T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KRİPTOANALİZ PROBLEMLERİNİN ÇÖZÜMÜNDE EVRİM STRATEJİSİ UYGULAMASI

YÜKSEK LİSANS TEZİ

Abdihalim Adam ABDIRAHMAN

Enstitü Anabilim	**:**	**BİLGİSAYAR VE BİLİŞ.MÜH**
Dalı Tez Danışmanı	**:**	**Yrd.Doç.Dr. Murat İSKEFİYELİ**

Ocak - 2018

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KRİPTOANALİZ PROBLEMLERİNİN ÇÖZÜMÜNDE EVRİM STRATEJİSİ UYGULAMASI

YÜKSEK LİSANS TEZİ

Abdihalim Adam ABDIRAHMAN

Enstitü Anabilim Dalı : BİLGİSAYAR VE BİLİŞİM MÜH

Bu tez 24.01.2018 tarihinde aşağıdaki jüri tarafından oybirliği ile kabul edilmiştir.

Prof.Dr.	Prof.Dr.	Yrd.Doç.Dr.
Resul KARA	Cemil ÖZ	Murat İSKEFİYELİ
Jüri Başkanı	Üye	Üye

BEYAN

Tez içindeki tüm verilerin akademik kurallar çerçevesinde tarafımdan elde edildiğini, görsel ve yazılı tüm bilgi ve sonuçların akademik ve etik kurallara uygun şekilde sunulduğunu, kullanılan verilerde herhangi bir tahrifat yapılmadığını, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, tezde yer alan verilerin bu üniversite veya başka bir üniversitede herhangi bir tez çalışmasında kullanılmadığını beyan ederim.

Abdihalim Adam ABDIRAHMAN

12.12.2017

TEŞEKKÜR

Yüksek lisans eğitimim boyunca değerli bilgi ve deneyimlerinden yararlandığım, her konuda bilgi ve desteğini almaktan çekinmediğim, araştırmanın planlanmasından yazılmasına kadar tüm aşamalarında bana danışmanlık ederek bilgi ve tecrübesiyle beni yönlendiren ve aydınlatan değerli hocam Sayın Yrd.Doç.Dr.Murat İSKEFİYELİ teşekkürlerimi sunarım.

Yüksek lisansımı tamamlamamda bana yardımcı oldukları için Türkiye Bursları kurumuna teşekkür ediyorum. Sakarya Üniversitesi ve özellikle Bilgisayar ve Bilişim Bilimleri Fakültesi'nin değerli hocalarına katkılarından dolayı teşekkür ederim.

Laboratuvar olanakları konusunda anlayış ve yardımlarını esirgemeyen arkadaşım Hussein El-Sanabani'ye teşekkür ederim.

Ayrıca eğitim hayatım boyunca maddi ve manevi desteklerini esirgemeyen çok değerli annem, babam, kardeşlerim ve değerli eşime sonsuz minnet ve teşekkürlerimi sunarım.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR LİSTESİ

EC	:Evrimsel hesaplama (Evolutionary computation)
EA	: Evrimsel algoritmalar (Evolutionary algorithms)
ES	:Evrim Stratejisi (Evolution strategy)
FF	:Uygunluk fonksiyonu (Fitness function)
GA	:Genetik Algoritmalar (Genetic algorithms)
LFSR	:Doğrusal geri beslemeli kaydırma yazmacı (Linear feedback shift register)
ROT-13	:On üç yeri döndür (Rotate 13 places)

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar kelimeler: metasezgisel, Evrim stratejisi, uyugunluk fonksiyonu, bigram, trigram, karılmalı kripto, şifreleme, şifre çözmek, kriptoloji, kriptoanalizi, kriptografi.

Bu çalışmada, Kriptografi ve Kriptanaliz üzerinde duruldu ve çeşitli algoritmalar uygulandı. Optimizasyon, geometri ve diferansiyel hesabın iyileştirilmesini uygulayan en yaşlı matematiksel tiplerden biridir, fakat şimdi problemleri çözmek için bilimsel ve mühendislik tarafından uygulanan en iyi algoritmalardan biridir. Günümüzde, optimum özümü bulmak için optimizasyon problemi uygun şekilde tanımlanmalıdır, daha sonra optimum özüm ilgili Matematiksel yaklaşım kullanılarak bulunabilir. Stokastik algoritmalar için, Sezgisel ve Meta-heuristik olmak üzere iki tanıtıcı optimizasyon tekniği türü vardır. En yaygın kullanılan Meta-heuristics, Sürü zekası (swarm intelligence) ve evrimsel algoritmalar'dır (evolutionary algorithms).

Bu araştırma, üç farklı algoritma uyguladı: Evrim Stratejisi, uygunluk Fonksiyonu ve Sütun Transferi Şifreleri. Aynı şekilde, araştırmanın temel amacı, Evrim Stratejisini uygulayarak Sütun Transferi Şifrelemelerini deşifre etmektir. Ayrıca, çalışma, Kriptografi (Sütun Transferi Şifreleri) ve Kriptanaliz (uygunluk fonksiyonu ile Evrim stratejisi) içeren bir uygulama uyguladı; uygulama şifreleme (Kriptografi) düz metni ve son olarak şifre metnini deşifre ederek (Kriptanaliz) kullandı. Sütunlu Taşıma Şifreleri, düz metinleri 13 anahtar uzunluğuyla şifrelemektedir. Evrim Stratejisi Algoritması anahtarı ve düz metni bulmaya ve en sonunda şifre metnini hızlı bir şekilde deşifre etmeye çalışır; böylece ES, şifreli metnin bigramlarını (iki harfli) ve trigramları (üç harfli) toplamak için geliştirilmiş fitness fonksiyonunu kullanır.

Bulgulara göre, çalışma, tüm şifre metinlerini, projede açıklandığı üzere, daha önce uygulanan diğer algoritmalara kıyasla daha etkili ve daha kısa bir sürede deşifre etti.

APPLYING EVOLUTION STRATEGY TO SOLVE CRYPTANALAYSIS PROBLEMS

SUMMARY

Keywords: Meta-heuristic, Evolution strategy, Fitness function, bigrams, trigrams, transposition cipher, encryption, Decryption, Cryptology, Cryptanalysis, Cryptography.

In this study, the Cryptography and Cryptanalysis are focused on them and applied various algorithms. Optimization is one of the most aged mathematical types which applied the improvement of the geometry and differential calculus, but now it's one of the best algorithms applied by scientific and engineering to solve problems. At present, to find the optimum solution the optimization problem has to be defined suitably, then the optimum solution can be found using relevant Mathematical approach. For stochastic algorithms, there are two familiar types of optimization techniques, Heuristic and Meta-heuristic. the most widely used Meta-heuristics are Swarm Intelligence and Evolutionary Algorithms.

This research applied three different algorithms, Evolution Strategy combined with Fitness Function and Columnar Transposition Ciphers. Likewise, the main objective of the study is to decipher the Columnar Transposition Ciphers by applying Evolution Strategy. Moreover, the study applied an application which contain Cryptography (Columnar Transposition Ciphers) and Cryptanalysis (Evolution Stratgey with fitness function), the application is encrypting (Cryptography) the plain text and finally deciphering (Cryptanalysis) the cipher text. The Columnar Transposition Ciphers is encrypting the plain text using with 13 key length. The Evolution Strategy Algorithm attempts to find the key and the plain text and finally deciphering the cipher text quickly, so ES uses improved fitned fitness function to gather the bigrams (two letters) and trigrams (three letters) of cipher text.

According to the findings the study deciphered all the cipher texts effectively and less time than all the other algorithms previously applied, as explained insdie the project.

BÖLÜM 1. GİRİŞ

Modern dünyada hassas bilgiler, gayri meşru kullanıcılardan veya saldırganlardan yetkisiz bir şekilde erişilmektedir ve bu, tüm kurumların ona karşı savunan ilk meydan okumadır.

Ağ, saldırganların ağlara sıkça saldırdığı için verilerin hassas olmasını sağlar. Aynı şekilde, verileri tehdit etmek için önlenebilen ve burada kriptolojinin bilgiyi güvence altına alması için birçok çeşitli yol vardır.

Yukarıdaki cümleler ortaya çıktığı için, kriptoloji bilgi güvenliğinin temelini oluşturmaktadır; bu alan da temel olarak matematik ile ilgilidir [1].

Kriptografi, mevcut güvenlik protokollerini kontrol etmek için araçlar sağlar. Bununla birlikte, düzgün yapması zor olmasına rağmen, dünyanın dört bir yanındaki dağıtım sistemlerinin çoğunu savunmaktadır [2].

Kriptoloji, kriptosistemlerin çalışması olarak tanımlanabilir ve kriptanaliz ve kriptografi olmak üzere iki kategoriye ayrılabilir.

