiyonu Microsoft SQL Server 1.0'ın temelini oluşturmuştur. İlk sürümü 1989 yılında, OS/2 üzerinde 16 bit olarak geliştirilmiştir. Bir veri tabanı sunucusu olarak MSSQL aynı yazılım üzerinden veya genel ağ dahil tüm ağlarda çalışan diğer yazılım uygulamaları tarafından talep edilen verileri depolama, geri çağırma, yedekleme gibi işlevleri gerçekleştirir. Microsoft tarafından geliştirilen MSSQL; farklı tekil kullanıcılar, tek makineli küçük uygulamalar ile genel ağ üzerinden çalışan birçok eşzamanlı kullanıcıya sahip büyük uygulamalar için pek çok sistem üzerinde farklı kitlelerce ve projelerde kullanılabilir. MSSQL; Microsoft'un profesyonel veri tabanı konusundaki en yaygın kullanılan ürünüdür, ayı zamanda Microsoft Azure üzerinde platform uygulaması şeklinde kullanılmaktadır. Tüm sürümleri ücretli olan MSSQL'in, MSSQL Express sürümleri bazı erişim ve özellik kısıtları ile ücretsiz olarak kullanılabilir (URL-4) (URL-5) (URL-6).

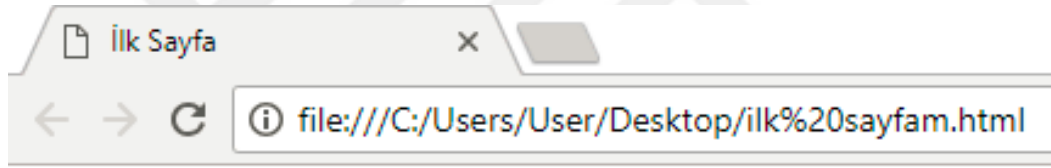
3.1.3. HTML

Bağlantılı Metin İşaretleme Dili (HTML), 1980 yılında CERN'de araştırmacıların dokümanlarını kurum içerisinde paylaşabilmeleri için Tim Berners-Lee tarafından ENQUIRE ismiyle önerilmiştir. 1990 yılında geliştirilmiş ve HTML adını almıştır (URL-7). CERN'de veri paylaşımı işlemini kolaylaştırma amacıyla doğan HTML günümüzde hayatımızın vazgeçilmezi olan genel ağın temelini oluşturmuştur. Temel amacı metin, resim, video gibi farklı verileri birbirine basitçe ve hızlıca bağlamak olan HTML, bir programlama dili olarak tanımlanmamaktadır. Bunun sebebi HTML'in komutlarıyla “.exe” uzantısına benzer şekilde çalıştırılabilir programlar geliştirilememesidir. Ağ tarayıcıları HTML komutlarını okur, yorumlar ve kullanıcıların anlayacağı görseller haline dönüştürür. Bu nedenle HTML dosyalarının

yorumlanmasında ağ tarayıcılarının farklılıklarından kaynaklanan bazı özelliklerin istenenden farklı çalışması ya da hiç çalışmaması gibi durumlar ile karşılaşılabilir. HTML kodlarının basamaklı sitil sayfası (CSS) ve JavaScript ile birlikte kullanımıyla görsel ve dinamik web siteleri oluşturulabilmektedir. HTML kodları “<” ve “>” karakterleri arasında yazılan özel etiketler ile kullanılmaktadır. Şekil 3.1’deki HTML kodu örneğinin çıktısı Şekil 3.2’deki gibi olacaktır.

```
<html> //HTML dosyası başlangıcı
  <head> //Sayfa içerisinde görünmeyen başlık bilgileri tanımlamaları başlangıcı
    <title>İlk Sayfa</title> //Sayfa başlığında görüntülenecek bilgi
  </head> //Başlık tanımlamaları sonu
  <body> //Sayfa içerisinde gösterilecek gövde içeriği başlangıcı
    <h3>Merhaba Dünya!</h3> //Sayfa içeriği, 3. Seviye başlık metni
  </body> //Gövde sonu
</html> //HTML dosyası sonu
```

Şekil 3.1. HTML kodu örneği



Merhaba Dünya!

Şekil 3.2. Örnek HTML sayfası çıktısı

3.1.4. JavaScript

JavaScript (JS), Netscape firması tarafından etkileşimli web siteleri yaratmak amacıyla C dilinden türetilmiştir. JS kodları derlenmez. HTML’e benzer biçimde düz yazı dosyası olarak kaydedilir. JS kodları, HTML kodlarının içine <script> </script> etiketlerinin arasına yazılır ve ağ tarayıcısına bağlı olarak yorumlanır. JS, bir betik dili olması nedeniyle son derece esnek bir dildir. JS çalışabilmek için komutları yorumlayabilecek bir ağ tarayıcısına ihtiyaç duyar. JS, yorumlanan bir dil olduğundan kendisi bir takım nesnelere oluşturmaz; kendisini yorumlayan tarayıcının nesnelere kullanır. Bu sebeple JS kullanılarak “.exe” uzantılı dosyaların benzeri, çalıştırılabilir programlar üretilmez. Bazı JS komutları veya kod parçacıkları Firefox, Google Chrome, Opera, Microsoft Edge gibi farklı tarayıcılarda farklı biçimlerde yorumlanabilir. Bu nedenle zaman zaman bir tarayıcıda çalışan kod bloğu diğer

tarayıcıda çalışmayabilir. Tarayıcıya göre yorumlanması JS'in zayıf yönüdür (URL-8).

3.1.5. JSON

JavaScript Object Notation (JSON), JavaScript uygulamalarında kullanmak amacıyla geliştirilmiş bir veri formatıdır. JavaScript Object Notation'ın baş harflerinin kısaltması ile isimlendirilmiştir. Ortaya çıkışının temel amacı veri aktarımında JSON kullanarak XML'den daha az yer kaplamasını sağlamaktır. Günümüzde yalnızca JS uygulamalarında değil, pek çok yazılım teknolojisinde JSON formatındaki veri aktarımı kullanılmaktadır. JSON formatında veriler anahtar ve değer olmak üzere iki kısımdan meydana gelmektedir. Anahtarlar, programlama dillerindeki değişken isimleri gibi bir özelliği adlandırılması için kullanılır ve string veri tipindedir. Değerler ise değişkenlerin tuttuğu değerler gibi içeriği tutmaktadır ve karakter, mantıksal ya da sayısal gibi herhangi bir veri tipinde olabilir. Aşağıda XML ve JSON veri formatları ile ilgili örnekler gösterilmektedir. Şekil 3.3'te XML formatındaki veri gösterilmektedir. Örnekten de anlaşılacağı üzere HTML'de olduğu gibi, hiyerarşik ve etiketlerin açılması ve kapatılması gibi pek çok akan veriyi arttıran yapı kullanılmaktadır.

```
<?xml version="1.0" encoding="UTF-8"?>
<kisiler>
  <kisi>
    <adi>Samet</adi>
    <soyadi>Diri</soyadi>
    <yasi>29</yasi>
  </kisi>
  <kisi>
    <adi>Zeynep İdil</adi>
    <soyadi>Diri</soyadi>
    <yasi>1</yasi>
  </kisi>
</kisiler>
```

Şekil 3.3. XML veri formatı örneği

Şekil 3.4'te ise JSON formatı gösterilmektedir.

```
{ "kisiler": {  
  "kisi": [  
    {  
      "adi": "Samet",  
      "soyadi": "Diri",  
      "yasi": "29"  
    },  
    {  
      "adi": "Zeynep İdil",  
      "soyadi": "Diri",  
      "yasi": "1"  
    }  
  ]  
}}
```

Şekil 3.4. JSON veri formatı örneği

JSON formatındaki veriyi yorumlamak ve işlemek XML'e göre daha kolaydır. Nedeni, söz diziminin programlama dillerindeki yapılara ve dizilere benzemesidir. Bunlara ek olarak, örnekte de görüleceği gibi daha az veri aktararak aynı veriler aktarılmaktadır. JSON'u en avantajlı kılan nokta da az veri aktarımı gerçekleşmesidir.

3.2. Uygulama Alt Yapısı ve Ara Yüzü

Akıllı Soru Bankası Sistemi (ASBS) iki aşamadan oluşmaktadır. İlk aşama, sınav öncesi yapılması gereken işlemlerin olduğu kısımları, ikinci aşama ise sınav sonrası yapılması gereken işlemlerin olduğu kısımları içermektedir. Genel olarak, sınav öncesi aşamasında; konu, soru ve sınav oluşturma işlemleri, sınav sonrası aşamasında ise; sınavda uygulanan soruların değerlendirilmesi ve sonuçlara göre soru değerlerinin güncellenmesi işlemlerinden meydana gelmektedir.

3.2.1. Veri Tabanı Tasarımı

Tez çalışması kapsamında tasarladığımız ASBS, veri tabanı yönetim sistemi (VTYS) olarak MSSQL kullanmaktadır. VTYS olarak MSSQL seçilmiş olmasının sebebi Veri İşleme Dilinin (DML) geliştirme sırasındaki sağladığı kolaylıklar ve CF'nin MSSQL için sağladığı destektir. Veri tabanı, ünvandan öğrencilerin sorulara verdikleri cevaplara kadar ASBS üzerinde girilen tüm bilgileri saklamaktadır ve işlenmesini kolaylaştırmaktadır. ASBS'de ait veri tabanı tasarımı şablonu Şekil 3.5'te gösterilmiştir.



Şekil 3.5. ASBS veri tabanı şablonu

ASBS veri tabanında 8 tablo bulunmaktadır. Bunlar “ünvan”, “kullanici”, “konular”, “soru”, “cevaplar”, “sinav”, “sinavsorulari” ve “ogrencicevaplari” tablolarıdır.

3.2.1.1. “Unvan” Tablosu

“Unvan” tablosu, kullanıcıların ana giriş ekranlarında ve sınav kağıtlarında gösterilen ünvanların bulunduğu tablodur. Bu tablo içerisinde “unvanNo” ve “unvanAdi” alanları bulunmaktadır.

“UnvanNo” alanı, ünvan sayısının çok fazla olamayacağı düşünülerek tinyint veri tipinde tanımlanmıştır ve “unvan” tablosunun birincil anahtarıdır. “kullanici” tablosunda var olan “unvanNo” alanı ile de yabancı anahtar ilişkisi bulunmaktadır.

“UnvanAdi” alanı, ünvanı temsil ettiği için varchar tipindedir ve çok uzun ünvanlar olamayacağı düşünülerek 50 karakter uzunluğunda tanımlanmıştır.

3.2.1.2. “Kullanici” Tablosu

“Kullanici” tablosu, kullanıcıların ana giriş ekranlarında ve sınav kağıtlarında gösterilen bilgileri ile oturum açmak için kullandıkları parola, e-posta bilgilerinin bulunduğu tablodur. Bu tablo içerisinde “kullaniciNo”, “kullaniciAdi”, “kullaniciSoyadi”, “unvanNo”, “kullaniciParola” ve “kullaniciEPosta” alanları bulunmaktadır.

“kullaniciNo” alanı, kullanıcı sayısının çok olabileceği düşünülerek int veri tipinde tanımlanmıştır ve “kullanici” tablosunun birincil anahtarıdır. “konular” ve “sinavlar” tablolarında var olan “kullaniciNo” alanları ile de yabancı anahtar ilişkisi bulunmaktadır.

“kullaniciAdi” ve “kullaniciSoyadi” alanları, isim ve soy isim bilgilerini temsil ettiği için varchar tipindedir ve çok uzun ünvanlar olamayacağı düşünülerek 50 karakter uzunluğunda tanımlanmıştır.

“unvanNo” alanı, “unvan” tablosunun yabancı anahtarı olduğu için “unvan” tablosundaki “unvanNo” alanının özellikleri ile aynıdır.

“kullaniciParola” ve “kullaniciEPosta” alanları, oturum açarken kullanılacak olan parola ve parolanın unutulması durumunda doğrulama için kullanılacak olan e-posta bilgilerini temsil ettiği için varchar tipindedir ve çok uzun parola ve e-posta adresleri olamayacağı düşünülerek 50 karakter uzunluğunda tanımlanmıştır.

3.2.1.3. “Konular” Tablosu

“Konular” tablosu, kullanıcıların tanımlayacağı soruları ilişkilendirdikleri ve sınav oluştururken kullanılacak olan konuların bulunduğu tablodur. Bu tablo içerisinde “konuNo”, “konuAdi”, “kullaniciNo”, “silindi” ve “eklenmeTarihi” alanları bulunmaktadır.

“konuNo” alanı, konu sayısının çok olabileceği düşünülerek int veri tipinde tanımlanmıştır ve “konular” tablosunun birincil anahtarıdır. “soru” tablosunda var olan “konuNo” alanı ile de yabancı anahtar ilişkisi bulunmaktadır.

“konuAdi” alanı, konun isim bilgisini temsil ettiği için varchar tipindedir 200 karakter uzunluğunda tanımlanmıştır.

“kullaniciNo” alanı, “kullanici” tablosunun yabancı anahtarı olduğu için “kullanici” tablosundaki “kullaniciNo” alanının özellikleri ile aynıdır.

“silindi” alanı, tanımlanmış olan bir konunun silinip silinmediği bilgisini temsil ettiği için bit tipindedir ve konu eklendiğinde varsayılan olarak 0 değerini almaktadır. Konunun silinmesi halinde değer 1 olarak güncellenmektedir.

“eklenmeTarihi” alanı, konunun eklendiği tarihin zaman etiketi bilgisini temsil ettiği için datetime tipindedir ve konu eklendiğinde varsayılan değer olarak SQL’in “getDate()” metodunun geri dönüş değerini almaktadır.

3.2.1.4. “Soru” Tablosu

“soru” tablosu, kullanıcıların tanımlayacağı soru gövdelerinin ve soruya ait bazı değerlerin bulunduğu tablodur. Bu tablo içerisinde “soruNo”, “soruKoku”, “konuNo”, “zorluk”, “çeldirici”, “frekans”, “resim”, “silindi” ve “eklenmeTarihi” alanları bulunmaktadır.

“soruNo” alanı, soru sayısının çok olabileceği düşünülerek int veri tipinde otomatik artan sayı olarak tanımlanmıştır ve “soru” tablosunun birincil anahtarıdır. “cevaplar”, “sinavSorulari” ve “ogrenciCevaplari” tablolarında var olan “soruNo” alanları ile de yabancı anahtar ilişkisi bulunmaktadır.

“soruKoku” alanı, sorunun gövdesini temsil ettiği için varchar tipinde 2000 karakter uzunluğunda tanımlanmıştır.

“konuNo” alanı, “konular” tablosunun yabancı anahtarı olduğu için “konular” tablosundaki konuNo” alanının özellikleri ile aynıdır.

“zorluk” ve “çeldirici” alanları, sorunun zorluk ve ayırt edicilik değerlerini temsil ettiği için float veri tipindedir. Her ne kadar ilk değer tanımlaması yapıldığında kullanıcı tarafından girilen değerler tam sayı olsa da, sınavlardan sonra yapılan hesaplamalar sonucunda bulunan değerler kayan noktalı sayı olabileceği için float veri tipi tercih edilmiştir.

“frekans” alanı, sorunun daha önce sınavlarda sorulma sayısını temsil ettiği için tinyint veri tipindedir. Soru, sınavda sorulmak üzere seçildiğinde ASBS tarafından frekans değeri 1 arttırılmaktadır.

“silindi” alanı, tanımlanmış olan bir sorunun silinip silinmediği bilgisini temsil ettiği için bit tipindedir ve soru eklendiğinde varsayılan olarak 0 değerini almaktadır. Sorunun silinmesi halinde değer 1 olarak güncellenmektedir.

“eklenmeTarihi” alanı, sorunun eklendiği tarihin zaman etiketi bilgisini temsil ettiği için datetime tipindedir ve soru eklendiğinde varsayılan değer olarak SQL’in “getDate()” metodunun geri dönüş değerini almaktadır.

3.2.1.5. “Cevaplar” Tablosu

“cevaplar” tablosu, kullanıcıların tanımlayacağı soruların cevap şıklarının bulunduğu tablodur. Bu tablo içerisinde “cevapNo”, “soruNo”, “cevapAdi”, “cevapKoku”, “dogruCevap”, “resim”, “silindi” ve “eklenmeTarihi” alanları bulunmaktadır.

“cevapNo” alanı, her bir soru için 5 tane cevap şıkkı olacağı ve soru sayısının çok olabileceği düşünülerek int veri tipinde otomatik artan sayı olarak tanımlanmıştır ve “cevaplar” tablosunun birincil anahtarıdır.

“soruNo” alanı, “soru” tablosunun yabancı anahtarı olduğu için “soru” tablosundaki “soruNo” alanının özellikleri ile aynıdır.

“cevapAdi” alanı, sorunun seçeneğinin sırasını (A, B, C, D, E) temsil ettiği için char veri tipinde, 1 karakter uzunluğunda tanımlanmıştır.

“cevapKoku” alanı, cevap seçeneğinin gövdesini temsil ettiği için varchar tipinde, 2000 karakter uzunluğunda tanımlanmıştır.

“dogruCevap” alanı, tanımlanmış olan cevap şıkkının doğru-yanlış durumunu temsil ettiği için bit tipindedir ve yanlış cevaplar 0, doğru cevaplar ise 1 değerini almaktadır.

“silindi” alanı, tanımlanmış olan bir cevabın silinip silinmediği bilgisini temsil ettiği için bit tipindedir ve cevap eklendiğinde varsayılan olarak 0 değerini almaktadır. Cevabın silinmesi halinde değer 1 olarak güncellenmektedir.

“eklenmeTarihi” alanı, cevabın eklendiği tarihin zaman etiketi bilgisini temsil ettiği için datetime tipindedir ve cevap eklendiğinde varsayılan değer olarak SQL’in “getDate()” metodunun geri dönüş değerini almaktadır.

3.2.1.6. “Sinav” Tablosu

“Sinav” tablosu, kullanıcıların tanımlayacağı sınavların özelliklerinin bulunduğu tablodur. Bu tablo içerisinde “sinavNo”, “sinavAdi”, “kullaniciNo”, “sinavTarihi”, “kitapcikSayisi”, “kitapcikBaslangicKodu”, “soruSayisi”, “zorluk”, “çeldirici”, “frekans”, “sinavSuresi”, “konuListesi”, “silindi” ve “eklenmeTarihi” alanları bulunmaktadır.

“sinavNo” alanı, ASBS’de kayıtlı çok fazla kullanıcının tanımlayacağı sınav sayısının çok olabileceği düşünülerek int veri tipinde otomatik artan sayı olarak tanımlanmıştır ve “sinav” tablosunun birincil anahtarıdır. “sinavSorulari” ve “ogrenciCevaplari” tablolarında var olan “sinavNo” alanları ile de yabancı anahtar ilişkisi bulunmaktadır.

“sinavAdi” alanı, sınavın belgelerde gösterilecek olan ismini temsil ettiği için varchar tipinde 200 karakter uzunluğunda tanımlanmıştır.

“kullaniciNo” alanı, “kullanici” tablosunun yabancı anahtarı olduğu için “kullanici” tablosundaki “kullaniciNo” alanının özellikleri ile aynıdır.

“sinavTarihi” alanı, sınavın düzenleneceği tarihin bilgisini temsil ettiği için datetime veri tipindedir.

“kitapcikSayisi” alanı, sınavda uygulanacak kitapçıkların sayısını temsil ettiği için tinyint veri tipindedir.

“kitapcikBaslangicKodu” alanı, sınavda uygulanacak kitapçıkların hangi harf kodu ile başlayacağını temsil ettiği için char veri tipinde, 1 karakter uzunluğunda tanımlanmıştır.

“soruSayisi” alanı, sınavda sorulacak ve ASBS tarafından seçilmesi istenen soru sayısını temsil ettiği için tinyint veri tipindedir.

“zorluk” ve “çeldirici” alanları, sınavın ortalama zorluk ve ayırt edicilik değerlerini temsil ettiği için float veri tipindedir. Her ne kadar kullanıcı tarafından ara yüz üzerinden yapılan seçimler ilk aşamada tam sayı olsa da, ASBS geliştirildiği takdirde bu değerler kayan noktalı sayı olabileceği için float veri tipi tercih edilmiştir.

“frekans” alanı, seçilmesi istenen soruların daha önceki sınavlarda sorulma sayısını temsil ettiği için tinyint veri tipindedir.

“sinavSuresi” alanı, sınav kitapçığında gösterilecek olan sınavın süresini temsil ettiği için tinyint veri tipindedir.

“konuListesi” alanı, ASBS tarafından sınavda soru seçilmesi istenen konuların listesini temsil ettiği için varchar veri tipinde, 100 karakter uzunluğunda tanımlanmıştır. Bu alanda, CF ile işlemesi kolay olduğu için konu numaraları liste şeklinde tutulmaktadır.

“silindi” alanı, tanımlanmış olan bir sınavın silinip silinmediği bilgisini temsil ettiği için bit tipindedir ve sınav eklendiğinde varsayılan olarak 0 değerini almaktadır. Sınavın silinmesi halinde değer 1 olarak güncellenmektedir.

“eklenmeTarihi” alanı, sınavın oluşturulduğu tarihin zaman etiketi bilgisini temsil ettiği için datetime tipindedir ve sınav oluşturulduğunda varsayılan değer olarak SQL’in “getDate()” metodunun geri dönüş değerini almaktadır.

3.2.1.7. “SinavSorulari” Tablosu

“SinavSorulari” tablosu, kullanıcıların oluşturduğu sınavların kitapçıklarına dağıtılan soruların bilgilerinin bulunduğu tablodur. Bu tablo içerisinde “sinavSoruNo”, “sinavNo”, “kitapcikKodu”, “kitapcikSoruNo”, “soruNo” ve “eklenmeTarihi” alanları bulunmaktadır.

“sinavSoruNo” alanı, çok fazla sayıda sınav için çok fazla kitapçık oluşturulabileceği planlanarak int veri tipinde otomatik artan sayı olarak tanımlanmıştır ve “sinavSorulari” tablosunun birincil anahtarıdır.

“sinavNo” alanı, “sinav” tablosunun yabancı anahtarı olduğu için “sinav” tablosundaki “sinavNo” alanının özellikleri ile aynıdır.

“kitapcikKodu” alanı, sınavda uygulanacak her bir kitapçığın harf kodunu temsil ettiği için char veri tipinde, 1 karakter uzunluğunda tanımlanmıştır.

“kitapcikSoruNo” alanı, sorunun kitapçığın kaçınıcı sorusu olduğunu temsil ettiği için tinyint veri tipindedir.

“soruNo” alanı, “soru” tablosunun yabancı anahtarı olduğu için “soru” tablosundaki “soruNo” alanının özellikleri ile aynıdır.

“eklenmeTarihi” alanı, sınavın oluşturulduğu tarihin zaman etiketi bilgisini temsil ettiği için datetime tipindedir ve sınav oluşturulduğunda varsayılan değer olarak SQL’in “getDate()” metodunun geri dönüş değerini almaktadır.

3.2.1.8. “OgrenciCevaplari” Tablosu

“OgrenciCevaplari” tablosu, kullanıcıların oluşturduğu sınavların değerlendirmelerinin bulunduğu tablodur. Bu tablo içerisinde “ogrenciCevapNo”, “ogrenciNo”, “kitapcikKodu”, “sinavNo”, “soruNo”, “cevap” ve “dogruCevap” alanları bulunmaktadır.

“ogrenciCevapNo” alanı, çok fazla sayıda sınav için çok fazla sınav değerlendirmesi yapılabileceği planlanarak int veri tipinde otomatik artan sayı olarak tanımlanmıştır ve “ogrenciCevaplari” tablosunun birincil anahtarıdır.

“ogrenciNo” alanı, sınavı değerlendirilen öğrencinin numarasını temsil ettiği için int veri tipindedir.

“kitapcikKodu” alanı, sınavda uygulanan kitapçığın harf kodunu temsil ettiği için char veri tipinde, 1 karakter uzunluğunda tanımlanmıştır.

“sinavNo” alanı, “sinav” tablosunun yabancı anahtarı olduğu için “sinav” tablosundaki “sinavNo” alanının özellikleri ile aynıdır.

“soruNo” alanı, “soru” tablosunun yabancı anahtarı olduğu için “soru” tablosundaki “soruNo” alanının özellikleri ile aynıdır.

“cevap” alanı, öğrencinin sınavda işaretlediği sorunun seçeneğini temsil ettiği için char veri tipinde, 1 karakter uzunluğunda tanımlanmıştır.

“dogruCevap” alanı, öğrenciye sınavda sorulan sorunun sistem üzerindeki doğru cevabını temsil ettiği için char veri tipinde, 1 karakter uzunluğunda tanımlanmıştır. Bu alanın tanımlanmış olması, soru üzerinde herhangi bir düzenleme yapılırken doğru cevap seçeneğinin değiştirilmesi durumunda geriye dönük değerlendirme işlemi yapılabilmesine olanak tanımaktadır.

3.2.2. Sınav Öncesi Yapılması Gereken İşlemler

Sınav sürecinin yönetilmesi işlemlerinde öncelikli olarak hazırlanacak olan sınavda kullanılacak soruların tanımlanması ve sınav kitapçığının oluşturulması gerekmektedir.

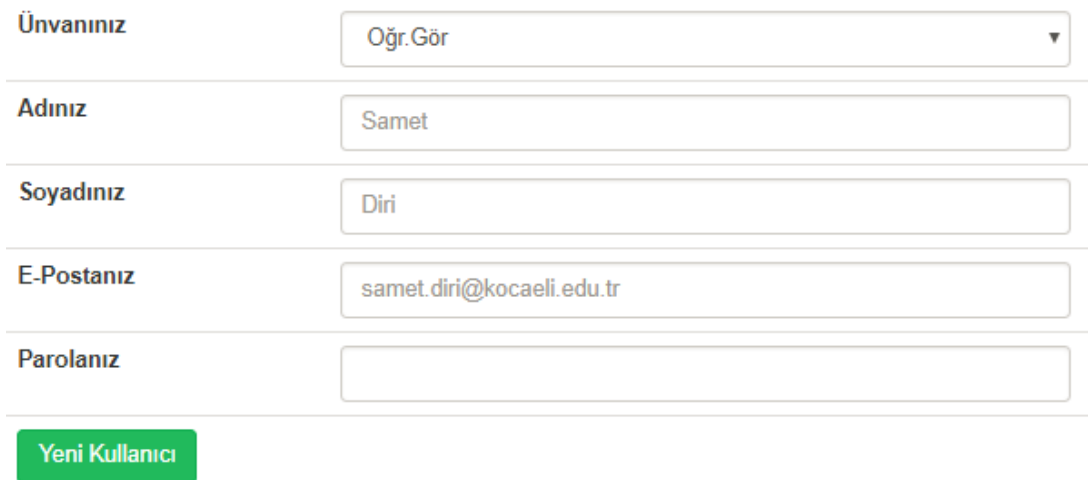
3.2.2.1. Kullanıcı Kaydı Oluřturma

ASBS’de her kullanıcının kendine ait bir hesabı bulunmaktadır. Tüm süreç kullanıcılara ait kişisel hesaplar üzerinden işletilmektedir. Her bir kullanıcı yalnızca kendi hesabına ait veri giriři, düzenleme ve görüntüleme işlemlerini yapabilmektedir. Bu sebeple tüm kullanıcıların Şekil 3.6’da gösterilen kullanıcı oturum açma ekranını kullanarak oturum açması gerekmektedir. Böylece kullanıcılar kişisel hesaplarına ve hesaplarında tanımlamış oldukları tüm verilere erişebileceklerdir. Şekil 3.6’da görüldüğü üzere henüz kayıt işlemini gerçekleştirmemiş olan kullanıcılar “Kayıt Ol” bağlantısını kullanarak kendilerine ait bir kullanıcı kaydı oluşturabilmektedirler.



Şekil 3.6. Kullanıcı oturum açma ekranı

“Kayıt Ol” bağlantısını tıklayan kullanıcı Şekil 3.7’deki yeni kullanıcı kaydı oluřturma ekranına yönlendirilecektir.



Şekil 3.7. Yeni kullanıcı kaydı oluřturma ekranı

Burada, yeni kullanıcı kaydı oluşturulurken, sınav kitapçıklarında ve kişisel sayfasının giriş ekranında görüntülemek amacıyla kullanıcının ünvanı, adı ve soyadı bilgileri ile birlikte kişisel hesabına erişmek için kullanacağı parola bilgisini, parolasını unutması durumunda parolasını güncellemek için iletişim kurabileceği e-posta adresini tanımlaması gerekmektedir. Bir e-posta adresi ile yalnızca bir kez kayıt olunabilmektedir. Kullanıcı, gerekli alanları doldurduktan sonra sayfanın alt kısmında yer alan yeşil renkli “Yeni Kullanıcı” düğmesine tıklayarak yeni kullanıcı kayıt işlemini tamamlamalıdır. Kullanıcı, yeni kullanıcı kayıt işlemi tamamlandıktan sonra, Şekil 3.8’de gösterilen kayıt başarılı ekranı ile karşılaşılacaktır ve ekranda bulunan “Giriş için tıklayınız” bağlantısını tıklayarak Şekil 3.6’da gösterilen kullanıcı oturum açma ekranına yönlendirilecektir.

Kullanıcı Numaranız: 5
Parolanız: 1234
Giriş için tıklayınız.

Şekil 3.8. Yeni kullanıcı kaydı tamamlandı ekranı

Kayıt sırasında girilen e-posta adresi daha önce sisteme farklı bir kullanıcı tarafından oluşturulan hesapta kullanılmış ise Şekil 3.9’da gösterilen uyarı mesajı kullanıcıya gösterilmektedir.

Girdiğiniz e-posta adresi başka bir hesap ile ilişkilendirilmiş durumdadır.
Eğer parolanızı hatırlamıyorsanız samet.diri@kocaeli.edu.tr adresine e-
posta göndermeniz gerekmektedir.

Şekil 3.9. E-posta adresi sistemde kayıtlı ekranı

Kayıt işlemi gerçekleştirilirken herhangi bir hata meydana geldiğinde ise Şekil 3.10’da gösterilen hata mesajı kullanıcıya gösterilmektedir.

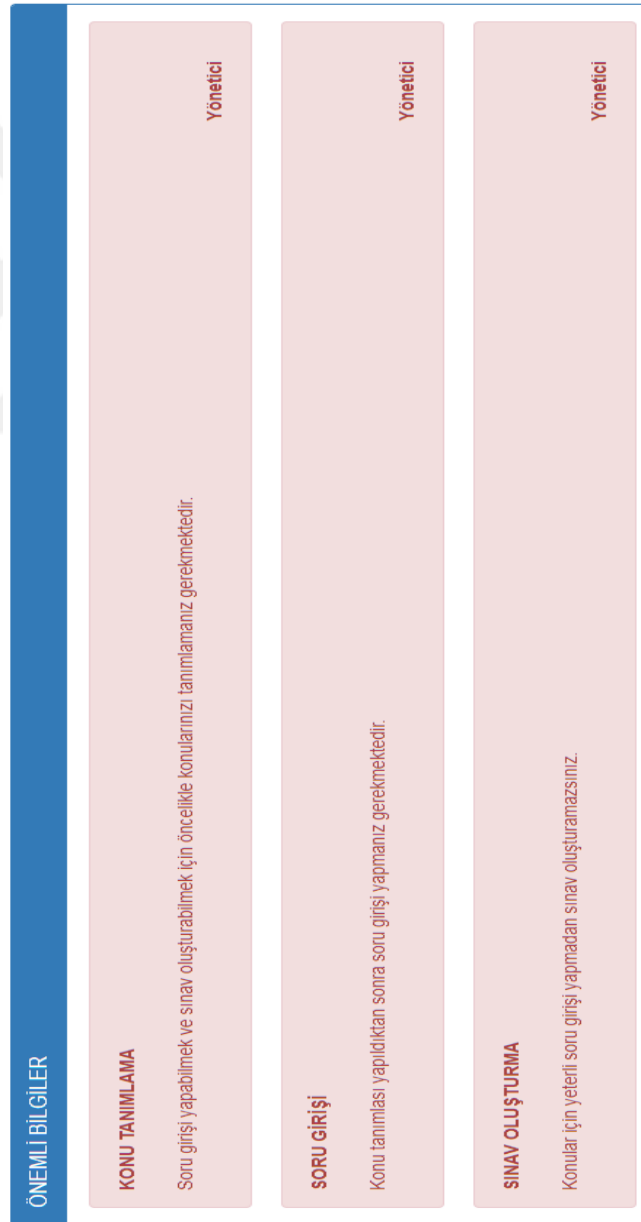
Kullanıcı kaydınız oluşturulamadı!. Lütfen tekrar deneyiniz.

Şekil 3.10. Kullanıcı kaydı hata mesajı

Kullanıcının kendisine ait parolayı unutması durumunda, kayıt esnasında tanımladığı e-posta adresinden Şekil 3.9’da gösterilen “samet.diri@kocaeli.edu.tr” adresine e-posta göndermesi gerekmektedir.

3.2.2.2. Oturum Açma ve Ana Giriş Ekranı

Kullanıcı kodu ve parolasını doğru bir şekilde giren kullanıcı Şekil 3.11’de gösterilen kullanıcı ana giriş ekranına yönlendirilmektedir. Ana giriş ekranının üst kısmında ana menü bulunmaktadır. Ana menüde “Ana sayfa” ve “Çıkış” bağlantıları ile “Konu İşlemleri”, “Soru İşlemleri”, “Sınav İşlemleri” menüleri bulunmaktadır. Kullanıcılar yapmak istedikleri işlemlere ilgili menüleri kullanarak erişebilmektedir. Menülerin sağ tarafında ise oturumu açık olan kullanıcının bilgileri gösterilmektedir. Ana menü tüm sayfalarda ortak şekilde gösterilmektedir.

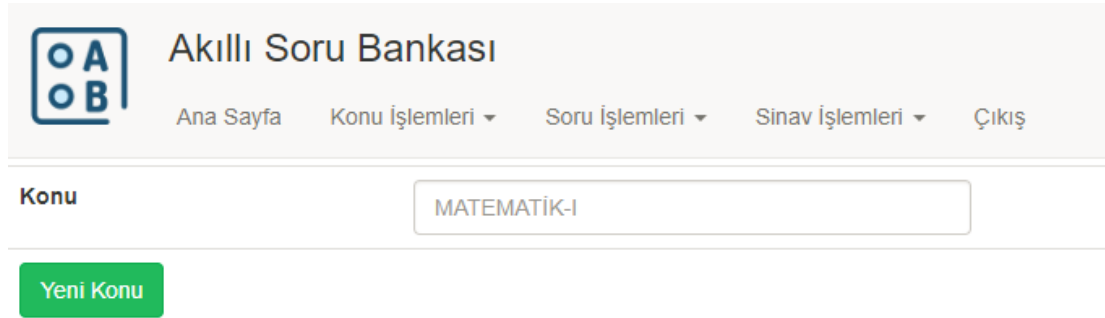


Şekil 3.11. Kullanıcı ana giriş ekranı

Ana giriş ekranının içerisindeki “Önemli Bilgiler” bölümünde ise sistemin kullanılması ile ilgili yönergeler, duyurular ve bilgilendirmeler bulunmaktadır. ASBS geliştirildikçe “Önemli Bilgiler” kısmında içerik de değiştirilebilir.

3.2.2.3. Konu Tanımlama

Konu tanımlama işlemi için Şekil 3.12’de bulunan konu tanımlama ekranı kullanılmaktadır. Kullanıcılar, konu tanımlama işlemi ekranına sayfanın üst kısmında bulunan ana menünün “Konu İşlemleri” sekmesinin altındaki “Konu Ekle” bağlantısına tıklanarak erişilebilmektedir. Burada ilgili konunun ismi kutucuğa girilmelidir ve ardından “Yeni Konu” düğmesine basılarak konu ekleme işlemi gerçekleştirilmelidir. Her bir kullanıcı kendi hesabı ile ilişkilendirilmiş sisteme sınırsız sayıda konu tanımlaması yapabilmektedir. Konu isimleri 200 karakterden uzun olmamalıdır, MSSQL’in varchar veri tipinin desteklediği karakterlerin tamamının girişi mümkündür. Konu adları büyük harf olarak tanımlanmalıdır. ASBS, kullanıcı küçük harf girerse dahi HTML’in sunduğu büyük harfe dönüştürme işlemi otomatik olarak gerçekleştirmektedir. Bu sayede kullanıcının veri girişi sırasında göz ardı edebileceği bazı harflerin büyük harf olarak tanımlanması gerekirken küçük, bazılarının küçük harf olarak tanımlanması gerekirken büyük harf olarak girilmesi gibi şekilsel bozuklukların önüne geçilmesi amaçlanmaktadır.



The screenshot shows the 'Akıllı Soru Bankası' (Smart Question Bank) interface. At the top, there is a logo with 'A' and 'B' in circles. Below the logo, the title 'Akıllı Soru Bankası' is displayed. A navigation menu contains 'Ana Sayfa', 'Konu İşlemleri', 'Soru İşlemleri', 'Sınav İşlemleri', and 'Çıkış'. The main content area has a 'Konu' label and a text input field containing 'MATEMATİK-I'. Below the input field, there is a green button labeled 'Yeni Konu'.

Şekil 3.12. Konu tanımlama ekranı

Kullanıcıların ASBS üzerinden otomatik sınav üretebilmesi için konularını ve konuları ile ilişkilendirmiş olduğu soru havuzunu tanımlamış olması gerekmektedir. Burada konu ile kastedilen, bir dersin adı veya bir derse ait konu başlığı olabilir. Örneğin: kullanıcı isterse “MATEMATİK-I” isimli bir konu tanımlaması yaparak derse ait soru girişlerinde bulunabilir. Bu konu başlığını kullanarak ASBS üzerinden Matematik-I dersi için genel sorulardan oluşan bir sınav düzenleyebilir. İkinci bir durumda ise

“FONKSİYONLAR” isimli bir konu tanımlayabilir. Fonksiyonlar konusu Matematik-I dersine göre daha özel bir konu başlığı olduğu için öğrencilerin doğrudan fonksiyonlar konusu ile ilgili bilgilerini ölçen bir sınav düzenleyebilir. Yukarıda bahsedilen iki durumun bir arada kullanılması da mümkündür. Kullanıcı, “MATEMATİK-I” ve “FONKSİYONLAR” konularını ayrı ayrı tanımlayabilir ve ASBS’yi kullanarak sınav oluşturma aşamasında, her iki konuyu da seçerek Matematik-I dersine ait genel soruların yanı sıra, fonksiyonlar konusuna özel sorulardan oluşan bir sınav düzenleyebilir. ASBS üzerinde konu tanımlaması işleminin genel olması ya da detaylı olması kullanıcının tercihine bağlıdır. Buradaki en önemli nokta, her bir sorunun belirli bir konu ile ilişkilendirilmek zorunda olmasıdır.

3.2.2.4. Konu Listeleme

Kullanıcılar ASBS üzerinde kendi hesaplarında tanımladığı tüm konuları Şekil 3.13’te gösterildiği gibi tek bir ekranda listeleyebilmektedir. Konu listeleme ekranına, sayfanın üst kısmında bulunan ana menünün “Konu İşlemleri” sekmesinin altındaki “Konu Listele” bağlantısına tıklanarak erişilebilmektedir. Bu bağlantı kullanarak erişilen sayfada, tanımlanmış olan tüm konuların isimleri, her bir konuya ait soru sayısı ve konuların oluşturulma tarihleri görüntülenebilmektedir. Bunlara ek olarak, her bir konunun düzeltme ve silme sayfaları ile yeni bir konu ekleme sayfasına erişim bağlantıları bulunmaktadır. Kullanıcılar, kendi konularına ait işlemleri bu sayfa aracılığıyla yönetebilmektedir. Düzeltme işlemi gerçekleştirebilmek için ilgili konunun bulunduğu satır üzerindeki sarı renkli düğmeye, silme işlemi için ise ilgili satırdaki kırmızı renkli düğmeye tıklanmalıdır. Yeni konunun eklenmesi için konu listeleme sayfasından hızlı erişim sağlanabilmektedir. Bunun yapabilmesi için listenin sağ üst köşesinde bulunan yeşil renkli düğmeye tıklanmalıdır.