Kriptografi de kriptosistemlerin incelenmesini içerirken, kriptoanalizi kriptografik yöntemleri çiğneme çalışmalarıdır.

Bu iki sahne neredeyse birbirine yakın, kriptosistemlerin güvenlik ayarlaması gerektiğinde, gelişmelerinde önemli rol oynadığı için analiz edilmelidir [3].

Ayrıca, Kriptografi, bilgi işlem sistemleri hakkındaki bilgileri savunmak için yararlı olan önemli bir mekanizmadır.

Buna karşılık veriyi korumak için kullanılan teknikler şifreleme alanına aittir.Kelimenin tam anlamıyla, bu alanla ilgili endişeleri olan üç kelime vardır: Değişkenlik kullanılabilir Kriptoloji, Kriptografi ve Kriptanaliz. Öte yandan, düz metinleri şifrelemek için kullanılan algoritmalar olan "Cryptosystems" araştırmasında defalarca kullanılabilecek başka bir terim vardır; düz metin ve anahtar alır ve çıktıları Şifre metni olacaktır.

Kriptoanalizi, çeşitli teknikler kullanarak kriptosistemleri parçalamak için kullanılan yöntemdir, saldırganlar daima düz metin anahtarını elde etmek için Kaba kuvvet saldırısı kullanmaktadır, ancak ihlali zor olan birçok Kriptosistem bulunmaktadır.

Bu modern dünyada teknoloji daha hızlı büyüyor, bu nedenle zorlukları ve çözümleri rekabet etmekle birlikte, bu zorlu Kriptosistemleri parçalayabilen yeni teknolojiler Meta-heuristic Algoritmalar olarak ortaya çıkıyor.

Günümüzde, optimum çözümü bulmak için, optimizasyon sorununun uygun bir şekilde saptanması gerekir, daha sonra optimum özüm ilgili Matematiksel yaklaşım kullanılarak bulunabilir.

Stokastik algoritmalar için, Sezgisel (Heuristic) ve Meta-heuristik olmak üzere iki tanıtıcı optimizasyon tekniği türü vardır. Sezgisel, zorlu bir optimizasyon sorununun iyi bir çözümünü elde etmek anlamına gelir, ancak en doğru çözümü bulmak garanti değildir, bu nedenle yerel optimum çözümü getirebilir.

Benzer şekilde, yerel en iyi çözümü engellemek için yapılan iyileştirmeler ve burada Meta-heuristics oynamaya başladı; böylece Meta-heuristics, global optimum çözümü getirmeye çalışan önceki buluşsal yöntemler geliştirildi.

Dolayısıyla, en yaygın kullanılan Meta-heuristics, Swarm Intelligence ve Evrim Algoritmalarıdır. Bu araştırma Evrim Algoritmalarından kaynaklanan Evrim stratejilerini bir sonraki oturumlarda açıklanacağı gibi uygulamıştır.

Algoritma, şifre metninin harflerini toplamak için iyileştirilmiş uygunluk fonksiyonuyla (fitness Function) birlikte uygulandı.

Bu çalışmada kriptografi ve şifreleme analizi ile birlikte iki şey var, şifreleme, Şekil 1'de gösterildiği gibi, transpozisyon şifresini (düz metin, şifre metni ve şifrelemek için kullanılan anahtarı) içeriyor; kriptografi, Uygunluk fonksiyonuyla birleştirilen evrim stratejisini, bu farklı fikirleri şifreli düz metinleri Şekil 2.2.'de gösterildiği gibi kıracaktır.

1.1. Geçmiş

Daha önce de belirtildiği gibi araştırma, Evrim Algoritması'nı kullanarak şifreli şifre metnini kırmak üzeredir, bu nedenle bu algoritma Evrim Algoritmalarından türetilmiştir; bu bölüm bu araştırmanın arka planına kısaca değinecek ve bazı terminolojileri araştırmanın arka planına girmeden önce Evrimsel Hesaplama, Evrim Stratejisi, uygunluk Fonksiyonu ve Klasik Şifreler (Değiştirme Şifresi) gibi Meta Heuristic Algoritmalar, tezin temel bloğu olduğu için anlaşılmalıdır.

Sezgisel Yöntemler (Heuristic Method): Sezgisel yöntem, uygun olmak için biraz makul derecede yakın olan doyurucu bir sonucu keşfetmek için kullanılan bir uygulamadır. İyi tasarlanmış sezgisel teknik her zaman neredeyse en uygun veya herhangi bir çözümün bulunamadığı bir çözüm verebilir. Sezgisel tekniğin dezavantajı çoğunlukla farklı uygulamalar yerine özel bir probleme uyacak şekilde tasarlanmıştır [8].

Metaheuristics: Metaheuristic, genel bir tasarım stratejisi talimatlarını belirli bir sezgisel yöntem türünde iyileştirmeye yönelik genel bir çözüm yöntemidir [8]. Benzer şekilde, Meta Heuristic Algoritmalar yalnızca çözümlerin araştırmacıya yardımcı olması için nadir bulunan ve çok az olduğunda kullanılır; bu nedenle araştırmacı, en iyi çözümün ne olacağını bilmemekte ve bulguları elde etmek için düzenli bir yol bulunmadığında, çok az sezgisel devam etmek [9].

Ayrıca, Bilgisayar Bilimlerinde kullanılan popülasyon tabanlı yöntemler kavramı daima Biyolojiden alınmıştır, özellikle Evrimsel Hesaplama (EC) tekniği, EC'nin birlikte çeşitli algoritmaları içerdiğinden, Biyoloji'den ödünç alır. Bu koleksiyondan alınan herhangi bir Algoritma, Evrimsel Programlama olarak adlandırılır. Çoğu EA, yineleme, Genetik Algoritma ve Evrim Stratejisi [9] başına bir kez tümü yeniden oluşturan nesiller algoritmalarına ayrılmıştır.

Bu çalışmada, Evrim Stratejisi, daha önce Kriptografinin Klasik Şifreleri ile şifrelenmiş şifre metninin şifresini çözmek için uygulanacaktır.

Evrim Stratejisi, Ingo Rechenberg ve Berlin Teknik Üniversitesi'nden Hans-Paul Schwefel tarafından 1960'ların ortalarında geliştirilen algoritma ailesini içeriyor.ES, Kesme Seçimi adlı kişileri seçmek için kolay bir yaklaşım uygular ve her zaman Mutasyon'u çimdik operatörü olarak kullanır [10,11].

Genellikle Evrimsel Stratejiler olarak adlandırılan Evrim Stratejileri [11, 12, 13, 14] ve Evrimsel programlama [15], biyolojik evrim ilkelerinin motive ettiği arama kalıplarıdır. Evrim Algoritmaları ailesinin (EA) devlet optimizasyon problemleri ve Evrim Stratejisi bunlardan biridir, EA daima küçük farklılıklar ile tekrarlanan süreci uygular ve onu seçer ve her nesil (iterasyon), aday çözüm (yeni yavrular) ebeveynlerinden, fitness (kalite) değerlendirilir ve daha iyi adaylar (yavrular) ebeveynlerinin yerine geçmek üzere seçilir.

Fitness fonksiyonu: Zindelik değerlendirme modunun tasarımı ağırlık modu, Toemeh'in modunda bigramların ve trigramların alanlarında yalnızca sınırlı olan, transpozisyon şifrelemelerine saldırmada çok önemli bir faktördür [16]. Bu araştırmada, uyum işlevinin şifreleme özelliğini kolumnar transpozisyon şifrelerini kırmak için fitness fonksiyonu uygulanmış ve şifrelenmiş kelimeler bigramlar (iki kelime) ve trigramlar (üç kelime) olarak düzenlenmiştir.

Geçiş şifreleri, tüm düz metinlerin kendiliğinden değiştirilmesiyle değiştirilen klasik şifrelemelerden biridir. Metin uzunsa, her cümle bağımsız olarak kendiliğinden bir permütasyonla değiştirilir.

Ayrıca, araştırma en yaygın olarak araştırılmış klasik şifreler olan Sütun Transkripsiyon Şifrelemelerini kullandı. Columnar Transposition'da anahtar ve düz metin yatay olarak sabit satırlarla yazılır ve dikey olarak okunur. Satırların genişliği ve anahtar kelime tarafından belirlenen sütunun permütasyonu. Örneğin, UNIVERSITY anahtar kelimesi 10 karakter içeriyorsa, satırlar 10 olur. Permütasyon, anahtar kelimenin alfabetik (A dan Z ye) sırasına göre Tablo 2.1.'de tanımlandığı şekilde düzenlenir [17].