Akıllı Soru Bankası













Ana Sayfa Konu İşlemleri ▾

Soru İşlemleri ▾

Sınav İşlemleri ▾

Çıkış

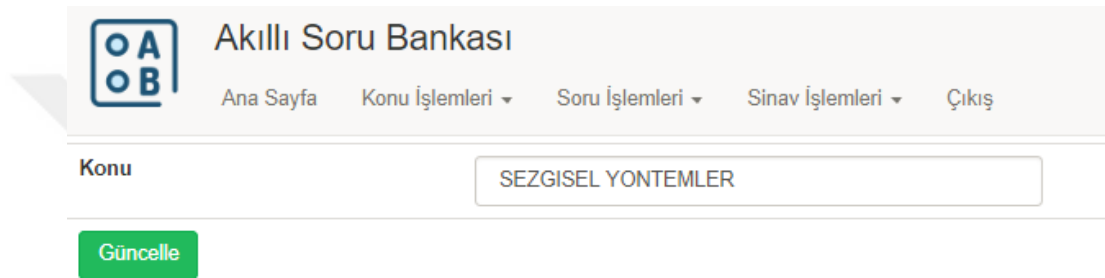
Öğr.Gör Samet Diri

Konu Adı	Soru Sayısı	Eklenme Tarihi	
FONKSİYONLAR	6	03/04/2018	 
BILGISAYAR PROGRAMLAMA II	5	04/04/2018	 
GENETİK ALGORITMA	1	02/04/2018	 
OPTİMİZASYON TEORİSİ	6	03/04/2018	 
TEMEL BILGISAYAR BILGISI	6	09/04/2018	 
YAPAY SINIR AĞLARI	5	03/04/2018	 

Şekil 3.13. Kullanıcının tanımlı konularını listeleme ekranı

3.2.2.5. Konu Güncelleme

Bir önceki bölümde de bahsedildiği gibi, konu listeleme ekranında düzeltilmek istenen konu başlığının bulunduğu satırdaki sarı renkli düğme kullanılarak Şekil 3.14’de gösterilen konu güncelleme ekranına erişim sağlanabilmektedir. Sayfa yüklendiğinde hatalı olan konu başlığı metin kutusu içinde girilmiş durumda olacaktır. Konu başlığı üzerindeki hatalı kısmın düzeltildikten sonra “Güncelle” düğmesine tıklanarak düzeltme işlemi tamamlanmalıdır. Kullanıcı, düzeltme işlemi tamamlandıktan sonra, ASBS tarafından otomatik olarak konu listeleme ekranına yönlendirilecektir.



Şekil 3.14. Kullanıcının tanımlı konularını listeleme ekranı

3.2.2.6. Konu Silme

Konu silme bağlantısına, daha önceki sınavlarda hiç kullanılmamış konular için erişilebilmektedir. Konu silme bağlantısına tıkladığında, ilgili konuya ait silme onayı gelecektir. “Tamam” butonuna tıkladığında konu silme işlemi tamamlanacaktır. “İptal” butonuna tıkladığında ise silme işlemi gerçekleşmeyecektir. ASBS üzerinde konu silme işlemi gerçek anlamda veri tabanından verinin silinmesi değil, “konular” tablosundaki mantıksal tipteki “silindi” alanının “false” değerinden “true” değerine dönüştürülmesidir. Bu sayede “silindi” alanının değeri “true” olan konular listeleme ekranında görüntülenmemektedir.

3.2.2.7. Soru Tanımlama

Soru tanımlama işlemi için Şekil 3.15’te bulunan soru tanımlama ekranı kullanılmaktadır. Soru tanımlama işlemi ekranına, sayfanın üst kısmında bulunan ana menünün “Soru İşlemleri” sekmesinin altındaki “Soru Ekle” bağlantısına tıklanarak erişilebilmektedir.

Kullanıcılar, soru giriş işlemi yaparken; sorunun ait olduğu konu, soru gövdesi, çoktan seçmeli testleri diğer sınav türlerinden ayıran özelliklerinden olan çeldiricileri, doğru cevabı, kullanıcının soru hakkında ön gördüğü madde güçlük değerini ve ayırt edicilik değerini tanımlamalıdır. Sayfa üzerinde görülen tüm alanların doldurulması zorunludur.

Akıllı Soru Bankası
Ana Sayfa Konu İşlemleri Soru İşlemleri Sınav İşlemleri Çıkış

Konu Seçiniz...

Soru

A)

B)

C)

D)

E)

Öngörülen Zorluk (3) 1 5

Öngörülen Ayırt Edicilik (3) 1 5

Soru Kaydet

Şekil 3.15. Yeni soru giriş ekranı

Soru tanımlaması yapmak isteyen bir kullanıcı, öncelikle soruyu ilişkilendireceği konuyu seçmelidir; henüz hiçbir konu tanımlaması yapmamış ise soruyu kaydetmek istediğinde hata mesajı ile karşılaşacaktır.

İkinci aşamada sorunun gövdesi tanımlanmalıdır. Soru gövdesi tanımlanırken bazı hususlara dikkat edilmesi gerekmektedir. Örneğin: soru, sade ve anlaşılır bir dil ile dil bilgisi ve imla kurallarına uygun biçimde hazırlanmalıdır. Soruyu okuyan tüm öğrenciler sorudan aynı anlamı çıkartmalıdır. Bir soru ile yalnızca bir yeterlilik ölçülmelidir ve soruda gereksiz ipucu ve detay vermekten kaçınılmalıdır. Soru

konunun özüyle alakalı olmalı ve tek yanıtı olan kesin yargılar taşınmalıdır. Konuyu öğrenebilmiş olan öğrenciler soruyu doğru yanıt verebilmelidirler. Öğrenciyi geliştirici olmalıdır (Kutlu, 2003).

Üçüncü aşamada soruya ait doğru cevap ve çeldiriciler tanımlanmalıdır. ASBS’de her bir soru 5 cevap şıkkı ve bu cevap şıkları arasından yalnızca bir tane doğru cevap olacak şekilde tasarlanmıştır. Her bir şık harf ile belirtilen satırdaki boş kutucuğa girilmelidir. Doğru cevap için ise, şıkkın önündeki kutucuk işaretlenmelidir. Çeldiriciler, konuyu bilmeyen öğrencinin dikkatini çekecek nitelikte olmalıdır. Burada dikkat edilmesi gereken detaylar; çeldiricilerin doğru cevap olabilecek nitelikte kesin yargılar içermesi, anlam karmaşası yaratmayacak sadelikte ve dil ile hazırlanmış olması, hepsi, hiçbiri gibi yönlendirici içermemesi ve eşit şekilde dağıtılmış olmasıdır. Çeldiricilerin kalitesi sorunun zorluk ve ayırt edicilik seviyelerini belirleyen en önemli faktörlerdendir.

Son aşamada ise, kullanıcının hazırladığı soru için öngördüğü zorluk ve ayırt edicilik değerlerini belirlemesi gerekmektedir. Kullanıcının sisteme girdiği sorunun sınavlarda seçilebilmesi amacıyla, bu değerler, ilk değer olarak tanımlanmalıdır. Kullanıcının öngördüğü zorluk ve ayırt edicilik değerleri ile sınav uygulandıktan sonra elde edilen değerler birbirlerinden çok farklı olabilirler. Bu, konunun anlaşılma düzeyinin kullanıcının öngördüğünden farklı olduğunu ya da soru ile ilgili ciddi bir hata olabileceğini gösterir. Sınav değerlendirmesinden sonra zorluk ve ayırt edicilik değerleri verilen cevaplara göre güncellenmektedir. Zorluk indeksi Denklem (3.1)’deki formül ile hesaplanabilmektedir (Tekin, 2000).

$$P_{\text{zorluk}} = \frac{d_{\text{üst}} + d_{\text{alt}}}{N} \quad (3.1)$$

Burada, $d_{\text{üst}}$, sınavdan yüksek puan alan %27’lik öğrenci grubunu,

d_{alt} , sınavdan düşük puan alan %27’lik öğrenci grubunu,

N , alt ve üst gruptaki toplam öğrenci sayısını göstermektedir. Ancak, formül uygulandığı takdirde 0.00 ile 1.00 arasında bir sonuç elde edilecektir. Değer 0’a yaklaştıkça soru zorlaşmakta, 1’e yaklaştıkça kolaylaşmaktadır (Gelbal, 2013). Zorluk indeksinin 0.5 olması ise sorunun ortalama zorlukta olduğunu gösterir (Tekin, 2000).

ASBS'ye üye olan bazı kullanıcılar madde güçlük indeksi hesaplamayı bilmeyebilir veya zorluk derecesini yorumlamada güçlük yaşayabilir. 0.00 ile 1.00 arasında elde edilen değerler ifade ettiği zorluk derecesini algılamada güçlükler yaşanabileceği ön görülerek 0.00-1.00 aralığı; 1-çok kolay, 2-kolay, 3-orta, 4-zor ve 5-çok zor şeklinde ölçeklenmiştir.

Tablo 3.1. Zorluk indeksinin kullanıcı dostu ölçeği (Beyazşekeroğlu, 2015)

Hesaplanmış Zorluk İndeksi Aralığı	Ölçeklenen Zorluk İndeksleri
0.00-0.19	1
0.2-0.39	2
0.4-0.59	3
0.6-0.79	4
0.8-1.00	5

Ayırt edicilik indeksi ise aşağıdaki formül ile hesaplanabilmektedir (Adıgüzel ve Özüdoğru, 2013).

$$P_{\text{ayırt edicilik}} = \frac{d_{\text{üst}} - d_{\text{alt}}}{n} \quad (3.2)$$

Burada, $d_{\text{üst}}$, sınavdan yüksek puan alan %27'lik öğrenci grubunu,

d_{alt} , sınavdan düşük puan alan %27'lik öğrenci grubunu,

n , yalnızca alt veya üst gruptaki öğrenci sayısını göstermektedir. Ayırt edicilik indeksinde, zorluk indeksinden farklı olarak 0.00 ile 1.00 aralığı yerine ölçek -1.00 ile 1.00 aralığında olabilir. Değer 1'e yaklaştıkça soru ayırt edici ve iyi bir soru denilebilirken 0 ve altına indikçe kötü ve testlerden çıkartılması gereken bir soru denebilir. Yine zorluk indeksinde olduğu gibi, ASBS'ye üye olan bazı kullanıcılar madde ayırt edicilik indeksi hesaplamayı bilmeyebilir veya ayırt edicilik değerini yorumlamada güçlük yaşayabilir. -1.00 ile 1.00 arasında elde edilen değerler ifade ettiği ayırt edicilik derecesini algılamada güçlükler yaşanabileceği ön görülerek ve -1.00 ile 0.00 aralığında ayırt edicilik değerine sahip bir sorunun zaten testten atılması gerektiği varsayılarak 0.00-1.00 aralığı; 1-çok kötü, 2-kötü, 3-orta, 4-iyi ve 5-çok iyi şeklinde ölçeklenmiştir.

Tablo 3.2. Ayırt edicilik indeksinin kullanıcı dostu ölçeği (Tekin, 2000)

Hesaplanmış Ayırt Edicilik İndeksi Aralığı	Ölçeklenen Ayırt Edicilik İndeksleri
-1.00-0.00	Çıkartılmış
0.00-0.19	1
0.2-0.39	2
0.4-0.59	3
0.6-0.79	4
0.8-1.00	5

3.2.2.8. Soru Listeleme

Kullanıcılar, ASBS üzerinde kendi hesaplarında tanımladığı tüm soruları Şekil 3.16’da gösterildiği gibi tek bir ekranda listeleyebilmektedir. Soru listeleme ekranına sayfanın üst kısmında bulunan ana menünün “Soru İşlemleri” sekmesinin altındaki “Soru Listele” bağlantısına tıklanarak erişilebilmektedir. Bu bağlantı kullanarak erişilen sayfada, tanımlanmış olan tüm soruların gövdeleri, her bir sorunun ait olduğu konu, her bir soruya ait zorluk değeri ve ayırt edicilik değeri görüntülenebilmektedir. Bunlara ek olarak, her bir sorunun düzeltme ve silme sayfaları ile yeni bir soru ekleme sayfasına erişim bağlantıları bulunmaktadır. Kullanıcılar, kendi sorularına ait işlemleri bu sayfa aracılığıyla yönetebilmektedir. Düzeltme işlemi gerçekleştirebilmek için ilgili sorunun bulunduğu satır üzerindeki sarı renkli düğmeye, silme işlemi için ise ilgili satırdaki kırmızı renkli düğmeye tıklanmalıdır. Yeni soru eklemek için soru listeleme sayfasından hızlı erişim sağlanabilmektedir. Bunu yapabilmek için listenin sağ üst köşesinde bulunan yeşil renkli düğmeye tıklanmalıdır.

Akıllı Soru Bankası		Öğr.Gör Samet Diri	
Ana Sayfa Konu İşlemleri Soru İşlemleri Sınav İşlemleri Çıkış			
Soru Adı	Konu Adı	Zorluk	Çeldirici
Crossover (eşleme) nedir?	GENETİK ALGORITMA	3	3
Quadrik denklemler kaç'ncü derecedir?	OPTİMİZASYON TEORİSİ	4	3
Quadrik denklemler kaç'ncü derecedir?	OPTİMİZASYON TEORİSİ	4	3
gradient vektör nedir?	OPTİMİZASYON TEORİSİ	3	3
hessian matris nedir?	OPTİMİZASYON TEORİSİ	3	3

Şekil 3.16. Tanımlı soruların listeleme ekranı

3.2.2.9. Soru Güncelleme

Bir önceki bölümde de bahsedildiği gibi, soru listeleme ekranında düzeltilmek istenen sorunun bulunduğu satırdaki sarı renkli düğme kullanılarak Şekil 3.17’de gösterilen soru güncelleme ekranına erişim sağlanabilmektedir. Sayfa yüklendiğinde hatalı olan sorunun gövdesi, cevap şıkları, ilişkili olduğu konu başlığı, zorluk ve ayırt edicilik değerleri ilgili alanlara doldurulmuş olacaktır. Kullanıcı; sorunun ait olduğu konu başlığını, sorunun gövdesini, cevap şıklarının içeriğini ve doğru cevap şikkını güncelleyebilmektedir. Herhangi bir sınavda sorulmamış sorular için zorluk ve ayırt edicilik değerlerinin düzenlenmesi işlemi de yapılabilmektedir. Fakat sınavda sorulmuş bir soruya ait zorluk ve ayırt edicilik değerleri ASBS tarafından güncellendiğinden, bu değerler kullanıcı tarafından değiştirilemeyecektir. Soru ile ilgili hatalı veya düzeltilmesi istenen kısımlar değiştirildikten sonra “Güncelle” düğmesine tıklanarak düzeltme işlemi tamamlanmalıdır. Kullanıcı, düzeltme işlemi tamamlandıktan sonra, ASBS tarafından otomatik olarak soru listeleme ekranına yönlendirilecektir.

Akıllı Soru Bankası
Ana Sayfa Konu İşlemleri Soru İşlemleri Sinav İşlemleri Çıkış

Konu Sezgisel Yöntemler

Soru Genetik algoritmalar için hangisi doğrudur?

A) Rulet, bir sonraki nesil için aktarılacak genlerin seçiminde kullanılır.

B) Rulet, gendeki eleman sayısını belirlemek için kullanılır.

C) İyi tanımlanmış amaç fonksiyonu ile çözüm daha kolay bulunur.

D) Çaprazlama birden fazla noktadan yapılabilir.

E) Elitizm ile güçlü gen sonraki nesillere aktarılır.

Öngörülen Zorluk (4) 1 5

Öngörülen Ayırt Edicilik (2) 1 5

Soru Kaydet

Şekil 3.17. Soru düzenleme ekranı

3.2.2.10.Soru Silme

Soru silme bağlantısına, daha önceki sınavlarda sorulmamış sorular için erişilebilmektedir. Soru silme bağlantısına tıkladığında, ilgili soruya ait silme onayı gelecektir. “Tamam” butonuna tıkladığında soru silme işlemi tamamlanacaktır. “İptal” butonuna tıkladığında ise silme işlemi gerçekleşmeyecektir. ASBS üzerinde soru silme işlemi gerçek anlamda veri tabanından verinin silinmesi değil, “soru” tablosundaki mantıksal tipteki “silindi” alanının “false” değerinden “true” değerine dönüştürülmesidir. Bu sayede, “silindi” alanının değeri “true” olan sorular listeleme ekranında görüntülenmemektedir.

3.2.2.11.Sınav Tanımlama

Sınav tanımlama işlemi için, Şekil 3.18’de bulunan sınav tanımlama ekranı kullanılmaktadır. Sınav tanımlama ekranına sayfanın üst kısmında bulunan ana menünün “Sınav İşlemleri” sekmesinin altındaki “Sınav Ekle” bağlantısına tıklanarak erişilebilmektedir.

Sınav Adı	Matematik-I
Sınav Tarihi	20.04.2018 17:01
Konular	FONKSİYONLAR BILGISAYAR PROGRAMLAMA II GENETİK ALGORITMA OPTİMİZASYON TEORİSİ TEMEL BILGISAYAR BILGISI
Sınav Soru Sayısı	10
Sınav Süresi	50
Kitapçık Sayısı	Kaç kitapçık?
Kitapçık Kodu Başlangıcı	İlk kitapçık kodu
Zorluk (3)	1 <input type="range"/> 5
Ayırt Edicilik (3)	1 <input type="range"/> 5
Frekans (0)	0 <input type="range"/> 10
<input type="button" value="Sınav Oluştur"/>	

Şekil 3.18. Sınav oluşturma ekranı

Sınav oluştururken sınavın ismi tanımlanmalıdır. Sınavın ismi basılacak olan sınav kitapçıklarının kapak sayfalarında ve iç sayfaların üst bilgi kısımlarında gösterilmektedir. Bunların yanı sıra, sınav listeleme ekranındaki künye bilgilerinde de kullanıcının oluşturduğu sınavları ayırt edebilmesi için sınav tarihi ile birlikte gösterilmektedir.

Sınav tarihi ve sınav süresi yine sınav isminde olduğu gibi basılacak olan sınav kitapçıklarının kapak sayfalarında ve iç sayfalarında üst bilgi kısımlarında gösterilmektedir. Bunların yanı sıra, sınav listeleme ekranındaki künye bilgileri de kullanıcının oluşturduğu sınavları ayırt edebilmesi için sınav adı ile birlikte gösterilmektedir.

Kitapçık sayısı, sınavda kullanılacak testin kaç farklı kitapçık ile uygulanacağını belirlemek için kullanılmaktadır. Sınav için GA ile soru seçim işlemi tamamlandıktan sonra, sorular kullanıcı tarafından belirlenmiş sayı kadar farklı kitapçık için, tek tek kendi içinde tamamen rastgele biçimde karıştırılarak veri tabanına kaydedilirler. Ardından, sorular kullanıcıya sınav detayı ekranında, her bir kitapçık için ayrı bir bağlantı ile kitapçık düzeninde gösterilir.

Kitapçık kodu başlangıcı, kitapçıkların kullanıcıya gösterilmesi sırasında kitapçıkların cevaplanacağı optik formlarda farklılıklar olabileceği göz önünde bulundurularak yine kullanıcı tarafından yapılmaktadır. Örneğin: 2 kitapçık oluşturularak uygulanan bir sınav için, kullanıcının elinde “K”, “L”, “M” ve “N” kodlarının olduğu bir optik form bulunması durumunda, uygulanacak kitapçıkların “A” ve “B” kodları ile basılması, sınavın uygulanması sırasında kargaşaya neden olabilir. Bu sebeple, sınavı oluşturan kullanıcı kitapçık sayısı olarak 2’yi seçer ve kitapçık başlangıç kodu olarak da “K” seçerse, ASBS otomatik olarak “K” ve “L” kitapçıklarını oluşturacaktır. Kitapçık başlangıç kodu seçimi ile, uygulamanın basit görünmesine rağmen sınav uygulaması sırasında sorunları en aza indirmek için etkili bir seçenek sunulmaktadır.

Konular, soru sayısı, zorluk, ayırt edicilik ve frekans alanları ise sınavda uygulanacak soruların GA ile seçilmesinde kullanılmaktadır. Konular kısmında kullanıcının ASBS üzerinde tanımladığı tüm konular listelenmektedir. Konuların sınav oluşturma işleminde seçilmesinin amacı, sınavda sorulması istenen konu başlıklarının

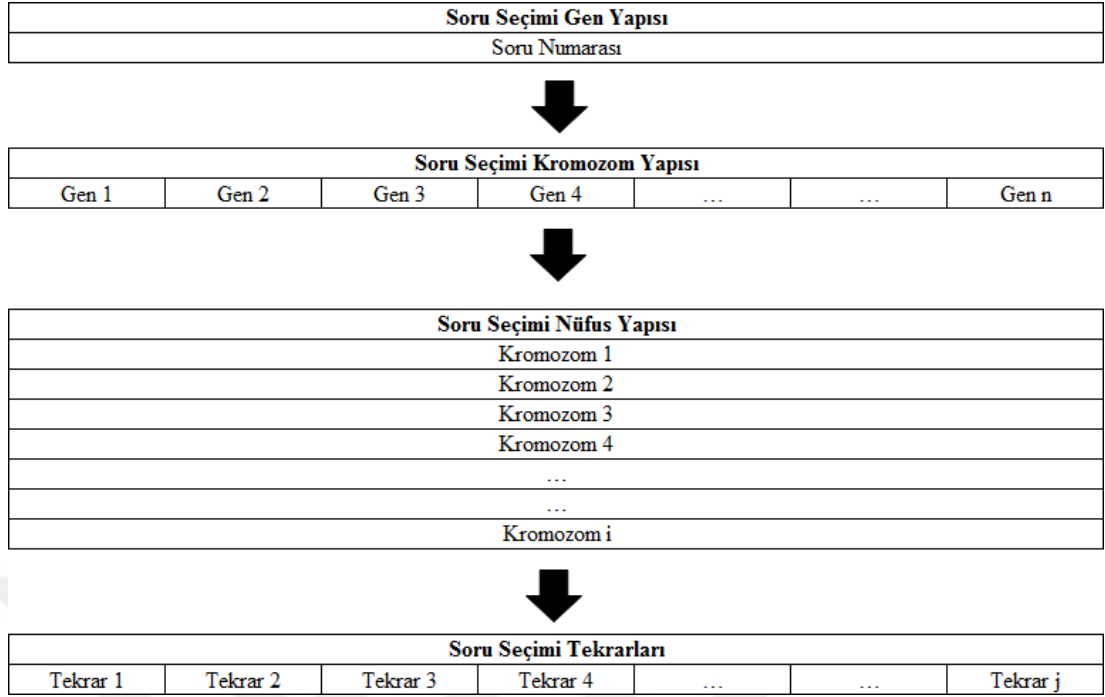
belirlenmesidir. Seçilen konulara ait soruların; numaraları, daha önce sorulmuş soruların sorulma sıklığı verileri, zorluk ve ayırt edicilik değerleri GA yazılımı için ilgili fonksiyona aktarılmaktadır.