Son olarak, araştırma Sütun Transkripsiyonu Şifrelemesi kullanılarak şifrelenmiş şifre metnini çiğneyecek, daha sonra deşifreleme süreci, Şekil 2.2.'de gösterildiği gibi düz metin şifrelemek için kullanılan anahtarı da iyileştirme fonksiyonuyla birleştirilen Evrim Stratejisi olacak.

1.2. Problem İfadesi

Kriptoanalizi, düz metinleri şifrelemek için kullanılan gerçek anahtarı almadan bir şifre metninin şifresini çözmek için kullanılan bir tekniktir. Her zaman, anahtarın daha az veya daha fazlasını tahmin etmek ve daha sonra düz metni kurtarmak için farklı yaklaşımlar kullanarak şifre metnine saldırır. Ayrıca, Cryptanalysis'i kullanan saldırganlar, şifre metnini şifresini çözmek için gizli anahtarı bulmaya çalışırlar; daha önce kullanılanlarla aynı olan başka bir algoritma almaya çalışıyor veya şifre metnini kırmak için bilgi edinmeye çalışıyorlar.

Aynı şekilde, anahtarları kurtarmak veya şifre metnini kırmak, herkesin kolaylıkla başaramayacağı gibi, henüz kırpılmamış bazı algoritmalar da mevcut. Bu nedenle, saldırganların çoğu, şifrelenmiş içeriğe erişmek için çeşitli yöntemler uygulamaya, gerçek tuşu alıncaya kadar birkaç tuş kullanır ve saldırgan bazı basit metinleri biliyorsa çok kolay. Her ne kadar Caeser Ciphers gibi anahtarın geri alınması için

daha az zaman harcayacak Vigener ve LFSR (Doğrusal Geri Bildirim Kaydırma Kaydı) gibi düzgün kripto sistemleri mevcut olsa da. Transpozisyon Şifresinin Çözülmesinin anahtarı elde etmek de zaman alabilir ve çaba gösterebilir, ancak daha önce açıklanan Sütun Transferi Şifresinin çözülmesi muhtemelen zordur.

Aynı şekilde, anahtarları kurtarmak ya da şifre metnini kırmak, herkesin kolaylıkla başaramayacağı gibi, henüz kırılmayan bazı algoritmalar da mevcut. Dolayısıyla, saldırganların çoğu, şifrelenmiş içeriğe erişmek için çeşitli yöntemler uygulamaya, gerçek tuşu alıncaya kadar birkaç tuş kullanır ve saldırgan bazı basit metinleri biliyorsa çok kolay. Caesar Ciphers gibi anahtarı geri almak için daha az zaman harcayacak Vigenere ve LFSR (Doğrusal Geri Bildirim Kaydırma Kaydı) gibi pürüzsüz kripto sistemleri mevcut olmasına rağmen. Transpozisyon Şifresinin Çözülmesinin anahtarı elde etmek de zaman alabilir ve çaba gösterebilir, ancak daha önce açıklanan Sütun Transferi Şifresinin muhtemelen deşifre edilmesi zordur.

Buna ek olarak, Sütunlu değiştirme şifreleri kaba kuvvet saldırıları kullanılarak ele geçirilebilir, ancak alınmış olan düz metin önceden şifrelenmiş olanla özdeşleşmeyebilir. Yukarıda bahsedildiği gibi, meta-heuristik olan sütunsal transpozisyon şifresini hızla deşifre edebilen yeni ortaya çıkan teknolojilerdir. Sonuç olarak, araştırmada kullanılacak Meta-heuristics, Evrimsel Hesaplamadan türetilen Evrim Stratejisi'dir. Evrim Stratejisi harika Meta- şifreleme metnini kırabilen ve Şekil 2.2.'de gösterildiği gibi düz metni şifrelemek için kullanılan anahtara erişebilen sezgisel bir tekniktir. Ayrıca, Evrim Stratejisi ve Fitnes İşlevi, bigramların (iki sözcük) ve trigramların (üç sözcük) yeniden düzenlenmesiyle eşzamanlı olarak uygulanacaktır).

Sonuç olarak, Evrim Stratejisi, düz metni ve anahtarı iyice çözen mükemmel bir tekniktir, böylece şifrelenmiş düz metine eşit küresel optimum çözümü getirir.

1.3. Araştırma Kapsamı

Bu çalışmada, araştırmanın üzerinde durulacak iki ana bölüm vardır: Bunlar Kriptografi ve Kriptanaliz, özellikle Sütunlu Transpozsiyon Şifreleme ve Evrim Stratejileri'dir. Kriptografi Sütunlu Transpozisyon Şifrelemesi içerir, düz metin, anahtar ve şifre metnini oluşturur, anahtar şifre metnini üreten anahtartır.

Öte yandan, ikinci sektör, Kriptanalizdir ve Evrim Stratejisini, sonuç sekmesini, grafikleri (yakınsama ve uyum), düz metin ve anahtarı kapsar. Son olarak, araştırma, Evrim Algoritmalarını (Evrim Stratejileri hariç) ve sütunsal transpozisyon şifreler haricindeki Kriptosistemleri kapsamaz.

1.4. Araştırma Özeti

Çalışma beş ana bölümden oluşuyor; bu bölüm, okuyucular için derin anlamlar sergiliyor; ilk önce, Giriş, okuyucuya çalışma hakkında ince bir anlayış kazandırdığından, en önemlisi, ikinci olarak, Edebiyat incelemesi, araştırmanın tüm ilgili bilgilerini inceliyor, dolayısıyla okuyucu firma hakkındaki verilere fayda sağlayacaktır.

Daha sonra, üçüncü bölüm, çalışmanın iki ana bileşeni olan Kriptoloji ve Meta-heuristik Algoritmalar ile ilgilidir. Bundan sonra, araştırmanın tüm çıktılarını ortaya koyduğu için araştırma ve analiz en önemli kısmıdır. Son olarak, sonuç ve ilerideki tartışmalar, bu bölüm bu tezin sonuçlarını ve çalışmasını sonuçlandıracaktır.

BÖLÜM 2. KRİPTOLOJİ VE METAHEURISTIC ALGORİTMALAR

2.1. Giriş

Daha önce tanıtıldığı gibi, araştırma Düz metinleri Sütun Transkripsiyon Şifrelemeleri kullanarak şifrelemek ve ardından şifre metnini Evrim Strateji Algoritması'nı kullanarak deşifre etmek üzerine kurulmuştur. Bu bölüm tüm ilgili parçaları ayrı ayrı açıkladı.

Araştırmanın bu kısmı, mevcut sonuçlara ve analizlere kıyasla, en önemlisi olan işi açıklamaktadır; çünkü bu araştırma, çeşitli meta-sezgisel algoritmalar uygulayarak deşifre edilen transkripsiyon şifrelerini inceleyen muazzam araştırmacılar olan sütunsal transpozisyon şifrelemelerini çözmekte Anahtar teknikleri tahmin etmek.

Dahası, Transpozisyon şifre metninin şifresini çözmek için uygulanan Meta-Heristik Algoritmalar, Evrimsel Hesaplamalar, özellikle Genetik Algoritmaları, Benzetimli Tavlama Algoritması, Güldürü Arama Algoritmaları ve ayrıca sütunları tahmin etmek üzere olan Sütun Transferi Şifrelemelerini deşifre edebilen başka bir yöntemdir. Şifrelerin satırları.

Ayrıca, yukarıda belirtilen bazı konular önümüzdeki numaralandırılmış konulara derinden verilmekte ve bu çalışmanın sonuçlarına kıyaslanmaktadır.

2.2. Transpozisyon Şifresi

Transpozisyon şifreleri düz metinleri (mesaj) saldırganı yanıltmaya yönelik bir yöntemle karıştırır, ancak alıcı kolayca yeniden düzenleyebilir. Transpozisyon şifreleri klasik şifrelere aittir, halen modern şifrelere uygundur.

Düz metni şifrelemek için kullanılan anahtar, şifre metnini yeniden düzenlemek için yeterli bilgi verir.

Transpozisyon Şifrelemeleri alt bölümlere sahiptir ancak bu çalışma, araştırmacıların çoğunlukla çeşitli teknikler kullanarak şifre çözmek için kullandıkları Sütun Transferi üzerine vurgu yapmaktadır, ancak bu çalışma Sütun Aktarımı Şifresini uygulayacaktır.

2.2.1. Sütunlu transpozisyon şifresi

Bu Transpozisyon tipleri herhangi bir anahtar kelimeye sahip değil, sadece harfleri karıştırıyor, örneğin bu Sade Metin LOOKTHESTAR'ı Sütunlu Hareketli Şifreleyicileri kullanarak şifrelemek. Öncelikle, şifreleri almak için düz metinleri sıralara yerleştirin ve dikey olarak okuyun (sütunlar).

L O O K
T H E S
T A R X

Karakterler 11 olduğu için bir harf eksik ve boş X konumu alınırsa, şifre metni LTTOHAOERKSX olur. Aynı şekilde, Alıcı, Transpozisyon Şifresinin sütunlarının sayısı biliniyorsa, düz metinleri kurtarabilir ancak saldırganlar tüm bölenleri test ederek sütun sayısını bulmaya çalışır, örneğin karakterler 12, dolayısıyla 2,3,4,6 ise bölenleri 12 olduğunda, saldırgan olası tüm sütunları oluşturuyor ve nihayetinde düz metin elde ediyor. Bununla birlikte, anahtarsız sütun transpozisyonunun bu tipleri

kelimenin tam anlamıyla çözülmesi kolaydır, ancak bazıları tahmin etmek ve düz metin almak [23] için zorlanıyor.

2.2.2. Anahtarlı sütun transferi şifresi

Sütunlu Taşınması, Şifre metninin sırasını belirleyen bir anahtar eklenerek zorlaştırılabilir, bu tür Geçiş şifresi anahtarları uygulayan Anahtar Kelime Sütun Transferi olarak anılır.

Örneğin, bu düz metin WENEEDTOUSETHISMETHODTOENCRYPT, Tablo 2.1.'de gösterildiği gibi UNIVERSITY'i şifrelemek için aşağıdaki anahtara sahiptir; düz metin ve anahtar normal olarak tabloda bulunan satırlara, anahtara göre aphanelik sıraya göre dizilmiştir ve ilk sütun okundu şifreler sonra sıradaki sütunlar karmakarışıktır [23].

Tablo 2.1. Düz metinleri sütunlu aktarım şifreleyicileri kullanarak şifrelemek.

Pozisyon	8	4	2	9	1	5	6	3	7	10
Anahtar	U	N	I	V	E	R	S	I	T	Y
Açık Metin	W	E	N	E	E	D	T	O	U	S
	E	T	H	I	S	M	E	T	H	O
	D	T	O	D	E	C	R	Y	P	T

Açık metin = WENEEDTOUSETHISMETHODTOENCRYPT
Şifreli metin = ESNNHOOTYETTDMCTERUHPWEDEIESOT

Araştırma, harfleri bir araya getirerek ve iyileştirilmiş fitness işleviyle birleştirilmiş Evrim Stratejisini kullanarak bu türden Geçiş Şifrelemelerini çiğneyecek.

2.3. Evrim Stratejisi Algoritması

1960'lı yıllarda Rechenberg Evrim Stratejisini test etti [24, 25] ve Schwefel [26] derin araştırmalar yaptı. Giriş bölümünde daha önce açıklandığı gibi Evrim Stratejisi Evrim Algoritmalarına dayanmaktadır.

Evrim Stratejisi beş farklı faktörü içerir: Başlatma, rekombinasyon, mutasyon, değerlendirme ve seçim. Buna ek olarak, yukarıda adı geçen bileşenlerin her biri aşağıda tanımlandığı gibi kendi tanımına sahiptir [27].

Başlatma: Rassal olarak oluşturulan ulaşılabilir çözüm sayısı.

Rekombinasyon: İki kromozomun, genetik özelliklerini, tüm özelliklerine sahip yeni yavrular üretmek için birleştirdiği bir süreç var mı, iki ebeveyn yeni bir çocuk doğurduğunda da açıklanabilir.

Mutasyon: genetik farklılıkları bir nesilden diğerine sürdürmek için kullanılan süreçtir.

Nesnel mutasyon, algoritmanın, kromozom popülasyonunu birbirine benzemek üzere durdurarak, yerel minimumdan kaçınmasına izin vermektir.

Değerlendirme: Fitness fonksiyonu, bireyin genetiğiyle gösterilen çözümün kalitesini belirlemek için kullanılır.

Seçim: Evrim Stratejisinde seçim operatörleri, önce birleşim için ebeveynleri seçmek ve ikincisi, bir sonraki kuşağa geçen bireyleri belirlemek olmak üzere iki ana işleve sahiptir.

Evrim Stratejisi, daha önce de belirtildiği gibi Evrimsel Hesaplamayı temel alır, ayrıca Kesme Seçimi tekniğini kullanır; bu, bireysel seçimdir ve mutasyon en çok kullanılan operattır. Gelişme Stratejisi, iki farklı algoritma alt bölümlere ayrılabilir:

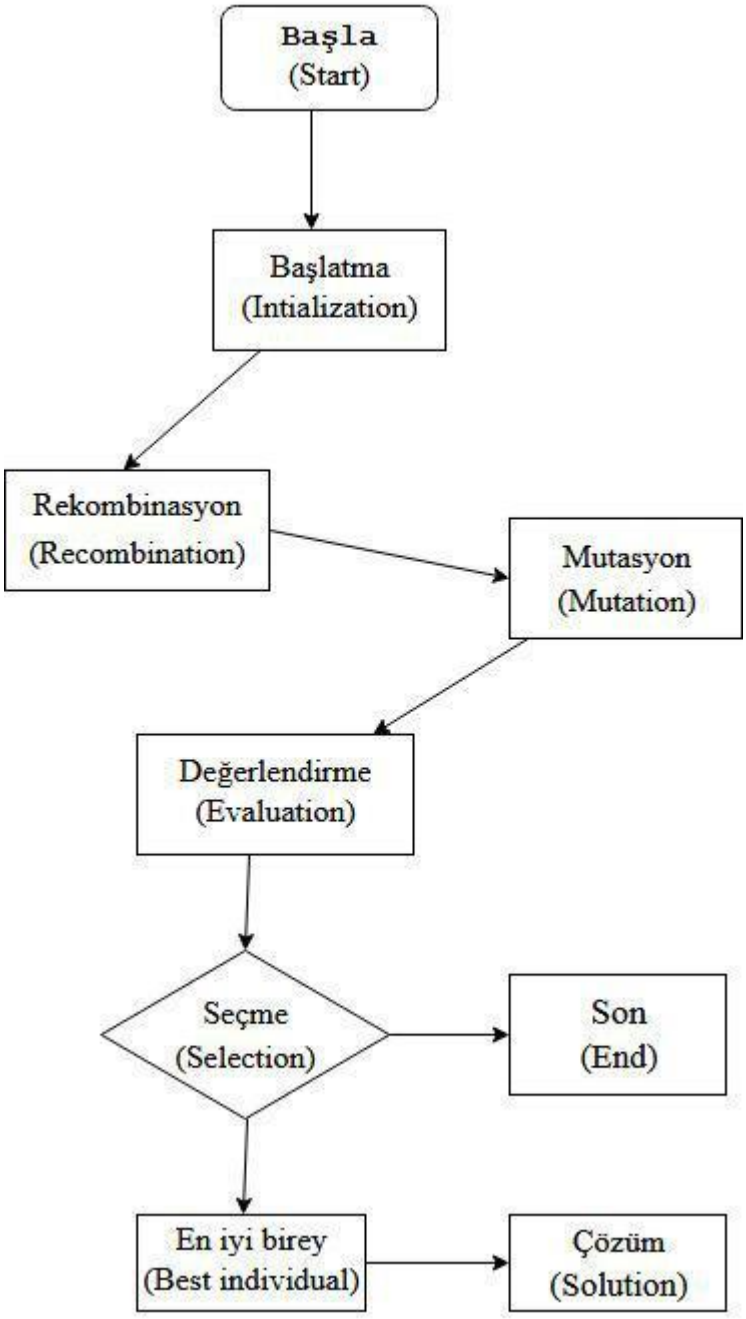
(μ, λ) ve (μ + λ), burada μ μ velilerin sayısıdır ve λ, çocuk sayısıdır. İlk algoritma ebeveynleri ve çocukları ayırt etti, ancak ikincisi her ikisini de ekledi, araştırma ilk algoritmayı (μ, λ) kullansa da [9].

Bu algoritma, rasgele seçilen, bu süreci uygulayarak yinelendirilen popülasyon sayısının λ'ı, tüm bireylerin uygunluğunun değerlendirildiği, ardından uygun (uygun olanların) seçildiği (μ kesit seçimi)), ebeveyn olduklarında normal mutasyonları kullanarak λ / μ yavruları üretirler, bu nedenle yeni neslin bir sonraki nesli haline gelecek ve silinen ebeveynlerin yerini alacak yeni nesiller üretilir, tekrar tekrar başlar.