Zorluk ve ayırt edicilik değerleri, ASBS tarafından GA kullanılarak oluşturulacak sınavın, kullanıcının istediği zorluk ve ayırt edicilikteki sorulardan seçilmesi için kullanılmaktadır. Bu değerler, GA'nın amaç fonksiyonunda çıkış koşulu olarak aranan değerlerdir. Normal şartlarda istenen durum, orta zorlukta ve ayırt edicilikte soruların olduğu bir sınav oluşturmaktır. Ancak, bazı koşullarda kullanıcılar, kendi istedikleri zorluk ve ayırt edicilik değerlerinde sınavlar oluşturabilmektedir. Örneğin: istenmedik bir durum olsa da kullanıcı, çok kolay ancak ayırt edici bir sınav oluşturmak isteyebilir.

Sınav soru sayısı, GA ile zorluk, ayırt edicilik ve sorulma sıklığı değerleri kullanılarak seçilmesi istenen soru sayısıdır. Buradaki en önemli kısıt, sınavda sorulması istenen soru sayısının seçilen konulara ait toplam soru sayısını aşmamasıdır. Konuların toplam soru sayısının sınavda sorulması istenen soru sayısından çok daha fazla olması durumunda, kullanıcının istediği değerlere daha yakın bir sınav önerisi bulunabilir. Aksi durumda, kullanıcının istediği değerlerden uzaklaşılabilir.

“Sınav Oluştur” düğmesine basıldığında, sayfanın içerisinde tanımlanmış olan HTML formu gönderilmektedir. Formun elemanları GA için geliştirilen kodu çalıştırmaktadır. Bu işlem seçilmek istenen soru sayısına, veri kümesinin büyüklüğüne, çalışan sistemin ayarlarına bağlı olarak biraz uzun sürebilir.

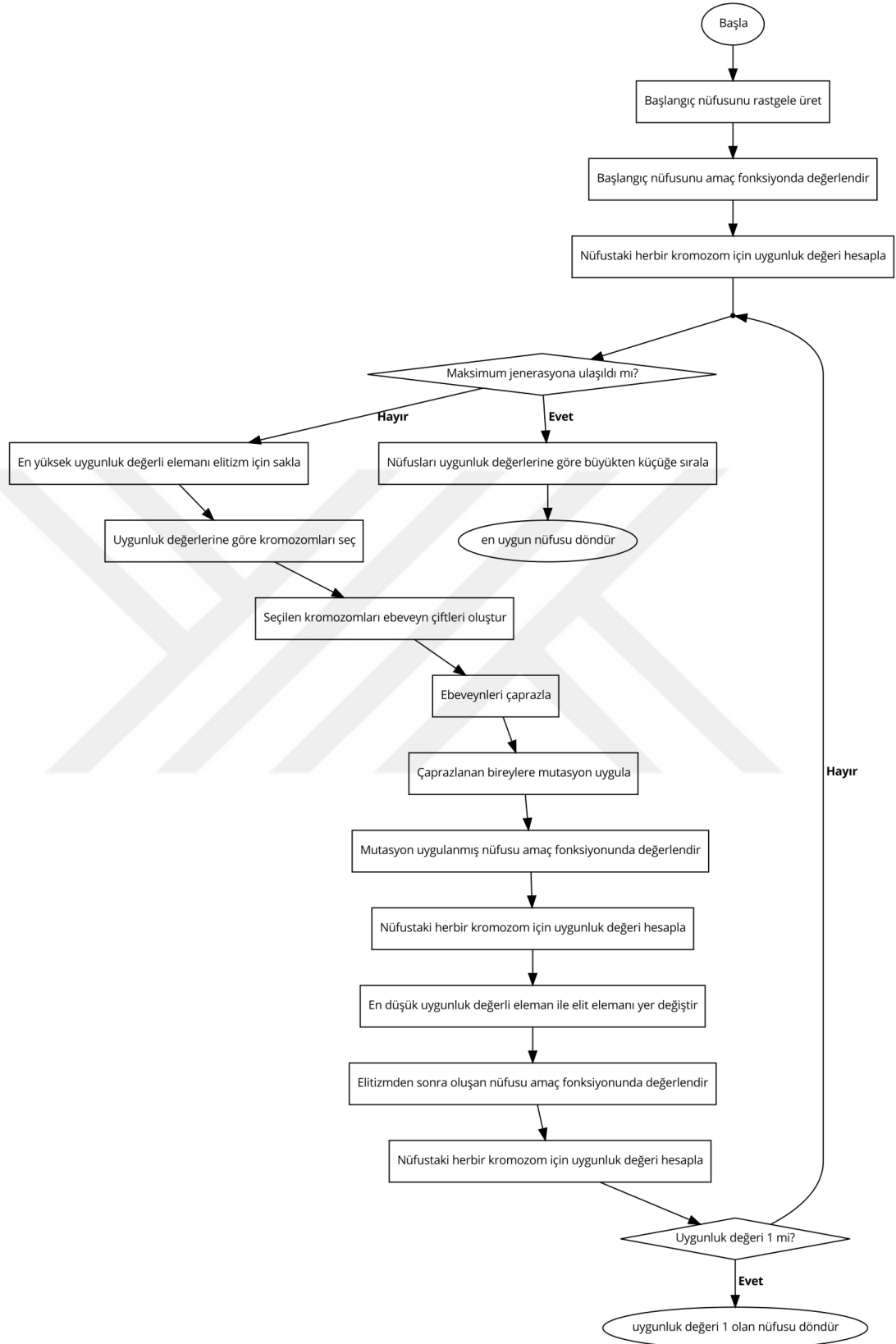
ASBS’de sınav için soru seçimi işlemine koşulan GA’nın gen, kromozom, nüfus ve tekrarlama yapıları Şekil 3.19’da gösterilmektedir ve çalışması Şekil 3.20’de gösterildiği gibi gerçekleşmektedir.



Şekil 3.19. Genetik algoritma gen, kromozom, nüfus ve tekrarlama yapıları

“Test” metodu ASBS’nin soru seçim işleminde GA’yı koşturan fonksiyondur. Soru seçimi işlemi test fonksiyonunda diğer metotları da kullanarak gerçekleştirilmektedir.

Soru seçimi için öncelikle, GA’nın “test” isimli metoduna string tipinde “veri” isimli JavaScript Object Notation (JSON) formatında,, ve numeric tipinde “soruSayisi”, “zorlukDerecesi” ve “celdiriDerecesi” isimlerinde 4 adet parametre gönderilir. “veri” isimli parametre sınav oluşturma sayfasında seçilen konulara ait soruların numaraları, sorulma sıklığı, zorluk ve ayırt edicilik derecelerini JSON biçimli string olarak almaktadır. “soruSayisi”, “zorlukDerecesi” ve “celdiriDerecesi” isimli parametreler ise yine sınav oluşturma sayfasındaki ilgili girdilerin değerlerine karşılık gelmektedir.



Şekil 3.20. Uygulamada geliştirilen genetik algoritmanın akış diyagramı

Test fonksiyonuna gönderilen JSON formatındaki veri parametresi CF içinde işlem yapılması kolay olduğundan “CFJSON2Query” metodu kullanılarak query (sorgu) tipine dönüştürülmektedir. “CFJSON2Query” metodu, JSON formatında gönderilen veriyi, oluşturulan sorgu tipindeki değişkene JSON anahtarlarına göre isimlendirilen sütunlara eklemektedir. Elde edilen sorgudan soruların numarası bir diziye aktarılmaktadır. Döngü içerisinde bazı koşullarda kullanmak üzere sorgudan dönen satır sayısı da saklanmaktadır. Seçim işlemleri başlamadan önce yapılması gereken son işlemler; mutasyon oranı, eş sayısı, nüfusta kaç kromozom olacağı ve en fazla kaç nesil için programın çalışacağını belirlemesidir. ASBS için çalışan GA'nın mutasyon oranı %0.2 olarak belirlenmiştir. Her bir nesildeki nüfus sayısı ise 50'dir. MATLAB gibi dizi ve matrislerle işlem yapmanın daha hızlı ve kolay olduğu deneysel diller kullanıldığında nüfus sayısı arttırılabilir ancak CFML, C#, JAVA gibi uygulamaya yönelik dillerde nüfus sayısını sınırlı tutmak uygulamanın hızlı çalışması açısından faydalı olacaktır.

GA'nın çalışması için başlangıç nüfusuna ihtiyaç vardır. Başlangıç nüfusunu oluşturmak için “initPopulation” isimli metot kullanılmaktadır. “initPopulation”a nüfus sayısı ve sınav için seçilmesi gereken soru sayısı, rastgele seçim yapmak için soru numaralarının olduğu bir dizi parametre olarak gönderilmektedir. Bu metot iç içe iki döngü içermektedir. Dışarıdaki döngü nüfus sayısı kez dönmektedir ve her tekrarda soru numaralarının olduğu diziyi “shuffle” metodu kullanarak karıştırmaktadır. İçerideki döngü ise karıştırılan dizinin ilk soru sayısı kadar elemanını başlangıç nüfusunu tutan diziye eklemektedir. Bu işlemler tamamlandığı takdirde geriye başlangıç nüfusu olan soru numaralarının olduğu nüfus sayısı kadar satır sayısı, soru sayısı kadar sütun sayısı olan iki boyutlu bir dizi döndürülmektedir.

Başlangıç nüfusunun üretilmesinin ardından, ilgili nüfus amaç fonksiyonundan geçirilerek maliyet değerleri hesaplanmaktadır. Amaç fonksiyonu olarak kullanılan metodun ismi “objectiveFunction”dur. “objectiveFunction” metodu; ilk nüfusun elemanlarını dizi, orijinal veri kümesini sorgu, istenen zorluk ve ayırt edicilik değerlerini ise sayısal tipte parametreler olarak almaktadır. “objectiveFunction” metodunda öncelikle, nüfus sayısı ve kromozom uzunluğu döngü koşullarında kullanılmak üzere değişkenlerde saklanmaktadır. Ardından zorluk, frekans ve ayırt

edicilik değerlerinin ve bu değerlerin toplamalarının tutulacağı sıfır değerlerinden oluşan vektörler oluşturulmaktadır.

Sıfırlardan oluşan vektörler “sifirlarDizisi” isimli fonksiyona satır sayısı ve sütun sayısı parametreleri gönderilerek oluşturulmaktadır. Sütun sayısı olarak 1 veya daha büyük bir sayı gönderilebilmektedir.

Amaç fonksiyonunun ilk değerleri oluşturulduktan sonra, nüfus sayısı kadar tekrarlayan dış döngünün içerisinde kromozom sayısı kadar tekrarlayan iç döngü tanımlanmıştır. İçerideki döngüde, parametre olarak gönderilen nüfustaki kromozomların ilgili genlerine ait zorluk, ayırt edicilik ve frekans değerleri veri kümesinden alınır ve Denklem (3.3)’e göre istenilen zorluk, ayırt edicilik ve frekans değerleri farklarının mutlak değerleri hesaplanmaktadır.

$$f(x)=\text{Minimize} \sum_{i=1}^n (|z_i-z_x|+|a_i-a_x|+|f_i-f_x|) \quad (3.3)$$

Burada, z_i , genin zorluk değerini,

z_x istenen zorluk değerini,

a_i , genin ayırt edicilik değerini,

a_x istenen ayırt edicilik değerini,

f_i , genin daha önce sorulma sayısı değerini,

f_x istenen daha önce sorulma sayısı değerini ifade etmektedir.

Örneğin: 10 kromozomu ve her bir kromozomunda 5 geni olan bir nüfusun, zorluk değeri olarak 3, ayırt edicilik değeri olarak 3 ve frekans değeri olarak 0’ın parametre olarak gönderildiğini varsayalım. Bu durumda, istenilen zorluk ve ayırt edicilik değerleri 3 olduğu için aslında GA’nın aradığı değer 5 gen için $3*5=15$ toplam zorluk ve ayırt edicilik değerleridir. Frekans 0 olduğu için toplam frekans yine 0 olacaktır. Amaç fonksiyonunda dış döngü 10 kez tekrarlayacaktır. İç döngü ise 5 kez tekrarlayacaktır. İç döngüde nüfusun 1 konumundaki kromozomun genlerinin toplam zorluk değerinin 19, toplam ayırt edicilik değerinin 12, toplam sorulma sıklığının 2

olduğunu varsayarsak; ilgili kromozoma ait zorluk maliyeti 4, ayırt edicilik maliyeti 3 ve sorulma sıklığı maliyeti 2'dir. Bu şekilde ilgili kromozoma ait toplam maliyet 9'dur. Tüm kromozomlar için aynı hesaplama işlemi yapılmaktadır ve veriler ilgili vektörün uygun satırına yazılmaktadır. Tüm kromozomlar için hesaplama işleminin tamamlanmasının ardından maliyet vektörü dönüş değeri olarak kullanılmaktadır.

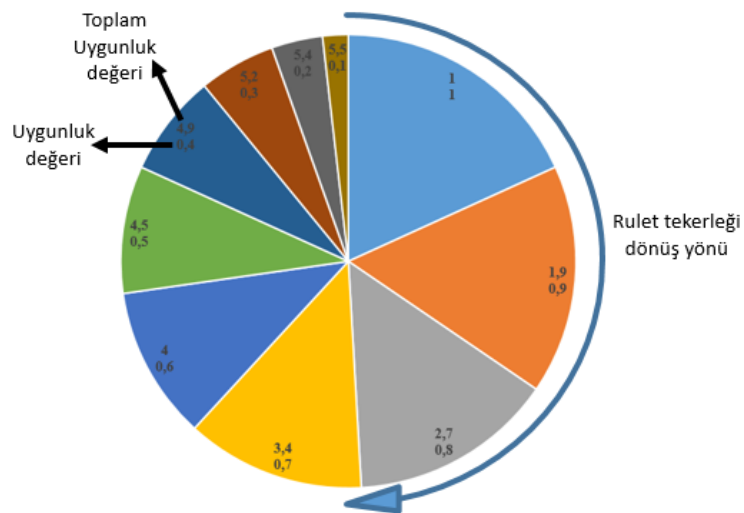
Amaç fonksiyonunda gerekli hesaplamaların tamamlanmasının ardından, amaç fonksiyonundan elde ettiğimiz maliyet vektörü ve nüfustaki kromozom sayısı "fitnessFunction" metoduna parametre olarak gönderilmektedir. "fitnessFunction" fonksiyonu öncelikle parametre olarak aldığı maliyet vektörünü sıralayıp orijinal dizideki yerlerini de bulan "diziSiraBul" metoduna gönderilmektedir. "diziSiraBul" metodundan geriye döndürülen sıralanmış maliyetlerin uygunluk değerleri hesaplanmaktadır. "fitnessFunction"un geri dönüş değeri hesaplanmış olan uygunluk vektörüdür.

Başlangıç nüfusu ile ilgili ön işlemler bittikten sonra, maksimum nesil sayısı kadar tekrarlayan bir döngü kurulmaktadır. Bu döngünün amacı, uygunluk değeri 1 olan nüfus bulunamazsa programın sonsuz döngüye girmesini engellemektir. Bu sayede sistem kaynaklarının kullanımı ve zaman aşımı ihtimali sınırlanmaktadır.

Döngünün içerisinde ilk olarak hem başlangıç nüfusunda hem de her tekrarda yeni nesiller üretilecek olan nüfuslarda kullanılması üzere elitizm işlemi yapılmaktadır. Elitizm işlemi için "maxElemanBul" metodu kullanılmaktadır. "maxElemanBul" fonksiyonu, nüfusun içerisinde uygunluk değeri en yüksek olan kromozomu ve uygunluk vektöründeki yerini iki elemanlı bir dizi olarak döndürmektedir. Bu metottan dönen eleman elitizm işlemi için saklanmaktadır. Yeni nesil üretildiğinde en düşük uygunluk değerine sahip olan kromozom ile elit kromozom yer değiştirilecektir. Bu sayede gelecek nesillerin kötüleşmesi önlenmeye çalışılmaktadır. Buna ek olarak uygulamanın performansı arttırılmaya çalışılmaktadır. Elitizm için kullanılan kromozomlar, seçkin kromozomlardır. Rulet tekerleği uygulandığında seçilmeme ihtimalleri göz önüne alındığından elitizm işlemi gerçekleştirilmektedir. Nüfustaki en uygun kromozomun gelecek nesil için seçilememesi performansı azaltabilir ve sonucun bulunamamasına neden olabilir.

Geliştirilen GA’da seçim için rulet tekerleği kullanılmaktadır. Rulet tekerleği seçim işlemi için “selectFunction” metodu kullanılmaktadır. “selectFunction” metodu uygunluk değerlerinin bulunduğu vektörü ve nüfus sayısını parametre olarak almaktadır. Metot ilk olarak uygunluk değerlerinin toplamını hesaplamaktadır. Ardından rulet için uygunluk vektörünün eleman sayısı kadar rastgele değer üretilmektedir. Rastgele sayı üretme işlemi “randomSayı” metodu tarafından üretilen 0-1 aralığındaki sayı ile uygunluk değerleri toplamının çarpımına epsilon kadar küçük bir sayı eklenmesi ile yapılmaktadır. Elde edilen sayı 0’dan büyük ve en fazla toplam uygunluk değerinden biraz küçük olacak şekilde düzenlenecektir. Bu şekilde rastlantısallık ilkesinin tamamen sağlanması amaçlanmaktadır. Rulet vektörü ile yapılmak istenen, çarkın nerede duracağını belirlemesidir.

Rulet için vektörün oluşturulmasının ardından, nüfus sayısı kadar tekrarlayan bir döngü içerisinde, ruletin döngü değişkeni değerindeki elemanın uygunluk değerleri toplamı hesaplanırken ruletin hangi elemana kadar olduğu bulunmaktadır. Bu sayede, uygunluk değerleri toplanarak dönen bir çarktaki gibi rastgele bir yerde durmaktadır. Örneğin: Şekil 3.21’de gösterilen rulet tekerleği için ruletin ilk elemanın 4.2 olduğunu varsayalım. Bu durumda tekerlek ilk elemandan itibaren toplama işlemi yapmaya başlayacaktır; 5. toplama işlemi sonunda 4.2’nin 4’ten büyük olduğunu tespit edecek ve uygunluk değeri 0.6 olan 5. elemanı bir sonraki nesil için seçecektir.



Şekil 3.21. Rulet tekerleği yöntemi

Bir sonraki nesil için seçilimi gerçekleştirilen kromozomların eşleştirilmesi ve yeni kromozomların oluşturulması gerekmektedir. Bunun için “parentFunction” metodu kullanılmaktadır. “parentFunction” metodu orijinal nüfus, seçim sonucu oluşturulan yeni nüfus ve nüfustaki kromozom sayısı parametre olarak gönderilmektedir. Burada kromozom sayısının yarısı kadar kez tekrarlayan bir döngü kurulmaktadır. Her bir tekrarda orijinal nüfustan, seçilmiş olan elemanlar baştan ve ortadan getirilerek eşleştirmeler yapılmaktadır. Örneğin: 10 elemanı olan bir nüfus için 1 ve 6, 2 ve 7, 3 ve 8, 4 ve 9, 5 ve 10 numaralı kromozomlar çaprazlama işlemi için eşleştirilmektedir. Kromozomlar zaten rastgele bir şekilde seçildiği için tekrardan eşleştirmeye girmeleri için rastgele bir sayı üretilip eşiği aşmış aşmadığı kontrol edilmemektedir. Bunun yapılmamasındaki temel nokta, yukarıda da bahsedildiği gibi, önceki aşamalarda tamamen rastgele seçim yapıldığı için ek bir rastgele sayı tutma, eşik kontrolü yapma gibi performansa etki edecek ilave işlemlerden kaçınılmasıdır.