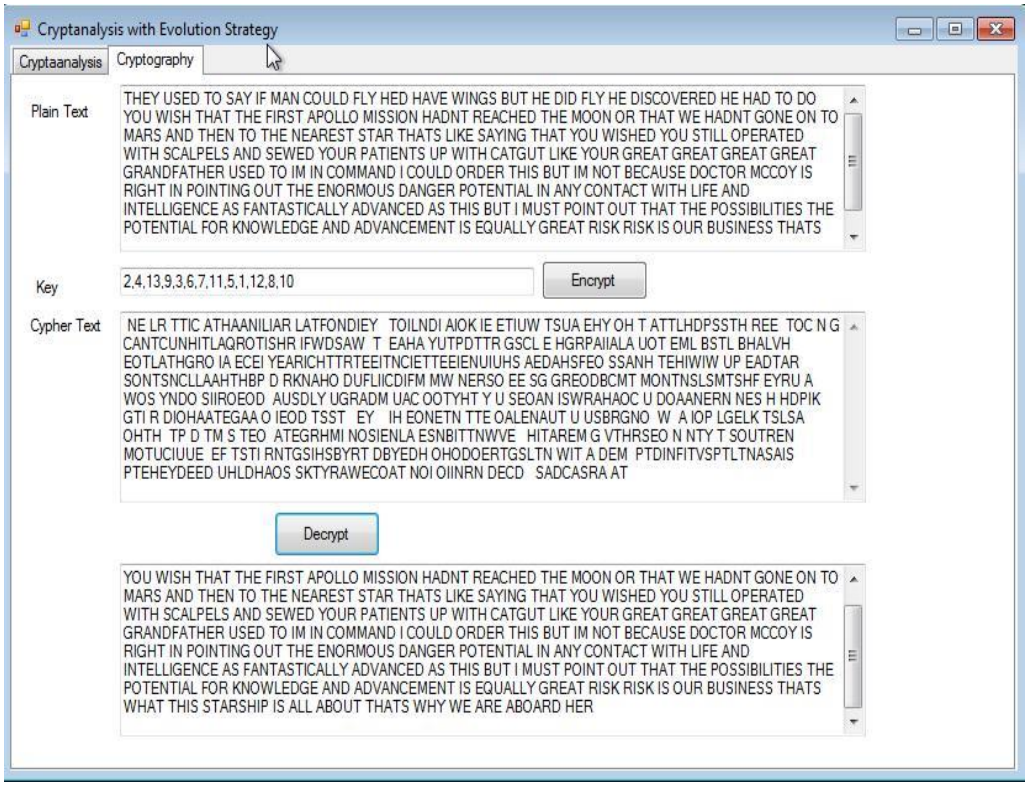
Ebeveynler (μ) her zaman, yeni nesillere geçmek için yeni çocuk (λ) oluşturur; dolayısıyla λ μ'in çarpımıdır, örneğin μ = 6 ve λ = 24 ise, 6'nın çarpanı 24'tür (6, 24). Algoritmanın sözde kodu aşağıdaki gibidir [9].

1. μ seçilen anne sayısı.
2. Anne ve babalar tarafından üretilen çocukların sayısı.
3. P {}.
4. λ times için yapın. Başlangıç Nüfusunu Oluşturun
5. P <----- P ∪ {yeni rasgele bireysel}
6. Uygun (en iyi) birey
7. Tekrar edin
8. Her bireysel Pi ∈P için
9. Tanıdığımızı Değerlendirin (Pi)
10. En iyisi = En iyi veya Zindeli (Pi)> Zindeli (En İyi) ise
11. En İyi Pi
12. Q Fitness () en büyük olan P'deki μ bireyleri. Kesme Seçimi
13. P {}. Katılmak yalnızca P'yi çocuklarla değiştirerek yapılır
14. Her birey için Qj ∈ Q do
15. λ / μ zamanları için yapın
16. P <----- ∪ {Mutate (Copy (Qj))}
17. En iyisi en ideal çözümdür oluncaya kadar veya zamanımız tükenene dek

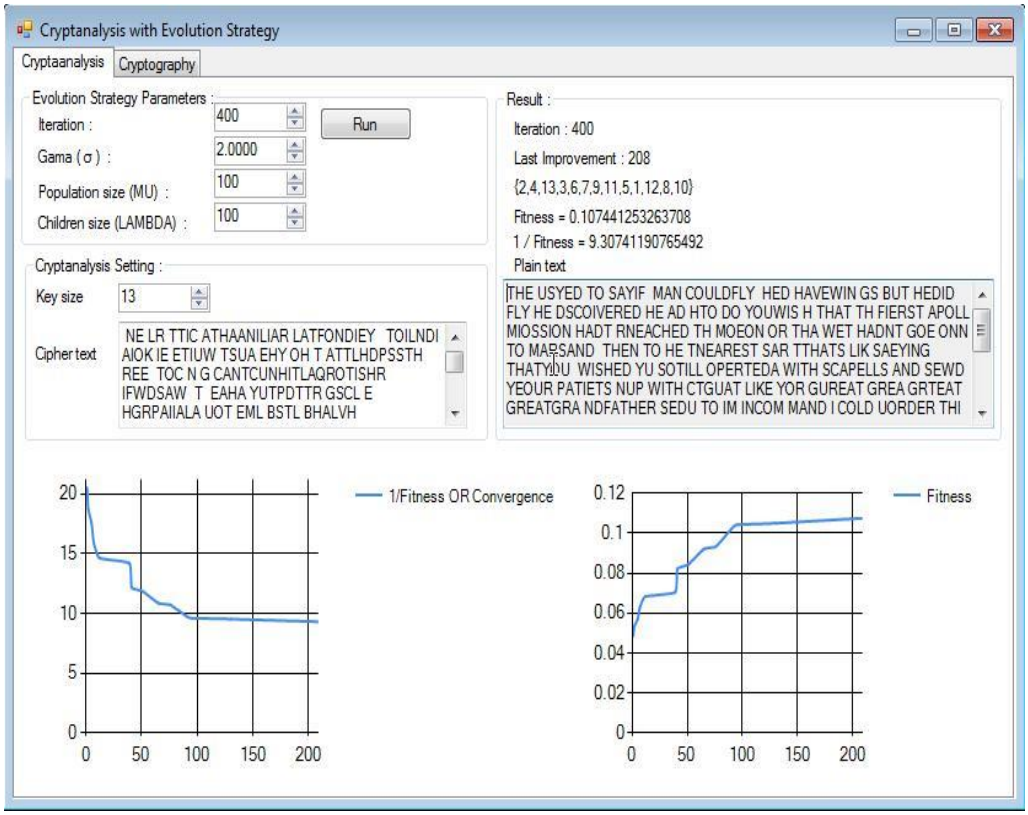
18. En iyisi dönün



Şekil 2.1. Evrim Stratejisinin nasıl çalıştığını açıklamaktadır.



Şekil 2.2. Düz metinleri 13 uzunluk anahtarlı Sütun Transkript Cipher kullanarak şifrelemek.



Şekil 2.3. ES'yi kullanarak Sütunlu Transpozisyon Şifresini çözmek, anahtar uzunluğunu elde etmek, uygunluk ve yakınsama grafikler çizmek.

2.4. Uygunluk Fonksiyonu

Fitness Fonksiyonu verilen bir problemin tüm değişkenleri için en iyi çözümü bulmak için kullanılır, araştırmalar daima bigramlara katılmak için fitness fonksiyonunu ve trigram frekans istatistiklerini kullanır. Bu çalışma, Evrim Stratejisi saldırısı için uygulanan aşağıdaki denklemi içeren bigram ve trigram modellerini uygular [28].

$$F_f = \alpha \sum_{i=1}^{6} C_i.B_i + \beta \sum_{j=1}^{4} D_j.T_j \tag{2.1}$$

Ci, bigram ve trigram harflerinin frekansını gösterirken, 1 <i <6 ve 1 <j <4 için, Bi ve Tj, sık sık bigram ve trigram harflerinin modelini doğru bir şekilde gösterir. Örneğin, Şekil 2.2.'de gösterilen şifre metni, "THY'yi hesaplamak için bu şekilde başlar, TH paragrafında üç kez görünür ve toplam sayı 33'tür, karakterleri ikiye (bigram) gruplarlar Örneğin, TH, HE, EY, Y_, _U, US, ..., EY gibi 32 bigram grubunu, burada _ alanı temsil eder, bu nedenle TH'nin frekansı 2/32 olur, trigramlar aynıdır. Tablo 2.2., araştırmada uygulanan fitness ağırlık modunu göstermektedir.

Araştırmada fitness fonksiyonunun amacı, şifreli metinden elde edilen bigramların ve trigramların frekansını hesaplamak ve bunları Tablo 2.2.'te gösterilen ilgili puanlarla çarpmaktır.

Tablo 2.2. Fitness fonksiyonunda kullanılan bigram ve trigramların puanı.

Bigram	Score	Trigram	Score
TH	2	THE	5
HE	1	ING	5
IN	1	AND	5
ER	1	EEE	-5
AN	1		
ED	1		

2.5. Sütunlu Transpozisyon Şifrelemelerini Çözmek

Kriptografi daima mesajı şifrelenmiş bir biçime dönüştürür, bu da mesajın karışık ve anlamsız olmasını sağlar. İletiyi okunaklı olmayan bir tasarıma dönüştürmek için kullanılabilecek iki yöntem vardır, Değiştirme ve Taşıma. Transpozisyon Şifreleri düz metin sırasını değiştirir, ancak bu çalışmanın ilgilendiği Sütunlu Transpozisyon Şifrelemeleri vardır, burada düz metin yatay olarak yazılır ve daha sonra Tablo 2.3.'de gösterildiği gibi dikey olarak okunur.

Ayrıca Sütunlu Transpozisyon Şifrelemeleri üç tipe ayrılabilir: birincisi ROT-13 şifreleme şemasını uygulayan tek bir Transpozisyon, ikincisi Tek Geçiş ve Çift Transpozisyonu birleştiren Caeser Cipher ve Triple Transposition'ı uygulayan Çift Transpozisyontur [18].

ROT13 ilk 1980lerde kullanılmaya başlanan ve Sezar şifresi olarak bilinen oldukça basit bir şifreleme tekniğidir.Mantık olarak İngilizcedeki her harfin kendisinden sonraki 13üncü harf ile değiştirilmesidir.Yani bakıcak olursak **hell-world**ün rot13 ile şifreli hali **uryy-jbeyq** dir. Bununla birlikte, Geçiş Şifrelemelerinin yukarıdaki üç tekniği şifreleme ve şifre çözme bakımından farklılık gösterir, ancak şifre metninin çözülmesinin tersine çevrilmesi olsa da aynı tekniklere sahip olabilirler. ROT tekniğini uygulayan Tek Geçiş, ilk önce normal metni alarak ve 13 pozisyon atlayıp daha sonra 13. konumda harfi yazarak basittir, ROT-13 şemasını kullanarak şifreledikten sonra Sütun Transkripsiyonu kullanılır, burada anahtar ve düz metin Tablo 2.3'de gösterildiği gibi tasarım tablosuna doldurulur [18].

Açık metin = Meet me at next mid night
Anahtar = FANCY
ROT-13 tekniği uygulanır, böylece düz metin şu şekilde görünecektir.
Şifreli metin = zrrg zr ng arkg zvq avtug
Daha sonra bu şifre metni aşağıdaki tabloda doldurulmuştur.
metni aşağıdaki tabloda doldurulmuştur.

Tablo 2.3. ROT-13 şifre metni tabloya doldurulmuştur.

F	A	N	C	Y
3	1	4	2	5
z	r	r	g	z
r	n	g	a	r
k	g	z	v	q
a	v	t	u	g

Şifreli metin dikey olarak okuduktan sonra aşağıdaki Sütun Transkript Cipher elde edildi.

Şifreli metin = rngv gavu zrka rgzt zrqg

Bu tekniği kullanan çalışma, öncelikle atılan adımları tersine çevirerek kırılmakta, ROT-13 tekniği kullanılmaktadır.

İkincisi, sözcükler tabloda doldurulur, örnek olarak sözcükler birlikte dört harf halinde gruplanır, sonra cracker sözcüklerin 5 olduğu ve 4 harf halinde gruplandığını fark eder ve bundan sonra tablo pusulanın çizileceğini 5 * 4 ve nihayet sözcüklerin yeniden düzenlenmesi gerekiyor.

Buna karşın bu stil sıkıcıdır ve kelimeleri yeniden düzenlerken bazı hatalar getirebilir, ancak bu çalışma şifreleme ve şifre çözme yöntemini, Evrim Stratejisi yöntemini kullanarak geliştirmiştir ve bu yöntem her ikisini de karşılar. Benzer şekilde, bu çalışma manuel olarak geliştirilmelidir ve düz metin şifrelendiğinde hatalar ve hatalar olabilir, anahtar sahibi bile şifre metnini tekrar elde etmek için bazı yorulmuş adımlar atmıştır.

2.6. Genetik Algoritmalar Kullanan Transpozisyon Şifrelemelerine Saldırı

Genetik Algoritmalar giriş bölümünde daha önce belirtildiği gibi Evrimsel Hesaplama Algoritmalarına aittir. Bu araştırmada şifreli metnin şifresini çözmek için

yeni bir uygunluk fonksiyonuna sahip Genetik Algoritmalar uygulanmıştır, en büyük bigramlar ve trigramlar temel alınarak fitness hesaplanırken popülasyonun büyüklüğü, çaprazlama ve mutasyon operatörleri rastgele seçilmiştir. Son olarak GA algoritması, 25'e kadar anahtarla Geçiş Şifrelemelerini deşifre eder [19].

Bu araştırmada, nüfus daha sonra bir dizi stokastik operatörün yinelemeli uygulamasıyla evrimleşti. Basit GA biçimi, başlatma, seçme, çaprazlama ve mutasyon olmak üzere dört tür operatör içerir. Araştırma Genetik Algoritma ve Fitnes fonksiyonunu uygulamış olmasına rağmen, fitness, çeşitli şifre metni uzunluğu ve anahtarıyla belirli zamanlarda bigramlar ve trigramlar toplamaktadır.

Ayrıca araştırma, örneğin 1000 karakter içeren şifre metninin 143 ila 148 saniye arasında (2.38 - 2.46 dakika) ayrıldığı, aynı zamanda 268 ile 2000 karakter şifreleme metninin deşifre edildiği şifre metinlerini parçaladı 276 saniye (4.46 - 4.6 dakika).

Benzer şekilde, Genetik Algoritma kullanılan çalışma gerçekten yavaş çünkü 2000 karakterin 4 dakikada çözülmesi Algoritmanın kalitesini düşürüyor ancak Evolution strateji Algoritması'nda uygulandığında anlamıyla daha az zaman, örneğin, 1000 karakter 18 saniye içinde şifresi çözülen şifre metni Ayrıca 2000 karakterden oluşan bir karakter 1.60 dakika çözülmüş, bu nedenle Evrim Stratejisi Genetik Algoritma metodundan [19] daha etkilidir.

2.7. Cuckoo Arama Algoritması Kullanılarak Geçiş Şifrelemeleri

Cuckoo Arama Algoritmaları, Cuckoo kuş davranışını temel alan ve Yang ve Deb tarafından geliştirilen yeni bir Metaheuristik tekniktir [20-21]. Dahası, optimizasyon problemlerini çözmeye yönelik algoritmalar daha fazladır ve Cuckoo Search bunları içerir. Bu algoritmalar, herhangi bir guguk kuşunun zorunlu kan parazitliği ile yumurtalarını diğer türlerin yuvalarına (kuşlar) koyarak motive eder.

Cuckoo Arama Algoritması üç önemli kurala [22] dayanmaktadır;

- Her Cuckoo bir kez yumurta koyar ve rastgele seçilmiş yuvasında yumurtasını atar.
- Kaliteli yumurtaları olan en güzel yuva, bir sonraki kuşağa geçecektir.
- Yuva, sabit sayıda yumurta tutabilir ve cuckoos tarafından döşenen yumurta araştırılır.

Ayrıca, bu araştırmada, guguk arama ile uygunluk fonksiyonu birleştirildi; bu fonksiyon, dil istatistiklerinden elde edildi ve aday anahtarı, bilinmeyen dilin n-gram istatistikleriyle eşleştirildi [23].

Evrim araştırması basit metinleri ve anahtarı çabucak getirirken, bu araştırmada sadece 13 karakter içeren anahtar 9 saniye içinde kurtarıldığında anahtara yazılan metin mesajını değil, yalnızca kurtarılan zamanı gösterecektir.