Eşleştirme işleminin tamamlanmasının ardından ebeveynler, nüfustaki kromozom sayısı, soru sayısı ve toplam veri dizisinin uzunluğu çaprazlama işlemi için “crossOverFunction” metoduna parametre olarak gönderilmektedir. “crossOverFunction” metodunda çaprazlama noktası “randomRangeSayı” fonksiyonundan (1, soru sayısı-1) aralığında dönen değer ile bulunmaktadır. Çaprazlama noktasının belirlenmesinden sonra ebeveynler bir döngü içerisinde birinci ebeveynin 1-nokta arasındaki parçası ile ikinci ebeveynin nokta-son gen parçası, ikinci ebeveynin 1-nokta arasındaki parçası ile birinci ebeveynin nokta-son gen parçası birleştirilerek çaprazlama işlemi yapılmaktadır. Tüm eşlerin çaprazlanması tamamlandıktan sonra bir kromozom içerisinde aynı genlerin olması ihtimaline karşı kromozomlar kontrol edilmektedir. Aynı çıkan genler mutasyona benzer biçimde veri kümesi içerisinde rastgele seçilen genler ile değiştirilir. Düzenleme işleminin tamamlanmasının ardından, geri dönüş değeri olarak elde edilen düzeltilmiş nüfus kullanılmaktadır.

Çaprazlama işlemi sonucunda elde edilen yeni nüfus, orijinal verinin boyutu ve mutasyon oranı “mutationFunction” metoduna parametre olarak gönderilmektedir. Bu fonksiyon tüm kromozomların her bir geni için “randomSayı” metodu tarafından üretilen rastgele değerin mutasyon oranından küçük olup olmadığını kontrol

etmektedir. Eđer rastgele retilen deęer mutasyon deęerinden kk ise orijinal veriden rastgele bir soru seilir ve mevcut kromozomun ierisinde yoksa rastgele seilen soru ilgili gen ile deęiştirilir. Bu sayede mutasyon iřlemi gerekleřtirilir. Mutasyon iřleminin uygulanmasının en nemli faydası, yerel en iyi olan ve algoritmanın daha fazla ilerlemesini engelleyen durumun ortadan kaldırılmasını saęlamaktır. “mutationFunction” metodunun geri dnř deęeri mutasyona uęramıř nfustur.

Mutasyona uęramıř nfus, ilk nfusta olduęu gibi yine amaca uygunluęunun test edilmesi iin “objectiveFunction” metoduna gnderilmektedir.

Ama fonksiyonun rettięi maliyetler uygunluklarının hesaplanması amacıyla “fitnessFunction” metoduna gnderilmektedir.

Uygunluk deęerleri hesaplanan nfusun en dřk uygunluk deęerine sahip kromozomu “minElemanBul” metodu ile bulunur. “minElemanBul” fonksiyonu, nfusun ierisinde uygunluk deęeri en dřk olan kromozomu ve uygunluk vektrndeki yerini iki elemanlı bir dizi olarak dndrmektedir. Bu metottan dnen kromozom, elitizm iřlemi iin saklanan en sekin kromozom ile yer deęiştirilir ve elitizm iřlemi tamamlanmıř olur. Daha nce de bahsedildięi gibi, bu sayede en iyi kromozomun kaybolması nlenmiř ve gelecek nesillerin daha hızlı iyileřmesi saęlanmaya alıřılmıř olur.

Mutasyon iřlemine tabi tutulmuř ve elit kromozomun de katılması ile oluřturulan nfus son kez ama ve uygunluk fonksiyonlarında iřlendikten sonra elde edilen uygunluk deęerinin ıkıř kořulunu saęlaması durumu kontrol edilmektedir. ıkıř kořulu saęlanıyorsa, dng kırılır ve en uygun kromozomdaki soru numaraları ASBS’ye dndrlr. ıkıř kořulu saęlanmıyorsa, dng maksimum nesile ulařana kadar tekrarlamaya devam eder ve son nesile ulařıldıęında, en yksek uygunluk deęeri olan kromozomdaki soru numaraları ASBS’ye dndrlr.

3.2.2.12. Sınav Listeleme

Kullanıcılar, ASBS zerinde kendi hesaplarında tanımladıęı tm sınavları Őekil 3.22’de gsterildięi gibi tek bir ekranda listeleyebilmektedir. Sınav listeleme ekranına

sayfanın üst kısmında bulunan ana menünün “Sınav İşlemleri” sekmesinin altındaki “Sınav Listele” bağlantısına tıklanarak erişilebilmektedir. Bu bağlantı kullanarak erişilen sayfada, tanımlanmış olan her bir sınavın; adı, tarihi, soru seçiminde kullanılan konuları, soru sayısı, kitapçık sayısı, kitapçık başlangıç kodu, istenen zorluk ve ayırt edicilik değerleri ile sorulma sıklıkları görüntülenebilmektedir. Bunlara ek olarak her bir sınavın kitapçıklarını görüntüleme, sınavı değerlendirme, sonuçları listeleyebilme, sınavda çıkmış soruların analizini görüntüleyebilme, oluşturulmuş sınavın sorularını kopyalayarak yeni sınav oluşturma, oluşturulmuş ancak uygulanmadığı için değerlendirme işlemi gerçekleştirilmemiş sınavı silme ve yeni bir sınav ekleme sayfasına erişim bağlantıları bulunmaktadır. Kullanıcılar kendi sınavlarına ait işlemleri bu sayfa aracılığıyla yönetebilmektedir.

Sınav için üretilen kitapçıkların listelendiği sayfaya erişmek için ilgili sorunun bulunduğu satır üzerindeki açık mavi renkli düğmeye tıklanmalıdır. Uygulanan sınavın değerlendirmesini yapmak için, ilgili sorunun bulunduğu satır üzerindeki yeşil düğmeye tıklanmalıdır. Değerlendirmesi tamamlanmış olan sınavın sonuçlarını listelemek için ilgili sorunun bulunduğu satır üzerindeki koyu mavi renkli düğmeye tıklanmalıdır. Değerlendirmesi tamamlanmış olan sınavın soru analizlerini görüntüleyebilmek için, ilgili sorunun bulunduğu satır üzerindeki beyaz renkli düğmeye tıklanmalıdır. Mevcut sınavın sorularını kullanarak yeni bir sınav oluşturmak için ilgili sorunun bulunduğu satır üzerindeki sarı renkli düğmeye tıklanmalıdır. Uygulanmamış sınavı silmek için ise ilgili satırdaki kırmızı renkli düğmeye tıklanmalıdır. Yeni sınav eklemek için sınav listeleme sayfasından hızlı erişim sağlanabilmektedir. Bunun yapabilmek için listenin sağ üst köşesinde bulunan uzun, yeşil renkli düğmeye tıklanmalıdır.

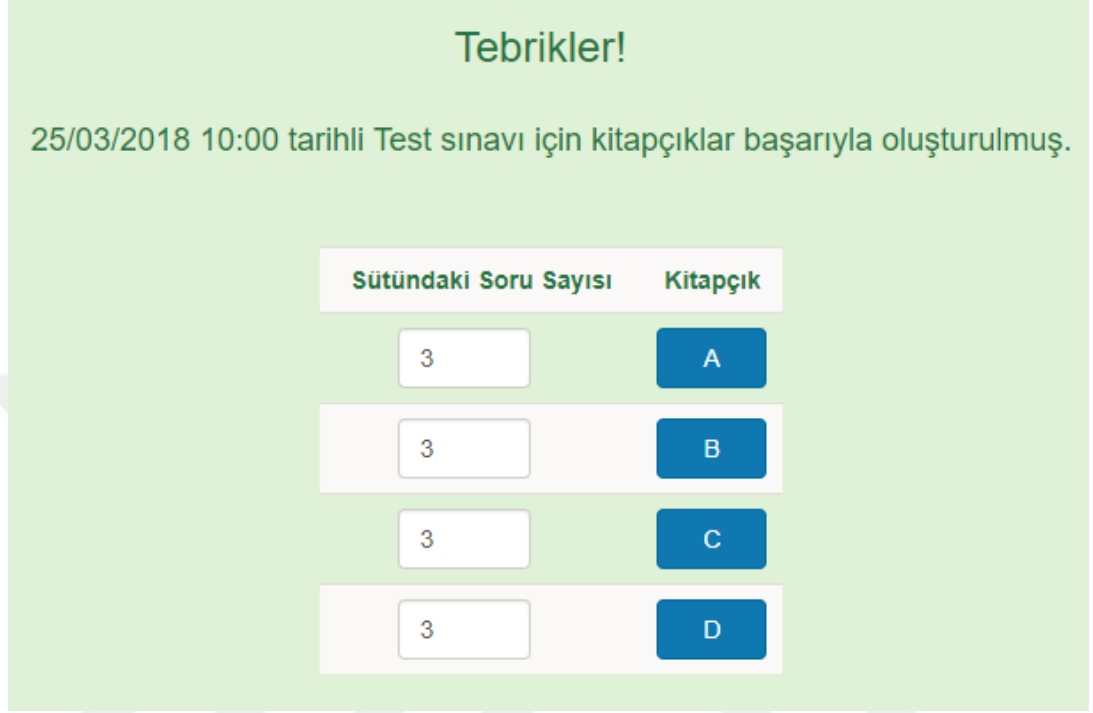
Sınav Adı	Tarihi	Konular	Soru Sayısı	Sınav Süresi	Kitapçık Sayısı	Kitapçık Kodu	Zorluk	Ayirt Edicilik	Frekans
Test	25/03/2018 10:00	Sezgisel Yöntemler, Veri Madenciliği	6	70	4	A	3	2	1

Şekil 3.22. Tanımlı sınav bilgileri listeleme ekranı

3.2.2.13. Kitapçık Listeleme

Sınav detaylarının listelendiği sayfadaki mavi renkli bağlantıya tıklanarak erişilen kitapçık listeleme sayfası Şekil 3.23'te gösterilmektedir. Burada ASBS tarafından seçilen sorulardan oluşan, kitapçık başlangıç kodundan başlayarak kitapçık sayısı kadar sınav kitapçığı alfabetik sırada listelenmektedir. Sütundaki soru sayısı parametresi kitapçıkta her bir sütunda olması gereken soru sayısını göstermektedir. Uzun sorulardan oluşan bir sınav düzenleniyorsa sütundaki soru sayısı azaltılabilir.

Aynı şekilde daha kısa sorulardan oluşan bir sınav için sütundaki soru sayısı artırılabilir. Görüntülemek istenen kitapçık için öncelikle sütundaki soru sayısı değeri girildikten sonra ilgili kitapçık kodu tıklanarak kitapçığın içeriğine erişilebilmektedir.



Şekil 3.23. Sınav kitapçıklarının listelenmesi

3.2.2.14. Kitapçık Görüntüleme

İlgili kitapçık koduna tıklanarak kitapçık detayları görüntülenebilmektedir. Şekil 3.24'te örnek bir sınav kitapçığının kapak sayfası gösterilmektedir. Kitapçığın üst bilgi kısmında sınavın ismi ve sınav tarihi gösterilmektedir. Alt bilgi kısmında ise sınavı düzenleyen kullanıcının kimlik bilgileri ve ASBS'nin ismi yazmaktadır. Alt ve üst bilgiler tüm sayfalarda gösterilmektedir. Kapak sayfasında öğrencinin numarasını, adını ve soyadını yazabileceği bir kısım bulunmaktadır. Bunların hemen yanında kitapçığın kodu gösterilmektedir. Kitapçık kodu kapak sayfasının yanı sıra sayfa numaralarının yanlarında, sayfanın alt-orta kısımlarında da gösterilmektedir. Bunların yanı sıra, yine kapak sayfasında sınav kuralları, konulara göre sınavda uygulanacak olan soru sayıları gösterilmektedir. Sınav kuralları sabit bir metin olarak tüm sınavlar için ortaktır. İç kısımlarda kitapçık üretilirken belirlenen satır sayısı kadar ve iki sütun olacak şekilde sınav soruları gösterilmektedir.

Test
25/03/2018 10:00

Öğrenci No:
Ad:
Soyad:
İmza:

A

1. Bu sınav için verilen toplam cevaplama süresi **70 dakikadır**.
2. Her sorunun sadece bir doğru cevabı vardır. Bir soru için birden çok cevap işaretlenirse o soru yanlış cevaplanmış sayılacaktır.
3. Cevap kağıdınızda doldurmanız gereken alanlar bulunmaktadır. Lütfen bu alanları doldurunuz. Cevap kağıdınızı başkaları tarafından görülmeyecek şekilde tutmanız gerekmektedir. Cevap kağıdına yazılacak her türlü yazıda ve yapılacak bütün işaretlemelerde kurşun kalem kullanılacaktır. Sınav süresi bittiğinde cevapların cevap kağıdına düzgün bir şekilde işaretlendiğinden emin olunuz. Soru kitapçığına işaretlenen cevaplar geçersiz sayılacaktır.
4. Soru kitapçığının kontrol edilmesi, doğru soru kitapçık türünün yazılması ve cevap kağıdına kodlanması sizin sorumluluğunuzdadır.
5. **Sınava cep telefonu, tablet, bilgisayar v.b. cihazlar ile girilmesi kesinlikle yasaktır.**
6. Sınav salonundan her hangi bir sebep ile ayrılan öğrenci tekrar sınava alınmayacaktır.
7. Sınav sırasında kopya çeken, çekmeye teşebbüs eden, kopya veren, kopya çekilmesine yardım edenlerin kağıdı gözetmenler tarafından alınarak yönetmeliğin kopya çekme ile ilgili maddesi uygulanacaktır.

Soru Sayıları
Sezgisel Yöntemler : 1

Şekil 3.24. Kitapçık kapağı örneği

3.2.2.15. Sınav Değerlendirme

Sınav detaylarının listelendiği sayfadaki yeşil renkli bağlantıya tıklanarak erişilen sınav değerlendirme sayfası Şekil 3.25'te gösterilmektedir. Burada ilk satırda, öğrencinin sınavda cevapladığı kitapçığın kodu, altında öğrencinin numarası, sonraki satırlarda ise “;” ile ayrılmış şekilde soru numaraları ve verilen cevaplar bulunan “.CSV” uzantılı bir dosya sisteme yüklenmelidir. Sınav değerlendirmesi yüklenecek dosya ile yapılmaktadır. ASBS “.CSV” uzantılı dosyayı işleyerek öğrencinin verdiği cevapları ve doğru-yanlış-boş olma durumlarını veri tabanına kaydetmektedir. Böylece, sınav sonuçlarına daha sonra erişim imkanı sağlanmasının yanı sıra soruların analizlerinin yapılabilmesine de olanak tanımaktadır.

Değerlendirme işlemi çeşitli nedenlerle yarım kalabilir ya da kullanıcının yüklediği dosyada eksiklik olabilir. Bu nedenle sınava ait değerlendirme daha önce yapılmışsa öncelikle sınavın değerlendirmesinin temizlenmesi ve tekrardan değerlendirme işlemi yapılması gerekmektedir. Sınav değerlendirme sayfasında “Bu sınavın değerlendirme işlemi daha önce yapılmıştır. Tekrar değerlendirme işlemini yapabilmek için öncelikle var olan değerlendirmeyi temizlemeniz gerekmektedir.” uyarısı gösterilmektedir. Kullanıcı “Değerlendirmeyi Sil” düğmesine tıklayarak sınava ait tüm değerlendirme girdilerini silebilmektedir.

Sınav değerlendirmesi tamamlandıktan sonra, ASBS soruların zorluk ve ayırt edicilik değerlerini Denklem (3.1) ve Denklem (3.2)'ye göre otomatik olarak güncellemektedir.

Belge Formatı	UTF-8
Belge *	<input type="button" value="Dosya Seç"/> Dosya seçilmedi
Format	
Açıklama:	
	Dosya uzantısı csv olmalı,alan araları noktalı virgül (;) ile ayrılmalı ve kaydedilirken karakter desteği olarak UTF-8 seçilmelidir. Aktarım işlemi dosyanın 2 satırından itibaren başlar bu yüzden birinci satırda alan isimleri mutlaka olmalıdır.
	Bu belgede olması gereken alan sayısı : 2 İlk Satırda Kitapçık Kodu İkinci satırda öğrenci numarası bulunması gerekmektedir Sonraki satırlarda öğrencinin verdiği cevapların yer alması gerekmektedir katalogId;7-A ogrenciNo;5426 1;A
Alanlar sırasıyla;	
	1- Soru Sırası 2- Verilen Cevap
	<input type="button" value="Sınavı Değerlendir"/>

Şekil 3.25. Sınav değerlendirme ekranı

3.2.2.16.Sınav Sonuç Listeleme

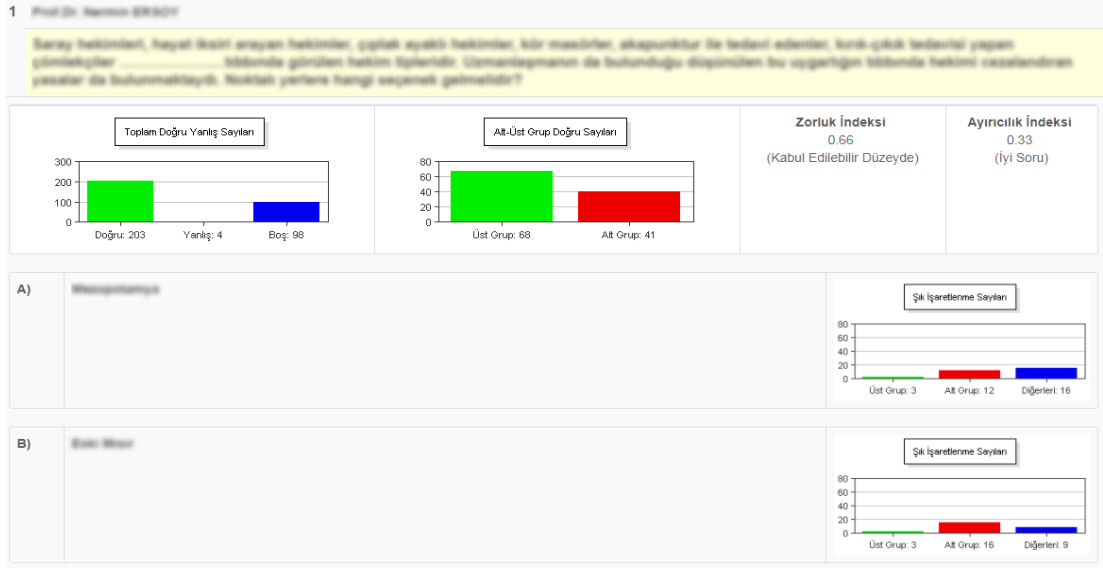
Sınav detaylarının listelendiği sayfadaki koyu mavi renkli bağlantıya tıklanarak erişilen sınav sonuçları listeleme sayfası Şekil 3.26’da gösterilmektedir. Sınav sonuç ekranında öğrencilerin numaraları, doğru, yanlış, boş cevap sayıları ve doğru cevaplama oranları listelenmektedir. Eğer kullanıcı isterse sayfanın sağ üst köşesinde bulunan indirme bağlantısına tıklayarak sonuçları “.XLS” uzantılı bir dosya olarak bilgisayarına indirebilir ve ardından işleyebilir.

25/03/2018 10:00 tarihli Test Sınavı Değerlendirmesi				
Öğrenci No	Doğru	Yanlış	Boş	Başarı
1234	4	2	0	% 66.67
1235	5	0	1	% 83.33
1236	6	0	0	%100.00
1237	1	4	1	% 16.67
1238	2	2	2	% 33.33

Şekil 3.26. Sınav sonuç listesi

3.2.2.17.Sınav Sorularının Analizi

Sınav detaylarının listelendiği sayfadaki beyaz renkli bağlantıya tıklanarak erişilen sınav sorularının analizini görüntüleme sayfası Şekil 3.27’de gösterilmektedir. Sınav analizi ekranında; sorulara alt ve üst gruptaki öğrenciler tarafından verilen doğru cevap sayıları, sorunun zorluk ve ayırt edicilik indeksleri ile her bir çeldiriciye tüm öğrencilerin verdikleri doğru, yanlış ve boş cevap sayıları ve bu sayılara ait grafikler gösterilmektedir. Analizler kullanıcıya sorusunun geçerliliği ve güvenilirliği hakkında fikir vermektedir. Kullanıcı isterse sorularının analiz sonuçlarına göre sorularında düzeltme veya güncelleme yapabilir.