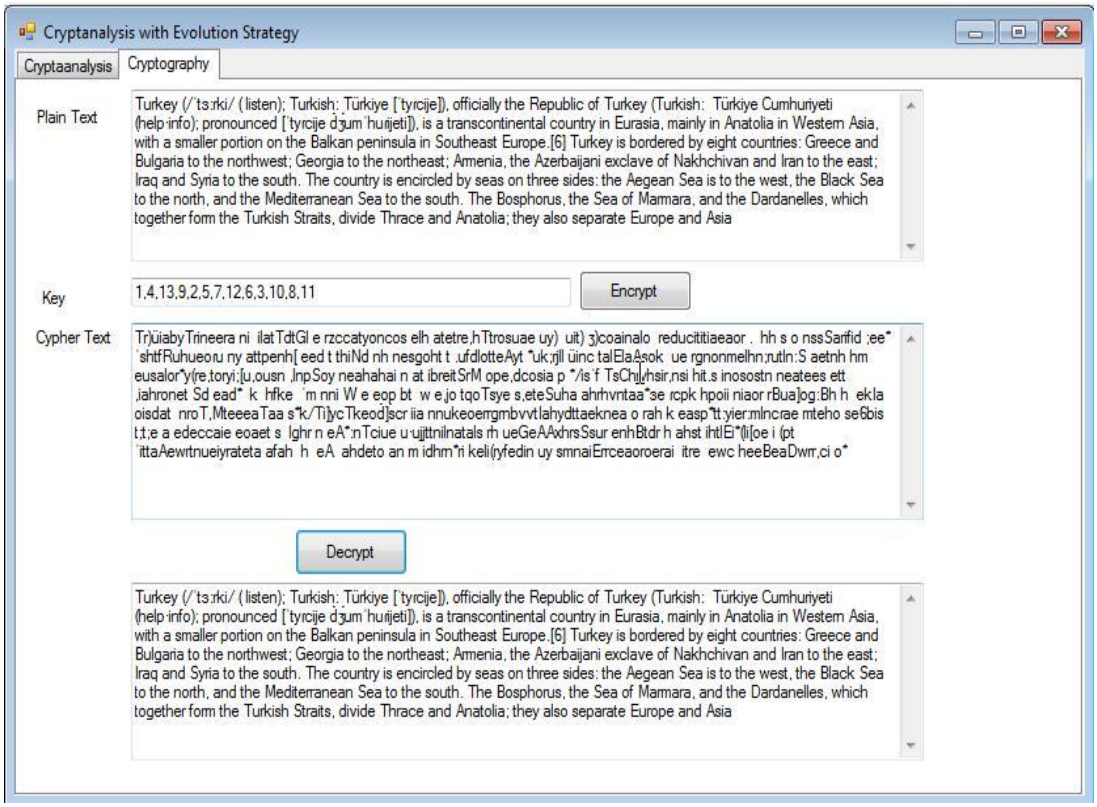
BÖLÜM 3. ARAŞTIRMA ANALİZİ

3.1. Giriş

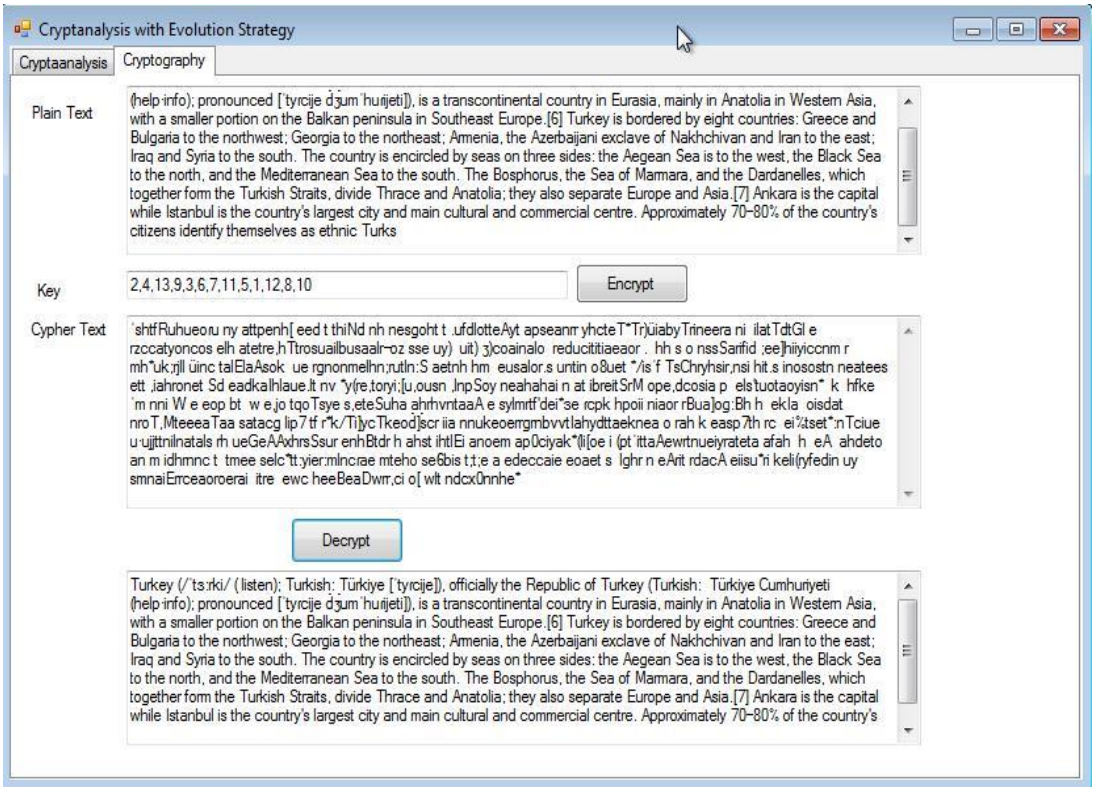
Çeşitli yöntemlerle ve iki farklı fiziksel halde kurutulmuş olan üzüm çekirdeği ekstraktlarının toplam fenolik madde miktarı Tablo 3.1.'de gösterilmiştir. Standart olarak gallik asit kalibrasyon eğrileri oluşturulmuştur (Şekil 3.1.).

3.2. Sütunlu Transpozisyon Şifresi ve Uygunluk Foksiyonu

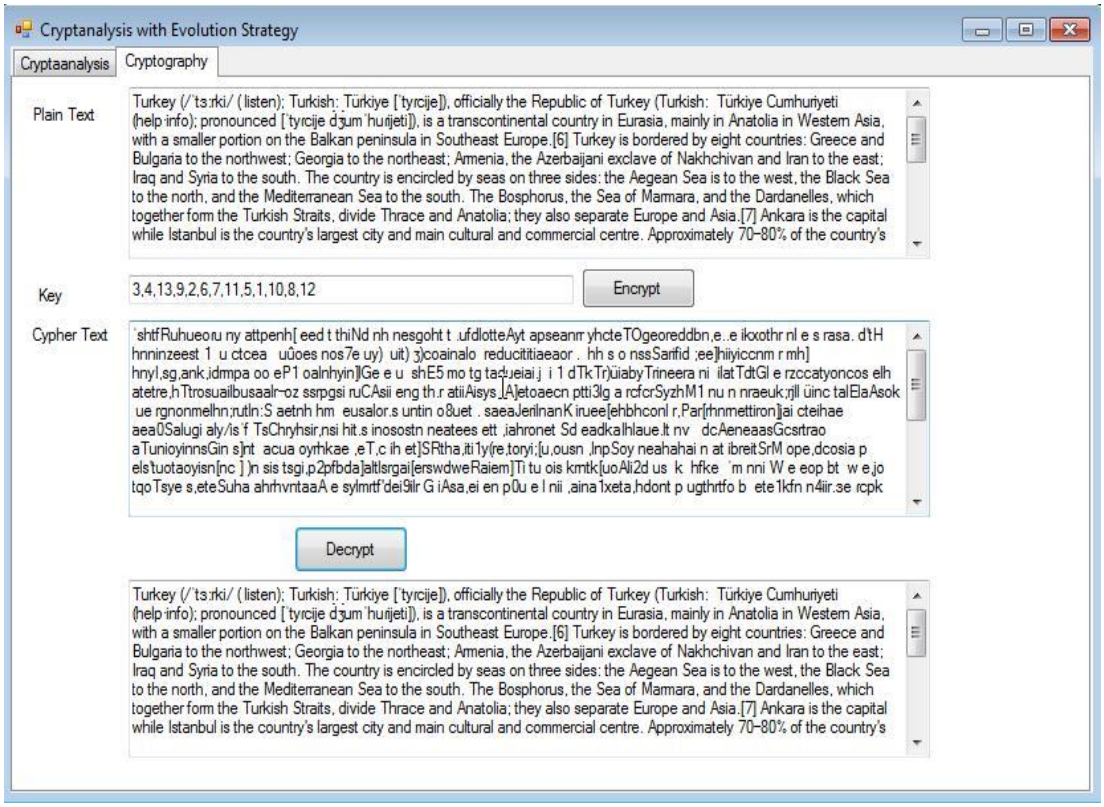
İlk olarak, çalışma, Şekil 2.1.'de gösterildiği gibi düz metinleri şifrelemek için Geçiş Şifrelemeleri'ni kullandı, düz metin, anahtar ve şifre metnini içeren 3 parst içermektedir. Uygulanan tuş on üç ve düz metin uzun veya kısa olabilir, ancak anahtar çeşitli numaralar kullanılarak düzenlenebilir, ancak uygulama on üç tuşla şifrelenmiş düz metinleri deşifre edebildiğinden, anahtar 13 olmalı. Tuşlar aşağıdaki örneklerde olduğu gibi harflerin herhangi bir türünü şifrelemek için kullanılabilir, ancak. Şekil 2.1.,3.1. ve 3.2. tamamen farklı düz metinler ve tuşlar göstermektedir.