Şekil 3.27. Sınav sorusu analizi

3.2.2.18. Sınav Kopyalama

Sınav detaylarının listelendiği sayfadaki sarı renkli bağlantıya tıklanarak erişilen sınav kopyalama sayfası Şekil 3.28’de gösterilmektedir. Sınav kopyalama ekranında, sınav oluşturma ekranından farklı olarak konu, soru sayısı, zorluk, ayırt edicilik ve sorulma sıklığı değerleri değiştirilemez biçimde gelmektedir. Kullanıcı, sınav adını, tarihini, sınav süresini, kitapçık sayısı ve başlangıç kodunu değiştirebilmektedir. Sınavın yeni bir kopyasını oluşturma, kullanıcının daha önceden uygulanmış olan bir sınavın tekrar uygulanarak öğrencilerin gelişimsel farklılıklarını gözlemesi için kullanılabilir. Benzer biçimde, yeniden sınav oluşturmada aynı sınavı farklı bir öğrenci kitlesine uygulayarak iki grubun birbirinden farkını incelemek için de kullanılabilir. Bunların dışında, kullanıcılar farklı çözümler geliştirip, kendi amaçları doğrultusunda da kullanabilirler.

Sınav Adı	<input type="text" value="Sezgisel Yöntemler"/>
Sınav Tarihi	<input type="text" value="26.04.2018 10:00"/>
Konular	<div style="border: 1px solid #ccc; padding: 2px;"><div style="background-color: #f0f0f0; padding: 2px;">Sezgisel Yöntemler</div><div style="padding: 2px;">Veri Madenciliği</div></div>
Sınav Soru Sayısı	<input type="text" value="10"/>
Sınav Süresi	<input type="text" value="50"/>
Kitapçık Sayısı	<input type="text" value="4"/>
Kitapçık Kodu Başlangıcı	<input type="text" value="K"/>
Zorluk (3)	1 <input type="range" value="1"/> 5
Ayırt Edicilik (3)	1 <input type="range" value="1"/> 5
Frekans (0)	0 <input type="range" value="0"/> 10

[Sınav Oluştur](#)

Şekil 3.28. Sınav kopyalama ekranı

3.2.2.19.Sınav Silme

Sınav detaylarının listelendiği sayfadaki kırmızı renkli bağlantıya tıklanarak sınav silme işlemi gerçekleştirilebilmektedir. Sınav silme bağlantısına daha önce değerlendirme işlemi yapılmamış sınavlar için erişilebilmektedir. Sınav silme bağlantısına tıklandığında, ilgili sınava ait silme onayı gelecektir. “Tamam” butonuna tıklandığında sınav silme işlemi tamamlanacaktır. “İptal” butonuna tıklandığında ise silme işlemi gerçekleşmeyecektir. ASBS üzerinde sınav silme işlemi gerçek anlamda veri tabanından verinin silinmesi değil, “sınav” tablosundaki mantıksal tipteki “silindi” alanının “false” değerinden “true” değerine dönüştürülmesidir. Bu sayede “silindi” alanının değeri “true” olan sınavlar listeleme ekranında görüntülenmemektedir.

4. SONUÇLAR VE ÖNERİLER

Hedefli öğrenme, günümüzün özel hedeflere dönük ihtiyaçların karşılanması doğrultusundaki eğilimiyle örtüşmektedir. Her şeyi bilen bireyler eğitmek üzere oluşturulmuş eğitim programları, bireylerin pek çok farklı niteliğini göz ardı etmektedir. Bunun yanında, çeşitli nedenlere bağlı olarak beliren ve özel bilgilere sahip bireyler gerektiren alanlarda gerçekleştirilecek eğitimin, geleneksel genel-maksat eğitimden farklı özellikleri yapısında barındırması gereği de aşikârdır.

İsteğe göre uyarlanmış eğitim yöntem ve içeriklerinin sunulduğu bireylerin başarısının değerlendirilmesi, uygulanagelmış geleneksel ölçme ve değerlendirme yöntemleriyle ele alınabilir. Ancak bu şekilde gerçekleştirilecek bir değerlendirme süreci, genel başarımda istenen sonuçların elde edilmesinde yeterli olmayacaktır. Bu durumda, rastlantısal yöntemlerle keyfi seçilmiş soruların oluşturduğu soru bankaları ile oluşturulacak sınavlarla yapılacak ölçme ve değerlendirme işlemleri, oluşturulan sınavın rastgele olmasından ötürü, gerçek manada hedeflenen sonuçların elde edilip edilmediğinin tespitini imkansız kılacaktır.

Sınav oluşturma süreçlerinin etkin yönetilebilmesi ve hazırlanan sınavların hedeflenen düzeyde olabilmesi için akıllı sistemlere ihtiyaç vardır. Geleneksel yöntemler ile uzun zaman alacak işlemler, bilgisayar sistemleri ve yapay zeka yöntemleri yardımıyla hem daha kısa sürecek, hem de değerlendirme sonuçlarına göre hazırlanacak raporlar ve sistemin kendi içerisinde zorluk, ayırt edicilik v.b. indeksleri güncellemesiyle, yeni oluşturulacak sınavlar hedeflenen bilgi düzeylerinin daha etkili bir biçimde ölçülmesine olanak sağlayacaktır.

Tez çalışması ile gerçekleştirilmiş olan Akıllı Soru Bankası Sistemi kullanılarak geleneksel yöntemler ile sürdürülen sınav süreçleri daha etkili yönetilmektedir. Konu başlıklarının kullanıcı tarafından dilenen biçimde genelleştirilerek veya özelleştirilerek tanımlanabilmesi ve soruların ilgili konulara eklenmesi, hedeflenen

bilgi düzeylerinin etkin olarak ölçülmesini sağlaması açısında ASBS'nin güçlü yönlerinden birisidir.

Sınav sonrasında elde edilen raporlar soruların ve sınavların geliştirilmesi için geri bildirimler sağlamaktadır. Bu sayede eğitimin özel ve genel hedeflerine ulaşma düzeyine yaklaşmayı kolaylaştırmaktadır.



KAYNAKLAR

Adıgüzel O. C., Özüdođru F., Üniversitelerde Ortak Zorunlu Yabancı Dil I Dersine Yönelik Bir Akademik Başarı Testinin Geliştirilmesi, *Trakya Üniversitesi Eğitim Fakültesi Dergisi*, 2013, 3(2), 1-11.

Akın O., Web Tabanlı Sınav Sistemi, Yüksek Lisans Tezi, Sakarya Üniversitesi, Fen Bilimleri Enstitüsü, Sakarya, 2007, 212180.

Akıncı S., Keman Eğitimine, "Öğrenmenin Geliştirilmesini Sağlayan Koşullar" Açısından Bakış, *M. Ü. Atatürk Eğitim Fakültesi Eğitim Bilimleri Dergisi*, 1998, 7(1), 1-10.

Arslan M., Ersozlu Z., Aydođan İ., İskender M., Helvacı M. A., Turhan M., *Eğitim Bilimine Giriş*, 1. baskı, Gündüz Eğitim ve Yayıncılık, Ankara, 2009.

Ata O., Ajax Tekniđi Kullanılarak Çoktan Seçmeli Sınav Sistemi Uygulaması, Yüksek Lisans Tezi, Beykent Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul, 2008, 231653.

Baş G., Beyhan Ö., Öğretmenlerin Eğitimde Ölçme ve Deđerlendirmeye Yönelik Özyeterlik Algılarının Bazı Deđerşkenler Açısından İncelenmesi, *Eğitimde ve Psikolojide Ölçme ve Deđerlendirme Dergisi*, 2016, 7(1), 18-32.

Baştürk S. ve diđerleri, *Eğitimde Ölçme ve Deđerlendirme*, 1. baskı, Nobel, Ankara, 2014.

Bektaş M., Kudubeş A. A., Bir Ölçme ve Deđerlendirme Aracı Olarak: Yazılı Sınavlar, *DEUHYO*, 2014, 7(4), 330-336.

Çelik Z., Web Tabanlı Otomasyon ve Ölçme Deđerlendirme Yönetim Sistemi : KTÜ Fatih Eğitim Fakültesi Örneđi, Yüksek Lisans Tezi, Karadeniz Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Trabzon, 2006, 182983.

Çinici M. A., Web Tabanlı Uzaktan Eğitimde Uyarlanır Deđerlendirme Sistemi Tasarımı ve Gerçekleřtirmesi, Yüksek Lisans Tezi, Hacettepe Üniversitesi, Fen Bilimleri Enstitüsü, Ankara, 2006, 183848.

Ertürk S., *Eğitimde Program Geliřtirme*, Meteksan, Ankara, 1994.

Fidan N., Erden M., *Eğitime Giriş*, Alkım Yayınevi, Ankara 1987.

Gelbal S., *Ölçme ve Deđerlendirme*, 1. baskı, Anadolu Üniversitesi WebOfset Tesisleri, Eskişehir, 2013.

Gezgin D. M., ASP Programlama Dili ve ASP.net Teknolojisi ile E-Sınav Uygulaması, Yüksek Lisans Tezi, Trakya Üniversitesi, Fen Bilimleri Enstitüsü, Edirne, 2006, 183882.

Goldberg D. E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Longman Publishing Co. Inc., 1st ed., Boston, MA, USA. 1989.

Günoğlu S., Web Destekli Sınav Otomasyon Sistemi Tasarlanması ve Modellenmesi, Yüksek Lisans Tezi, Bahçeşehir Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul, 2008, 215944.

Holland J. H., *Adaptation in Natural and Artificial Systems*, MI: University of Michigan Press, 1st ed., Ann Arbor, USA, 1975.

İçten T., Uzaktan Eğitim Öğrencileri İçin Web Tabanlı Çevrimiçi Sınav Sistemi Uygulaması Geliştirilmesi, Yüksek Lisans Tezi, Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Ankara, 2006, 180212.

Karakaya Z., Development and Implementation of On-line Exam for a Programming Language Course, Yüksek Lisans Tezi, Orta Doğu Teknik Üniversitesi, Fen Bilimleri Enstitüsü Ankara, 2001, 116290.

McCall J., Genetic Algorithms for Modelling and Optimisation, *Journal of Computational and Applied Mathematics*, 2005, **184** (1), 205-222.

Özdemir S., İlköğretim Öğretmenlerinin Alternatif Ölçme ve Değerlendirme Araçlarına İlişkin Yeterlikleri ve Hizmet İçi Eğitim İhtiyaçları, *Türk Eğitim Bilimleri Dergisi*, 2010, **8**, 787-816.

Sarıkaya Z. C., Enformatik Derslerinin Sınavları İçin Alternatif Bir Sistemin Geliştirilmesi, Yüksek Lisans Tezi, Afyon Kocatepe Üniversitesi, Fen Bilimleri Enstitüsü, Afyon, 2011, 283420.

Sönmez V., *Eğitim Felsefesi*, Adım Yayıncılık, 3. baskı, Ankara, 1993.

Tekin H., *Eğitimde Ölçme ve Değerlendirme*, Yargı Yayınları, Ankara, 2000.

Turgut M. F., *Eğitimde Ölçme ve Değerlendirme*, Saydam Matbaacılık, Ankara, 1984.

URL-1:https://www.w3.org/wiki/The_history_of_the_Web (Ziyaret tarihi: 31.05.2018).

URL-2:<http://www.rubicite.com/Tutorials/GeneticAlgorithms/CrossoverOperators/EdgeRecombinationCrossoverOperator.aspx> (Ziyaret tarihi: 23.04.2018)

URL-3:<https://wmaraci.com/nedir/coldfusion> (Ziyaret tarihi: 23.04.2018)

URL-4:https://www.wikizero.com/en/Microsoft_SQL_Server (Ziyaret tarihi: 23.04.2018)

URL-5:https://www.wikizero.com/en/History_of_Microsoft_SQL_Server (Ziyaret tarihi: 23.04.2018)

URL-6:<https://www.wikizero.com/en/SQL> (Ziyaret tarihi: 23.04.2018)

URL-7:<https://www.w3.org/History/1989/proposal.html> (Ziyaret tarihi: 23.04.2018)

URL-8:<http://ab.org.tr/ab02/tammetin/34.doc> (Ziyaret tarihi: 23.04.2018)

Yıldırım M., A Genetic Algorithm for Generating Test From A Question Bank, *Computer Applications in Engineering Education*, 2010, **18**(2), 298-305.

Yıldırım M., Heuristic Optimization Methods for Generating Test from a Question Bank, *MICAI 2007: Advances in Artificial Intelligence*, 2007, LNAI 4827, 1218-1229.

Yılmaz N., İnsan Kaynakları Yönetimi'nin Görünen Yüzü: Fortune 500 İşletmeleri Web İçerik Analizi, Yüksek Lisans Tezi, Namık Kemal Üniversitesi, Sosyal Bilimler Enstitüsü, Tekirdağ, 2015, 414661.



EKLER

EK-A

Kullanıcı Giriş Formu

```
<cform method="POST">
  <input type="hidden" name="aktif" value="1">
  <div class="row">
    <div class="col-lg-4 col-lg-offset-4" style="margin-top: 50px; border: solid 1px
#CCC; border-radius: 5px;">
      <!-- Logo kaynağı: https://www.flaticon.com/free-icon/test_240062-->
      <div align="center" style="margin: 20px;"><a
href="http://www.kocaeli.edu.tr/" target="_blank"></a></div>
      <p align="center">
        <cfif isDefined("Form.aktif")>
          <font color="000080"><b>Geçersiz, Tekrar Deneyin.</b></FONT>
        </cfif>
      </p>
      <div class="container-fluid">
        <form id="GIRIS" class="form-horizontal">
          <fieldset>
            <h4 class="text text-default" align="center"><b>AKILLI SORU
BANKASI SİSTEMİ</b></h4>
            <hr />
            <div class="col-lg-12">
              <div class="form-group">
                <div class="col-lg-8 col-lg-offset-2">
                  <input type="text" name="kullaniciNo" id="kullaniciNo"
class="form-control" maxlength="4" placeholder="Kullanıcı Numaranız" required />
                  <div class="alert alert-danger hidden"
id="kullaniciNoUyari"></div>
                </div>
              </div>
              <div class="col-lg-12" style="margin-top: 10px;">
                <div class="form-group">
                  <div class="col-lg-8 col-lg-offset-2">
                    <input type="password" name="Sifre" class="form-control"
id="Sifre" placeholder="Parolanız" required />
                    <div class="alert alert-danger hidden" id="SifreUyari"></div>
                  </div>
                </div>
              <div class="col-lg-12" style="margin-top: 10px; margin-bottom: 10px;">
                <div class="form-group">
                  <div class="col-lg-4"></div>
                  <div class="col-lg-4" align="center">
                    <button name="Ara" id="Ara" class="btn btn-primary btn-block"
/>GİRİŞ</button>
```

```
        <a href="yeniKayit.cfm" name="kaydol" id="Kaydol" class="btn
btn-warning btn-block" />KAYIT OL</a>
    </div>
    <div class="col-lg-4"></div>
</div>
</div>
</fieldset>
</form>
</div>
</div>
</div>
</cform>
```



EK-B

Konular.cfm dosyası

```
<cfinclude template="menuler.cfm">
<cfparam name="url.konuNo" default="0">
<cfoutput>
  <cfset sorgu = createObject("component", "sorgular")>
  <cfif url.konuNo gt 0>
    <cfquery datasource="soruBankasi" name="konuSil" result="deleteRes">
      DELETE FROM konular WHERE konuNo = <cfqueryparam
cfsqltype="cf_sql_integer" value="#url.konuNo#">
    </cfquery>
    <cfif not deleteRes.recordCount>
      <div class="alert alert-danger">
        Silme işlemi başarısız!
      </div>
    <cfelse>
      <div class="alert alert-success">
        Silme işlemi başarılı!
      </div>
    </cfif>
    <cfset sorular = sorgu.getKonular(Session.KullaniciNo)>
    <div class="table-responsive-md">
      <table class="table">
        <thead>
          <tr>
            <th>Konu Adı</th>
            <th>Soru Sayısı</th>
            <th>Eklenme Tarihi</th>
            <th colspan="2"><a href="konuOlustur.cfm" class="btn btn-success"
title="Yeni Konu Ekle"><span class="glyphicon glyphicon-plus"></span></th>
          </tr>
        </thead>
        <cfloop query="sorular">
          <tr>
            <td>#konuAdi#</td>
            <td>#SoruSayisi#</td>
            <td>#DateFormat(eklenmeTarihi, "dd/mm/yyyy")#</td>
            <td><a href="konuOlustur.cfm?konuNo=#konuNo#" class="btn btn-
warning" title="Konu Düzelt"><span class="glyphicon glyphicon-
edit"></span></td>
            <td><a href="konular.cfm?konuNo=#konuNo#&Action=Delete"
class="btn btn-danger" title="Konu Sil. Girilmiş soru olmaması
gerekmektedir."><span class="glyphicon glyphicon-remove"></span></td>
          </tr>
        </cfloop>
      </table>
```

```
</div>  
</cfif>  
</cfoutput>  
<cfinclude template="footer.cfm">
```



EK-C

KonuOlustur.cfm dosyası

```
<cfinclude template="menuler.cfm">
<cfparam name="url.konuNo" default="0">
<cfoutput>
<cfset kayitKontrol = true>
<cfset sorgu = createObject("component", "sorgular")>
<cfif isDefined("form.is_form_submitted")>
    <cfif url.konuNo gt 0>
        <cfquery datasource="soruBankasi" name="KonuGuncelle"
result="updateRes">
            UPDATE konular
            SET konuAdi = <cfqueryparam cfsqltype="cf_sql_char"
value="#UCase(form.konuAdi)#">
            WHERE konuNo = <cfqueryparam cfsqltype="cf_sql_integer"
value="#url.konuNo#">
        </cfquery>
        <div class="col-lg-4"></div>
        <cfif updateRes.RecordCount>
            <cfset kayitKontrol = false>
            <div class="alert alert-success col-lg-4" style="text-align:center;">
                "#Ucase(form.konuAdi)#" konusu başarıyla güncellendi.<br />
                <a href="#CGI.SCRIPT_NAME#">Yeni konu eklemek için lütfen
tıklayınız.</a>
            </div>
        <cfelse>
            <div class="alert alert-danger col-lg-4" style="text-align:center;">
                Güncelleştirme işlemi gerçekleştirilemedi! Lütfen tekrar deneyiniz.
            </div>
        </cfif>
        <div class="col-lg-4"></div>
    <cfelse>
        <cfquery datasource="soruBankasi" name="KonuEkle">
            INSERT INTO konular(konuAdi, kullanıcıNo)
            VALUES    (<cfqueryparam cfsqltype="cf_sql_char"
value="#UCase(form.konuAdi)#">,
                <cfqueryparam cfsqltype="cf_sql_char"
value="#session.KullaniciNo#">)
            SELECT @@IDENTITY AS KonuNo
        </cfquery>
        <div class="col-lg-4"></div>
        <cfif KonuEkle.RecordCount>
            <cfset kayitKontrol = false>
            <div class="alert alert-success col-lg-4" style="text-align:center;">
                "#Ucase(form.konuAdi)#" konusu başarıyla eklendi.<br />
                <a href="#CGI.SCRIPT_NAME#">Yeni konu eklemek için lütfen
tıklayınız.</a>
```

```

        </div>
    </cfelse>
    <div class="alert alert-danger col-lg-4" style="text-align:center;">
        Yeni konu oluřuturalamadı! Lütfen tekrar deneyiniz.
    </div>
</cfif>
<div class="col-lg-4"></div>
</cfif>
</cfif>
<cfif kayıtKontrol>
<form action="#CGI.SCRIPT_NAME#?konuNo=#url.konuNo#" method="post">
<div class="table-responsive-md">
    <table class="table">
        <tr>
            <th class="col-lg-2">
                Konu
            </th>
            <td class="col-lg-10">
                <div class="col-lg-4">
                    <cfif url.konuNo gt 0>
                        <input type="text" name="konuAdi" required class="form-control"
placeholder="Matematik-I" style="text-transform:uppercase;"
value="#sorgu.getKonu(url.konuNo).konuAdi#">
                    </cfif>
                    <input type="text" name="konuAdi" required class="form-control"
placeholder="Matematik-I" style="text-transform:uppercase;">
                </div>
            </td>
            <tr>
            <tr>
                <td colspan="2" style="text-align:center;">
                    <input type="hidden" name="is_form_submitted" id="is_form_submitted"
value="1">
                    <input type="submit" name="konuKaydet" value="<cfif url.konuNo gt
0>Güncelle</cfif>Add</cfif>" class="btn btn-submit btn-success">
                </td>
            </tr>
        </table>
    </div>
</form>
</cfif>
</cfoutput>
<cfinclude template="footer.cfm">

```


EK-D

SoruOlustur.cfm dosyası

```
<cfinclude template="menuler.cfm">
<cfif isDefined("form.is_form_submitted") and not isDefined('form.soruid')>
<cflock timeout="60">
<cftransaction>
<cfquery datasource="sorubankasi" name="insert_sor" result="insert_soru">
INSERT INTO dbo.soru
(soruKoku
,konuNo
,zorluk
,celdirici
,frekans
,silindi
,eklenmeTarihi)
VALUES
(<cfif len(form.soru)>#form.soru#<cfelse>NULL</cfif>
,<cfif len(form.konu)>#form.konu#<cfelse>NULL</cfif>
,<cfif len(form.zorluk)>#form.zorluk#<cfelse>NULL</cfif>
,<cfif len(form.celdiri)>#form.celdiri#<cfelse>NULL</cfif>
,0
,0
,getdate()
)
</cfquery>
<cfloop list="A,B,C,D,E" index="itm">
<cfquery datasource="sorubankasi" name="insert_cevap">
INSERT INTO dbo.cevaplar
(soruNo
,cevapAdi
,cevapKoku
,dogruecevap
,silindi
,eklenmeTarihi)
VALUES
(#insert_soru.identitycol#
,#itm#
,#evaluate("form.#itm#")#
,<cfif form.dogruecevap eq #itm#>1<cfelse>0</cfif>
,0
,getdate()
)
</cfquery>
</cfloop>
<cftransaction>
</cflock>
<cflocation url="sorular.cfm">
```

```

<cfelseif isDefined("form.is_form_submitted") and isDefined('form.soruId')>
<cflock timeout="60">
<cftransaction>
<cfquery name="make_passive_cevaplar" datasource="sorubankasi">
UPDATE dbo.cevaplar
SET silindi = 1
WHERE soruNo = #form.soruId#
</cfquery>
<cfquery name="update_soru" datasource="sorubankasi">
UPDATE dbo.Soru
SET soruKoku = '#form.soru#'
,konuNo = '#form.konu#'
,zorluk = #form.zorluk#
,celdirici = #form.celdirici#
WHERE soruNo = #form.soruId#
</cfquery>
<cfloop list="A,B,C,D,E" index="itm">
<cfquery datasource="sorubankasi" name="insert_cevap">
INSERT INTO dbo.cevaplar
(soruNo
,cevapAdi
,cevapKoku
,dogruCevap
,silindi
,eklenmeTarihi)
VALUES
(#form.soruId#
,'#itm#'
,'#evaluate("form.#itm#")#'
,<cfif form.dogru eq #itm#>1<cfelse>0</cfif>
,0
,getdate()
)
</cfquery>
</cfloop>
</cftransaction>
</cflock>
<cflocation url="sorular.cfm">
</cfif>
<cfif isdefined("url.soruid") and len(url.soruid)>
<cfquery name="get_soru_detail" datasource="sorubankasi">
SELECT
soruKoku
,konuNo
,zorluk
,celdirici
,frekans
,silindi

```

```

,eklenmeTarihi
FROM
dbo.soru
WHERE
soruNo = #url.soruid#
and silindi = 0
</cfquery>
<cfquery name="get_cevaplar_detail" datasource="sorubankasi">
select
cevapNo,soruNo,cevapAdi,cevapKoku,dogruCevap,resim,silindi,eklenmeTarihi
from SoruBankasi.dbo.cevaplar
where soruNo = #url.soruid# AND silindi = 0
</cfquery>
</cfif>
<cfoutput>
<cfset sorgular = createObject("component", "sorgular")>
<form action="#CGI.ScriptName#" method="post" enctype="multipart/form-data">
<cfif isDefined('url.soruid')>
<input type="hidden" name="soruId" id="soruId" value="#url.soruid#">
</cfif>
<div class="table-responsive-md">
<table class="table">
<tr>
<th class="col-lg-2">
<!--Konu-->Subject
</th>
<td class="col-lg-10">
<select name="konu" class="form-control">
<option value="0"><!--Seçiniz...-->Select...</option>
<cfset konular = sorgular.getKonular(session.kullaniciNo)>
<cfloop query="konular">
<option value="#konuNo#" <cfif isdefined('get_soru_detail') and
get_soru_detail.konuNo eq konuNo>selected</cfif>>
#konuAdi#
</option>
</cfloop>
</select>
</td>
</tr>
<tr>
<th>
<!--Soru-->Body
</th>
<td>
<textarea name="soru" width="100%" height="200" class="form-control"><cfif
isDefined('get_soru_detail')>#get_soru_detail.soruKoku#</cfif></textarea>
</td>
</tr>

```

```

<cfset cevapList = "A,B,C,D,E">
<cfloop index="x" list="#cevapList#">
<cfif isdefined('get_cevaplar_detail')>
<cfquery name="get_cevap_detail_sub" dbtype="query">
select * from get_cevaplar_detail where cevapAdi = '#x#'
</cfquery>
</cfif>
<tr>
<th>
<div class="radio">
<label><input type="radio" name="dogru"><cfif isdefined('get_cevap_detail_sub')
and get_cevap_detail_sub.dogruevap eq 1>checked</cfif>
value="#x#">#x#</label>
</div>
</th>
<td>
<textarea name="#x#" width="100%" height="200" class="form-control"><cfif
isdefined('get_cevap_detail_sub')>#get_cevap_detail_sub.cevapKoku#</cfif></textare
a>
</td>
</tr>
</cfloop>
<tr>
<th>
<!--Öngörülen Zorluk-->Difficulty (<span id="zorlukDeger">3</span>)
</th>
<td>
<div class="col-lg-1" style="text-align:right">1</div><div class="col-lg-4"><input
type="range" name="zorluk" id="zorluk" min="1" max="5"><cfif
isdefined('get_soru_detail')>value="#get_soru_detail.zorluk#"<cfelse>value="3"</cf
if> style="width:100%;"></div><div class="col-lg-1">5</div>
</td>
</tr>
<tr>
<th>
<!--Öngörülen Ayırt Edicilik-->Discrimination Index (<span
id="celdiriDeger">3</span>)
</th>
<td>
<div class="col-lg-1" style="text-align:right">1</div><div class="col-lg-4"><input
type="range" name="celdiri" id="celdiri" min="1" max="5" <cfif
isdefined('get_soru_detail')>value="#get_soru_detail.celdirici#"<cfelse>value="3"</
cfif> style="width:100%;"></div><div class="col-lg-1">5</div>
</td>
</tr>
<!------<tr>
<th>
Soru Resim

```

```

</th>
<td>
<input type="file" name="resim">
</td>
</tr>----->
<tr>
<td colspan="2" style="text-align:center;">
<input type="hidden" name="is_form_submitted" id="is_form_submitted"
value="1">
<input type="submit" name="soruKaydet" value="<!--Soru Kaydet-->Add"
class="btn btn-submit btn-success">
</td>
</tr>
</table>
</div>
</form>
</cfoutput>
<script>
var slider = document.getElementById("zorluk");
var output = document.getElementById("zorlukDeger");
output.innerHTML = slider.value;

slider.oninput = function() {
output.innerHTML = this.value;
}

var slider2 = document.getElementById("celdiri");
var output2 = document.getElementById("celdiriDeger");
output2.innerHTML = slider2.value;

slider2.oninput = function() {
output2.innerHTML = this.value;
}
</script>
<cfinclude template="footer.cfm">

```

EK-E

Sinavlar.cfm dosyası

```
<cfinclude template="menuler.cfm">
<style>.orta{ text-align:center;}</style>
<cfoutput>
  <cfset sorgular = createObject("component", "sorgular")>
  <cfset kitapkodulistesisi = "A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,R,S,T,U,V,Y,Z">
  <cfif isdefined("url.sinavNo")>
    <cfset sinavBilgileri = sorgular.getSinavBilgileri(url.sinavNo)>
    <cfdump var="#sinavBilgileri#">
  <cfelse>
  <div class="table-responsive-md">
    <table class="table table-hover table-condensed" style="text-align: center;">
      <tr>
        <th>
          Sınav Adı
        </th>
        <th style="text-align: center;">
          Tarihi
        </th>
        <th>
          Konular
        </th>
        <th style="text-align: center;">
          Soru<br />Sayısı
        </th>
        <th style="text-align: center;">
          Sınav<br>Süresi
        </th>
        <th style="text-align: center;">
          Kitapçık<br />Sayısı
        </th>
        <th style="text-align: center;">
          Kitapçık<br />Kodu
        </th>
        <th style="text-align: center;">
          Zorluk
        </th>
        <th style="text-align: center;">
          Ayırt<br />Edicilik
        </th>
        <th style="text-align: center;">
          Frekans
        </th>
        <th colspan="5"><a href="sinavOlustur.cfm" class="btn btn-success btn-
block" title="Yeni Sınav Ekle"><span class="glyphicon glyphicon-
plus"></span></th>
```

```

</tr>
<cfset sinavlar = sorgular.getSinavlar(Session.KullaniciNo)>
<cfloop query="sinavlar">
  <tr>
    <th>
      #sinavAdi#
    </th>
    <td style="font-size:12px;">
      <i>#DateFormat(sinavTarihi,
"dd/mm/yyyy")#<br/>#TimeFormat(sinavTarihi, "HH:mm")#</i>
    </td>
    <td style="text-align:left;font-size:12px;">
      <cfset konuListe = sorgular.getKonuListesi(konuListesi)>
      <cfset i = 0>
      <i>
        <cfloop query="konuListe">
          #konuAdi###+i lt konuListe.recordCount ? ", " : ""#
        </cfloop>
      </i>
    </td>
    <td class="orta">
      #soruSayisi#
    </td>
    <td class="orta">
      #sinavSuresi#
    </td>
    <td class="orta">
      #kitapcikSayisi#
    </td>
    <td class="orta">
      #kitapcikBaslangicKodu#
    </td>
    <td class="orta">
      #zorluk#
    </td>
    <td class="orta">
      #celdirici#
    </td>
    <td class="orta">
      #frekans#
    </td>
    <td><a
href="soruKitapcikOlustur.cfm?sinavNo=#sinavNo#&Action=Completed"
class="btn btn-info btn-block" title="Soru Kitapçıkları"><span class="glyphicon
glyphicon-book"></span></td>
      <td><a href="sinavDegerlendir.cfm?sinavNo=#sinavNo#" class="btn
btn-success btn-block" title="Sınav Değerlendir"><span class="glyphicon
glyphicon-ok-sign"></span></td>

```

```
<td><a href="sinavSonuclari.cfm?sinavNo=#sinavNo#" class="btn btn-
primary btn-block" title="Sınav Sonuçları"><span class="glyphicon glyphicon-
list"></span></td>
<td><a href="sinavOlustur.cfm?sinavNo=#sinavNo#&Action=Copy"
class="btn btn-warning btn-block" title="Sınavı Kopyala ve Düzenle"><span
class="glyphicon glyphicon-edit"></span></td>
<td><a href="sinavOlustur.cfm?sinavNo=#sinavNo#&Action=Delete"
class="btn btn-danger btn-block" title="Sınavı Sil"><span class="glyphicon
glyphicon-remove"></span></td>
</tr>
</cfloop>
</table>
</div>
</cfif>
</cfoutput>
<cfinclude template="footer.cfm">
```


EK-F

SinavOlustur.cfm dosyası

```
<cfinclude template="menuler.cfm">
<cfoutput>
  <cfset sorgular = createObject("component", "sorgular")>
  <cfset kitapkodulistesisi = "A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,R,S,T,U,V,Y,Z">
  <form action="#CGI.ScriptName#" method="post" enctype="multipart/form-
data">
  <cfif isDefined('url.sinavNo')>
    <input type="hidden" name="sinavNo" id="sinavNo" value="#url.sinavNo#">
  </cfif>
  <div class="table-responsive-md">
    <table class="table">
      <tr>
        <th class="col-lg-2">
          Sınav Adı
        </th>
        <td class="col-lg-10">
          <div class="col-lg-4">
            <input type="text" name="sinavAdi" required class="form-control"
placeholder="Matematik-I">
          </div>
        </td>
      </tr>
      <tr>
        <th class="col-lg-2">
          Sınav Tarihi
        </th>
        <td class="col-lg-10">
          <div class="col-lg-4">
            <input type="datetime-local" name="sinavTarihi" required
class="form-control" value="#DateFormat(Now(), 'yyyy-mm-
dd')#T#TimeFormat(Now(), 'HH:mm')#" min="#DateFormat(Now(), 'yyyy-mm-
dd')#T#TimeFormat(Now(), 'HH:mm')#">
          </div>
        </td>
      </tr>
      <tr>
        <th class="col-lg-2">
          Konular
        </th>
        <td class="col-lg-10">
          <div class="col-lg-4">
            <cfset konular = sorgular.getKonular(session.kullaniciNo)>
            <select name="konu" class="form-control" required multiple
title="Çoklu seçim için CTRL tuşuna basarak seçim yapınız.">
              <cfloop query="konular">
```

```

                <option value="#konuNo#">
                    #konuAdi#
                </option>
            </cfloop>
        </select>
    </div>
</td>
</tr>
<tr>
    <th class="col-lg-2">
        Sınav Soru Sayısı
    </th>
    <td class="col-lg-10">
        <div class="col-lg-4">
            <input type="number" name="soruSayisi" class="form-control"
value="10">
        </div>
    </td>
</tr>
<tr>
    <th class="col-lg-2">
        Sınav Süresi
    </th>
    <td class="col-lg-10">
        <div class="col-lg-4">
            <input type="number" name="sinavSuresi" required class="form-
control" value="50">
        </div>
    </td>
</tr>
<tr>
    <th class="col-lg-2">
        Kitapçık Sayısı
    </th>
    <td class="col-lg-10">
        <div class="col-lg-4">
            <select name="kitapciksayisi" class="form-control" required>
                <option value="0">Kaç kitapçık?</option>
                <cfloop index="i" from="1" to="10">
                    <option value="#i#">#i#</option>
                </cfloop>
            </select>
        </div>
    </td>
</tr>
<tr>
    <th class="col-lg-2">
        Kitapçık Kodu Başlangıcı

```

```

</th>
<td class="col-lg-10">
  <div class="col-lg-4">
    <select name="kitapcik kodlari" class="form-control" required>
      <option>İlk kitapçık kodu</option>
      <cfloop list="#kitapkodulistesesi#" index="eleman">
        <option value="#eleman#">#eleman#</option>
      </cfloop>
    </select>
  </div>
</td>
</tr>
<tr>
  <th>
    Zorluk (<span id="zorlukDeger">3</span>)
  </th>
  <td>
    <div class="col-lg-1" style="text-align:right">1</div><div class="col-
lg-4"><input type="range" name="zorluk" id="zorluk" min="1" max="5"<cfif
isdefined('get_soru_detail')>value="#get_soru_detail.zorluk#"<cfelse>value="3"</cf
if> style="width:100%;"></div><div class="col-lg-1">5</div>
  </td>
</tr>
<tr>
  <th>
    Ayırt Edicilik (<span id="celdiriDeger">3</span>)
  </th>
  <td>
    <div class="col-lg-1" style="text-align:right">1</div><div class="col-
lg-4"><input type="range" name="celdiri" id="celdiri" min="1" max="5" <cfif
isdefined('get_soru_detail')>value="#get_soru_detail.celdirici#"<cfelse>value="3"</
cfif> style="width:100%;"></div><div class="col-lg-1">5</div>
  </td>
</tr>
<tr>
  <th>
    Frekans (<span id="frekansDeger">0</span>)
  </th>
  <td>
    <div class="col-lg-1" style="text-align:right">0</div>
    <div class="col-lg-4">
      <input type="range" name="frekans" id="frekans" min="0" max="10"
value="0" style="width:100%;">
    </div>
    <div class="col-lg-1">10</div>
  </td>
</tr>
</tr>

```

```

        <td colspan="2" style="text-align:center;">
            <input type="hidden" name="is_form_submitted"
id="is_form_submitted" value="1">
            <input type="submit" name="sinavKaydet" value="Sınav Oluştur"
class="btn btn-submit btn-success">
        </td>
    </tr>
</table>
</div>
</form>
</cfoutput>
<script>
    var slider = document.getElementById("zorluk");
    var output = document.getElementById("zorlukDeger");
    output.innerHTML = slider.value;

    slider.oninput = function() {
        output.innerHTML = this.value;
    }

    var slider2 = document.getElementById("celdiri");
    var output2 = document.getElementById("celdiriDeger");
    output2.innerHTML = slider2.value;

    slider2.oninput = function() {
        output2.innerHTML = this.value;
    }

    var slider3 = document.getElementById("frekans");
    var output3 = document.getElementById("frekansDeger");
    output3.innerHTML = slider0.value;

    slider3.oninput = function() {
        output3.innerHTML = this.value;
    }
</script>
<cfinclude template="footer.cfm">

```

EK-G

Sorgular.cfc dosyası

```
<cfcomponent>
  <cffunction name="getUsers" access="public" returntype="query">
    <cfargument name="kullaniciNo" type="numeric" required="no">
    <cfquery datasource="soruBankasi" name="usersQry">
      SELECT TOP (200) kullanici.kullaniciNo, kullanici.kullaniciAdi,
kullanici.kullaniciSoyadi, unvan.unvanAdi
      FROM  kullanici INNER JOIN
            unvan ON kullanici.unvanNo = unvan.unvanNo
    <cfif isdefined("arguments.userID")>
      WHERE (kullaniciNo = #arguments.userID#)
    </cfif>
    ORDER BY  unvan.unvanNo
    </cfquery>
    <cfreturn usersQry>
  </cffunction>
  <cffunction name="getSorular" access="public" returnType="query">
    <cfargument name="userid" required="yes">
    <cfquery name="sorularSrg" datasource="sorubankasi">
      SELECT
        s.soruNo,
        S.soruKoku,
        K.konuAdi,
        S.celdirici,
        S.zorluk
      FROM
        dbo.soru S WITH(NOLOCK)
        INNER JOIN dbo.konular K WITH(NOLOCK) ON S.konuNo =
K.konuNo
        --INNER JOIN dbo.cevaplar C WITH(NOLOCK) ON C.soruNo =
S.soruNo
      WHERE kullaniciNo = #userid#
    </cfquery>
    <cfreturn sorularSrg>
  </cffunction>
  <cffunction name="getKonu" access="public" returnType="query">
    <cfargument name="konuNo" type="numeric" required="yes">
    <cfquery datasource="soruBankasi" name="konuQry">
      SELECT TOP (100) PERCENT konuNo, konuAdi, kullaniciNo, silindi,
eklenmeTarihi
      FROM  dbo.konular
      WHERE (konuNo = #arguments.konuNo#)
    </cfquery>
    <cfreturn konuQry>
  </cffunction>
  <cffunction name="getKonuListesi" access="public" returnType="query">
```

```

    <cfargument name="konuListe" type="string" required="yes">
    <cfquery datasource="soruBankasi" name="konuListeQry">
        SELECT TOP (100) PERCENT konuNo, konuAdi, kullaniciNo, silindi,
eklenmeTarihi
        FROM dbo.konular
        WHERE (konuNo IN (#arguments.konuListe#))
    </cfquery>
    <cfreturn konuListeQry>
</cffunction>
<cffunction name="getKonular" access="public" returnType="query">
    <cfargument name="userID" type="numeric" required="yes">
    <cfquery datasource="soruBankasi" name="konularQry">
        SELECT TOP (100) PERCENT konular.konuNo, konular.konuAdi,
konular.kullaniciNo, konular.silindi, konular.eklenmeTarihi, COUNT(soru.soruNo)
AS SoruSayisi
        FROM konular LEFT OUTER JOIN
            soru ON konular.konuNo = soru.konuNo
        WHERE (konular.kullaniciNo = #userID#) AND
            (konular.silindi = 0)
        GROUP BY konular.konuNo, konular.konuAdi, konular.kullaniciNo,
konular.silindi, konular.eklenmeTarihi
        ORDER BY konular.konuAdi
    </cfquery>
    <cfreturn konularQry>
</cffunction>
<cffunction name="getSinavBilgileri" access="public" returnType="query">
    <cfargument name="sinavNo" type="numeric" required="yes">
    <cfquery datasource="soruBankasi" name="SinavBilgisi">
        SELECT dbo.kullanici.kullaniciNo, dbo.unvan.unvanAdi,
dbo.kullanici.kullaniciAdi, dbo.kullanici.kullaniciSoyadi,
            dbo.sinav.sinavNo, dbo.sinav.sinavAdi, dbo.sinav.sinavTarihi,
dbo.sinav.kitapcikSayisi, dbo.sinav.kitapcikBaslangicKodu,
            dbo.sinav.soruSayisi, dbo.sinav.zorluk, dbo.sinav.celdirici,
dbo.sinav.frekans, dbo.sinav.sinavSuresi
        FROM dbo.sinav INNER JOIN
            dbo.kullanici ON dbo.sinav.kullaniciNo = dbo.kullanici.kullaniciNo
INNER JOIN
            dbo.unvan ON dbo.kullanici.unvanNo = dbo.unvan.unvanNo
        WHERE (dbo.sinav.sinavNo = #sinavNo#)
    </cfquery>
    <cfreturn SinavBilgisi>
</cffunction>
<cffunction name="getSinavKitapcikBilgileri" access="public"
returnType="query">
    <cfargument name="sinavNo" type="numeric" required="yes">
    <cfquery datasource="soruBankasi" name="SinavKitapcikBilgileri">
        SELECT DISTINCT TOP (100) PERCENT kitapcikKodu
        FROM dbo.sinavSorulari

```

```

WHERE (sinavNo = #sinavNo#)
ORDER BY kitapcikKodu
</cfquery>
<cfreturn SinavKitapcikBilgileri>
</cffunction>
<cffunction name="getKonuSoruSayilari" access="public" returnType="query">
  <cfargument name="sinavNo" type="numeric" required="yes">
  <cfquery datasource="soruBankasi" name="KonuSoruSayilari">
    SELECT TOP (100) PERCENT dbo.konular.konuAdi,
COUNT(dbo.konular.konuNo) AS konuSoruSayisi, dbo.konular.konuNo
FROM dbo.sinavSorulari INNER JOIN
    dbo.soru ON dbo.sinavSorulari.soruNo = dbo.soru.soruNo INNER JOIN
    dbo.konular ON dbo.soru.konuNo = dbo.konular.konuNo
WHERE (dbo.sinavSorulari.sinavNo = #sinavNo#)
GROUP BY dbo.konular.konuAdi, dbo.konular.konuNo
ORDER BY dbo.konular.konuAdi
  </cfquery>
  <cfreturn KonuSoruSayilari>
</cffunction>
<cffunction name="getSinavSorulari" access="public" returnType="query">
  <cfargument name="sinavNo" type="numeric" required="yes">
  <cfargument name="kitapcikkodu" type="string" required="yes">
  <cfquery datasource="soruBankasi" name="Sorular">
    SELECT dbo.soru.soruNo, dbo.soru.soruKoku,
dbo.sinavSorulari.kitapcikKodu, dbo.soru.resim, dbo.sinavSorulari.kitapcikSoruNo
FROM dbo.sinavSorulari INNER JOIN
    dbo.soru ON dbo.sinavSorulari.soruNo =
dbo.soru.soruNo
WHERE (dbo.sinavSorulari.sinavNo = #sinavNo#) AND
(dbo.sinavSorulari.kitapcikKodu =
'#kitapcikkodu#')
ORDER BY dbo.sinavSorulari.kitapcikSoruNo
  </cfquery>
  <cfreturn Sorular>
</cffunction>
<cffunction name="getSoruDetay" access="public" returnType="query">
  <cfargument name="soruNo" type="numeric" required="yes">
  <cfquery datasource="soruBankasi" name="soruDetay">
    SELECT cevapNo, cevapAdi, cevapKoku, dogruCevap, resim
FROM cevaplar
WHERE (soruNo = #soruNo#) AND (silindi = 0)
ORDER BY cevapAdi
  </cfquery>
  <cfreturn soruDetay>
</cffunction>
<cffunction name="getKonuSorulari" access="public" returnType="query">
  <cfargument name="sinavNo" type="numeric" required="yes">
  <cfargument name="konuNo" type="numeric" required="yes">

```

```

    <cfquery datasource="soruBankasi" name="KonuSorular">
        SELECT      dbo.soru.soruNo, dbo.soru.soruKoku,
dbo.sinavSorulari.kitapcikKodu, dbo.soru.resim, dbo.sinavSorulari.kitapcikSoruNo
        FROM dbo.sinavSorulari INNER JOIN
                dbo.soru ON dbo.sinavSorulari.soruNo =
dbo.soru.soruNo
        WHERE      (dbo.sinavSorulari.sinavNo = #sinavNo#) AND
                (dbo.soru.konuNo = #konuNo#)
        ORDER BY  dbo.sinavSorulari.kitapcikSoruNo
    </cfquery>
    <cfreturn KonuSorular>
</cffunction>
<cffunction name="getUnvanlar" access="public" returnType="query">
    <cfquery datasource="soruBankasi" name="unvanlarQry">
        SELECT  TOP (100) PERCENT unvanNo, unvanAdi
        FROM    dbo.unvan
        ORDER BY unvanNo
    </cfquery>
    <cfreturn unvanlarQry>
</cffunction>
<cffunction name="getSinavlar" access="public" returnType="query">
    <cfargument name="KullaniciNo" type="numeric" required="yes">
    <cfquery datasource="soruBankasi" name="Sinavlar">
        SELECT DISTINCT  sinavNo, sinavAdi, sinavTarihi, kitapcikSayisi,
kitapcikBaslangicKodu, soruSayisi, zorluk, celdirici, frekans, sinavSuresi,
konuListesi
        FROM              sinav
        WHERE              (kullaniciNo = #kullaniciNo#) AND
                (silindi = 0)
        ORDER BY          sinavTarihi DESC
    </cfquery>
    <cfreturn Sinavlar>
</cffunction>
</cfcomponent>

```


EK-H

Secim.cfc dosyası

```
<cfcomponent output="yes">
<cfsetting showdebugoutput="yes">
<cffunction name="Test" access="public" returntype="string">
<cfargument name="veri" type="string" required="yes">
<cfargument name="soruSayisi" type="numeric" required="yes">
<cfargument name="zorlukDerecesi" type="numeric" required="yes">
<cfargument name="celdiriDerecesi" type="numeric" required="yes">
<cfset veriSeti = CFJSON2Query(veri)><!-- JSON olarak gelen verinin dizi haline
döndürülmesi. --->
<cfset soruidList = ValueList(veriSeti.Soruid)>
<cfset soruidDizi = ListToArray(ValueList(veriSeti.Soruid))>
<cfset veriBoyutu = veriSeti.RecordCount><!-- Verinin uzunluğu ne kadar?--->
<cfset populasyonSayisi = 50><!-- Kaç birey olacak? --->
<cfset maxJenerasyonSayisi = 2><!-- Kaç jenerasyon olacak? --->
<cfset esSayisi = populasyonSayisi><!-- Kaç eş olacak?--->
<cfset mutasyonOrani = 0.02><!-- Mutasyon olasılığı kaç? --->
<cfset siraDizisi = siraDiziOlustur(veriBoyutu)><!-- 1'den N'e kadar olan sıralı
elemanları olan dizi oluşturur. --->
<cfset amacFonk = arrayNew(2)><!-- Çıktıyı tutan dizi --->
<!-- İlk populasyonun oluşturulması--->
<cfset populasyonDecision = initPopulation(populasyonSayisi,
Arguments.soruSayisi, populasyonSayisi, siraDizisi)>
<cfset maliyet = objectiveFunction(populasyonDecision, veriSeti,
Arguments.zorlukDerecesi, Arguments.celdiriDerecesi)>
<cfset fitness = fitnessFunction(maliyet, populasyonSayisi)>
<!-- Jenerasyon --->
<cfloop index="i" from="1" to="#maxJenerasyonSayisi#">
<!-- Elitist --->
<cfset maxElemanlar = maxElemanBul(fitness)>
<cfset p = maxElemanlar[1]>
<cfset r = maxElemanlar[2]>
<cfset elitist = populasyonDecision[r]>
<!-- Seçim --->
<cfset selected = selectFunction(fitness, populasyonSayisi)>
<cfset parent = parentFunction(populasyonDecision, selected, populasyonSayisi)>
<cfset parent1 = parent[1]>
<cfset parent2 = parent[2]>
<!-- Çaprazlama --->
<cfset crossOverPopulasyon = crossOverFunction(parent1, parent2,
populasyonSayisi, Arguments.soruSayisi, veriBoyutu)>
<!-- Mutasyon --->
<cfset mutasyonPopulasyon = mutationFunction(crossOverPopulasyon, veriBoyutu,
mutasyonOrani)>
<!-- Tekrardan amaç fonksiyondan maliyet üretiliyor. Ayrıca maliyete ait fitness
değeri üretiliyor. --->
```

```

<cfset maliyet = objectiveFunction(mutasyonPopulasyon, veriSeti,
Arguments.zorlukDerecesi, Arguments.celdiriDerecesi)>
<cfset fitness = fitnessFunction(maliyet, populasyonSayisi)>
<!--- Elitizm --->
<cfset minElemanlar = minElemanBul(fitness)>
<cfset p = minElemanlar[1]>
<cfset r = minElemanlar[2]>
<cfset mutasyonPopulasyon[r] = elitist>
<!--- Yeni nüfus --->
<cfset populasyonDecision = mutasyonPopulasyon>
<cfset maliyet = objectiveFunction(populasyonDecision, veriSeti,
Arguments.zorlukDerecesi, Arguments.celdiriDerecesi)>
<cfset fitness = fitnessFunction(maliyet, populasyonSayisi)>
<cfset amacFonk[i] = [i, arrayMin(maliyet)]>
<cfif amacFonk[i][2] eq 1>
<cfbreak/>
</cfif>
</cfloop>
<cfset cozum = arraySort(amacFonk, numeric)>
<cfreturn amacFonk>
</cffunction>
<cfscript>
//Mutasyon yapan fonksiyon.
public array function mutationFunction(required array crossOverPopulasyon,
required numeric veriBoyutu, required numeric mutasyonOrani){
muDec = crossOverPopulasyon;
x = arrayLen(crossOverPopulasyon);
y = arrayLen(crossOverPopulasyon[1]);
for(i = 1; i <= x; i++){
for(j = 1; j <= y; j++){
rndSayi = randomSayi();
if(rndSayi < mutasyonOrani){
do{
qNo = randomRangeSayi(1, veriBoyutu);
} while(arrayFind(crossOverPopulasyon[i], qNo) > 0);
muDec[i][k] = qNo;
}
}
}
writeDump(var = muDec, label = "Mutasyon Dizisi");
return muDec;
}

// Çaprazlama yapan fonksiyon.
public array function crossOverFunction(required array parent1, required array
parent2, required numeric populasyonSayisi, required numeric soruSayisi, required
numeric veriBoyutu){
donguDegeri = populasyonSayisi/2;

```

```

nokta = randomRangeSayi(1, soruSayisi-1);
cocuk1 = arrayNew(1);
cocuk2 = arrayNew(1);
for(n = 1; n <= donguDegeri; n++){
cocukDec[n] = diziBirlestir(parent1[n], parent2[n], nokta);
}
// Tekrarlı soruları düzeltme
duDec = cocukDec;
x = arrayLen(cocukDec);
y = arrayLen(cocukDec[1]);
for(i = 1; i <= x; i++){
for(j = 1; j <= y; j++){
for(k = i + 1; k <= y; k++){
if(cocukDec[i][j] == cocukDec[i][k]){
do{
qNo = randomRangeSayi(1, veriBoyutu);
} while(arrayFind(cocukDec[i], qNo) > 0);
duDec[i][k] = qNo;
}
}
}
}
return duDec;
}
// Gelen dizinin ilgili parçasını geri döndürür.
public array function diziParcasi(required array dizi, required numeric baslangic,
required numeric bitis){
diziParca = arrayNew(1);
for(i = 1; i <= bitis - baslangic; i++){
diziParca[i] = dizi[i + baslangic - 1];
}
return diziParca;
}

// Gelen iki diziyi noktadan birleştirir.
public array function diziBirlestir(required array dizi1, required array dizi2, required
numeric nokta){
n = arrayLen(dizi1);
diziBirlesik = sifirlarDizisi(n, 1);
for(i = 1; i <= nokta; i++){
diziBirlesik[i] = dizi1[i];
}
for(i = nokta + 1; i <= n; i++){
diziBirlesik[i] = dizi2[i];
}
return diziBirlesik;
}
// Ebevyinleri üreten fonksiyon

```

```

public array function parentFunction(required array populasyonDecision, required
array selected, required numeric populasyonSayisi){
ebeveyn = ArrayNew(2);
donguDegeri = populasyonSayisi/2;
for(i = 1; i <= donguDegeri; i++){
ebeveyn[1][i] = populasyonDecision[selected[i]];
ebeveyn[2][i] = populasyonDecision[selected[donguDegeri + i]];
}
return ebeveyn;
}

```

// Amaç fonksiyonu için gerekli maliyet dizisini döndürür.

```

public array function objectiveFunction(required array nufus, required query
veriSeti, required numeric zorlukDerecesi, required numeric celdiriDerecesi){
x = ArrayLen(nufus);
y = ArrayLen(nufus[1]);
zorluk = sifirlarDizisi(x, 1);
frekans = sifirlarDizisi(x, 1);
celdirici = sifirlarDizisi(x, 1);
maliyet = sifirlarDizisi(x, 1);
maliyet1 = sifirlarDizisi(x, 1);
maliyet2 = sifirlarDizisi(x, 1);
maliyet3 = sifirlarDizisi(x, 1);
for(i = 1; i <= x; i++){
for(j = 1; j <= y; j++){
zorluk[i] = zorluk[i] + Arguments.veriSeti.Zorluk[Arguments.nufus[i][j]];
maliyet1[i] = abs((Arguments.zorlukDerecesi * y) - zorluk[i]);

frekans[i] = frekans[i] + Arguments.veriSeti.Frekans[Arguments.nufus[i][j]];
maliyet2[i] = frekans[i];

celdirici[i] = celdirici[i] + Arguments.veriSeti.Celdiri[Arguments.nufus[i][j]];
maliyet3[i] = abs((Arguments.celdiriDerecesi * y) - celdirici[i]);

maliyet[i] = maliyet1[i] + maliyet2[i] + maliyet3[i];
}
}
return maliyet;
}

```

// İlk nüfusu oluşturan fonksiyon. Parametre aldığı diziyi karıştırarak rastgele sorular alıyor. NxM boyutlu bir dizi döndürüyor.

```

public array function initPopulation(required numeric initPopulation, required
numeric soruSayisi, required numeric populasyonSayisi, required array siraDizisi){
populasyon = arrayNew(2);
for(i = 1; i <= initPopulation; i++){
CreateObject("java", "java.util.Collections").Shuffle(Arguments.siraDizisi);
//Dizinin eleman sıralarını rastgele sıraladık.

```

```

for(j = 1; j <= soruSayisi; j++){
populasyon[i][j] = Arguments.siraDizisi[j]; //Karıştırılan sizin ilk n elemanını diğer
diziye atıp geri döndürüyoruz.
}
}
return populasyon;
}

```

```

// fitness değerlerinin olduğu diziyi döndürür.
public array function fitnessFunction(required array maliyet, required numeric
populasyonSayisi){
maliyetSiralı = diziSiraBul(maliyet);
mSiralı = diziSiraBul(maliyetSiralı[1]);
fitness = ArrayNew(1);
for(i = 1; i <= ArrayLen(maliyetSiralı[2]); i++){
fitness[i] = (((Arguments.populasyonSayisi - mSiralı[2][i]) /
(Arguments.populasyonSayisi - 1)) + epsilon());
}
return fitness;
}

```

```

// rulet çarkı için seçim yapan fonksiyon
public array function selectFunction(required array fitness, required numeric
populasyonSayisi){
secim = arrayNew(1);
rulet = arrayNew(1);
n = arraySum(fitness);
//rulet çarkı için değerleri üretiyor.
for(i = 1; i <= arrayLen(fitness); i++){
rulet[i] = randomSayi() * n;//fitness[i] * epsilon();
}
}

```

```

for(i = 1; i <= populasyonSayisi; i++){
dilim = 0;
j = 1;
while(dilim lt rulet[i]){
dilim += fitness[j];
j++;
}
// Ruletten seçim değeri atıyor.
secim[i] = j;
}
return secim;
}

```

```

// Sıfırlardan oluşan bir NxM boyutlu dizi döndürür.
public array function sıfırlarDizisi(required numeric n, required numeric m){
if(m > 1){

```

```

sifirlar = arrayNew(2);
for(i = 1; i <= n; i++){
arraySet(sifirlar[i], 1, m, 0);
}
}
else{
sifirlar = arrayNew(1);
arraySet(sifirlar, 1, n, 0);
}
return sifirlar;
}
// Dizinin en büyük elemanını ve indisini dizi olarak döndürür.
public array function maxElemanBul(required array dizi){
maxSonuc = arrayNew(1);
maxSonuc[1] = arrayMax(dizi);
maxSonuc[2] = arrayFind(dizi, maxSonuc[1]);
return maxSonuc;
}
// Dizinin en büyük elemanını ve indisini dizi olarak döndürür.
public array function minElemanBul(required array dizi){
maxSonuc = arrayNew(1);
maxSonuc[1] = arrayMin(dizi);
maxSonuc[2] = arrayFind(dizi, maxSonuc[1]);
return maxSonuc;
}
// Epsilon sayısı üretir. Epsilon
public numeric function epsilon(){
return rand()/10000;
}
// 0-1 arasında rastgele sayı üretir.
public numeric function randomSayi(){
return rand();
}
// Rastgele sayı üretir. Aralıkta
public numeric function randomRangeSayi(required numeric min, required numeric
max){
return randRange(min, max);
}
// CF tarafından üretilmiş bir JSON veriyi QUERY olarak döndürür.
public query function CFJSON2Query(required string JSONVeri)
{
if(isJSON(JSONVeri))
{
JSONCoz = DeSerializeJSON(JSONVeri);
Qry = QueryNew(ArrayToList(JSONCoz.Columns));
t = 1;
for(i in JSONCoz.Data)
{

```

```

k = 1;
QueryAddRow(Qry);
for(j in i)
{
QuerySetCell(Qry, JSONCoz.Columns[k++], j);
}
t++;
}
}
return Qry;
}
// CF tarafından üretilmiş bir Array veriyi QUERY olarak döndürür.
public query function CFArray2Query(required array veri)
{
if(IsArray(veri))
{
columnList = "Soruid,Dersid,Frekans,Zorluk,Celdiri";
Qry = QueryNew(columnList);
for(i = 1; i <= ArrayLen(veri); i++)
{
QueryAddRow(Qry);
for(j = 1; j <= ArrayLen(veri[i]); j++)
{
QuerySetCell(Qry, ListGetAt(ColumnList, j), veri[i][j]);
}
}
}
return Qry;
}
// 1'den N`e kadar olan sayıların olduğu dizi döndürür.
public array function siraDiziOlustur(required numeric n)
{
arr = ArrayNew(1);
for(i = 1; i <= n; i++)
{
arr[i] = i;
}
return arr;
}
// N elemanlı bir diziyi sıralayıp sırasız dizideki yerlerini geri döndürür.
public array function diziSiraBul(required array sirasiz)
{
arr = ArrayNew(2);
arr[1] = sirasiz;
arr[2] = sifirlarDizisi(ArrayLen(sirasiz), 1);
n = CreateObject("java", "java.util.Collections");
n.sort(arr[1]);
for(i = 1; i <= arrayLen(sirasiz); i++)

```

```
{
if(i neq 1 and arr[1][i] eq arr[1][i-1])
{
x = arr[2][i-1] + 1;
}
else
{
x = 1;
}
for(j = x; j <= arrayLen(sirasiz); j++)
{
if(sirasiz[j] == arr[1][i])
{
arr[2][i] = j;
break;
}
}
}
return arr;
}
</cfscript>
</cfcomponent>
```


KİŞİSEL YAYIN VE ESERLER

Altıntaş L., Alimođlu M. K., Alvur T. M., Yıldız G., Diri S., Ulusal ekirdek Eđitim Programı'nın Tıp Eđitimi Programına Entegrasyonunda Yazılım Destekli Uygulama rneđi; Kocaeli niversitesi Tıp Fakóltesi Deneyimi, *Tıp Eđitimi Dnyası*, 2012, **34**, 6-12.

Diri S., đc S., Yıldırım M., Genetic Algorithm and Fuzzy Logic Based Auto-Test System For Distance Education, *ICENS 2018*, Kiev, Ukrayna, 2-6 May 2018.

đc S., Diri S., Yıldırım M., Text Preprocessing for an Experiment Code On The Internet Assisted Laboratory System, *ICENS 2018*, Kiev, Ukrayna, 2-6 May 2018.

ÖZGEÇMİŞ

1988 yılında Bursa’da doğdu. İlk, orta ve lise öğrenimini Bursa’da tamamladı. 2006 yılında girdiği Kocaeli Üniversitesi Teknik Eğitim Fakültesi Bilgisayar Öğretmenliği Bölümü’nden 2011 yılında mezun oldu. 2011 yılında başladığı Kocaeli Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı’ndaki Yüksek Lisans eğitimine devam etmektedir. 2012-2017 yılları arasında Kocaeli Üniversitesi’nde Yazılım Geliştirici olarak çalıştı. 2017 yılından itibaren Kocaeli Üniversitesi’nde Öğretim Görevlisi olarak çalışmaktadır. Evli ve bir kız çocuk babasıdır.