Şekil 3.1. Sekiz yüz harfi on üç anahtar uzunluğunda şifrelemek.



Şekil 3.2. On üç uzunluk anahtarla bin harf şifrelemek.



Şekil 3.3. İki anahtar kelimeyi on üç anahtar uzunluğunda şifrelemek.

Uygulama, Sütunlu Taşıma Şifreleyicileri kullanarak farklı uzunlukta tüm düz metinleri şifrelemekte ve yalnızca Şekil 2.1.,2.2.,3.1.,3.2.,3.3.'te gösterildiği gibi karışık harfleri gösteren sayıları göstermektedir.

3.3. Evrim Stratejisi Algoritması Sonuçları

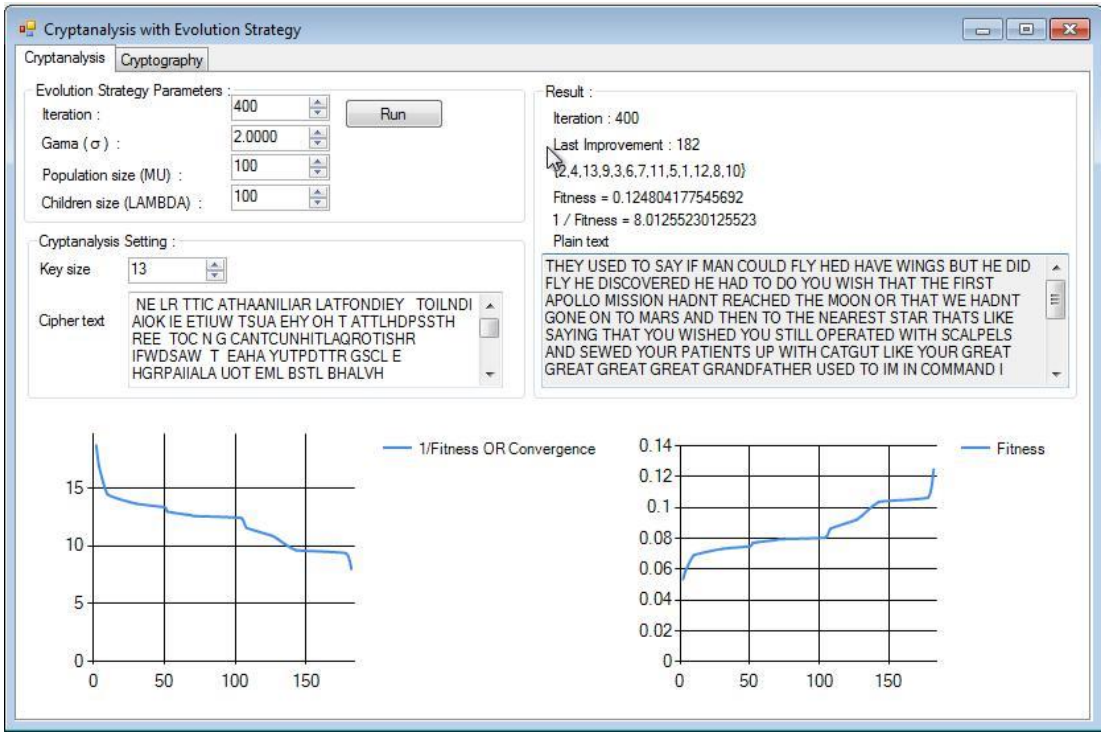
Bu araştırmada evrim stratejisi, şifre metnini Şekil 2.2.'de gösterildiği gibi kırmak için uygulanmıştır. Kullanılan parametreler yineleme, Gama, nüfus büyüklüğü ve çocuk büyüklüğü olmakla birlikte, Evrim Stratejisi fitness işleviyle birleştirildiğinde şifre metni fitness fonksiyonu bigramları ve trigramları şifreleme metninden toplar ve bunları bir araya getirirken, deşifre edilmesi kolaydır.

Uygulama Şekil 2.2.'de gösterildiği gibi otomatik olarak şifreleme metnini ve anahtar uzunluğunu şifreleme bölümünden alıyor. Bu bölümde Evrim Stratejisi Parametreleri (yineleme, nüfus büyüklüğü, gama, çocuk boyutu), Kriptoanalizi Parametreleri (anahtar boyutu, şifre metni) ve Sonuç (yineleme, son gelişme, anahtar, fitness, yakınsama, düz metin) tüm bu parametreler Bölüm 3'te belirlenir. Şifre ve anahtar

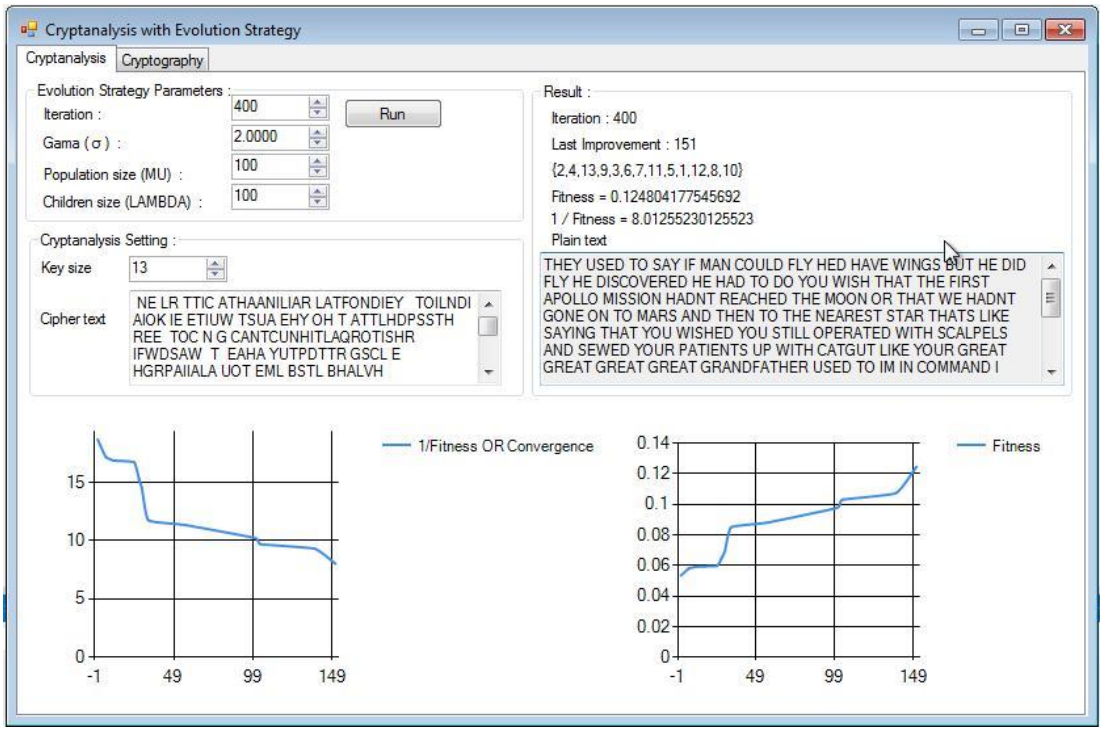
konumlarını getirdiğinde, uygulama metnin şifresini çözmeye hazırdır. Evrim Stratejisi parametreleri, daha önce belirtildiği üzere uygunluk fonksiyonuyla birleştirilir. Ayrıca yineleme 400, gamma 2000, popülasyon boyutu 100, çocukların boyutu 100 olarak belirlenmiş ancak bu değerler uzatılabilir.

Dahası, evrim parametreleri ayarlandığında ve çalıştır düğmesine tıklandığında, sonuç dokunma çözümü bulmaya başlıyor. Bu aşamada öncelikle, tüm parametreler aynı anda birlikte çalışıyor, yineleme sınırı bulana kadar son iyileştirme gösteriyor ve uygulamanın aynı özniteliklerle yeniden başlatılması durumunda uygulamanın uyumluluğundan sonra zamanla azalacak (Şekil 3.4., 3.5.) de gösterildiği gibi şifre metni, ayrıca anahtar uyumu fonksiyonu (düz metinlerin bigramlarını ve trigramlarını evrim parametreleriyle çalıştıkça düzenler), yakınsaklık ve düz metin. Benzer şekilde yineleme ES parametrelerinde belirtilen sınırlamaya ulaşmadığından yineleme, son iyileştirmeler, tuşlar, uygunluk fonksiyonu, yakınsaklık, düz metin, yakınsaklık ve uyum fonksiyonu grafikleri değişiyor.

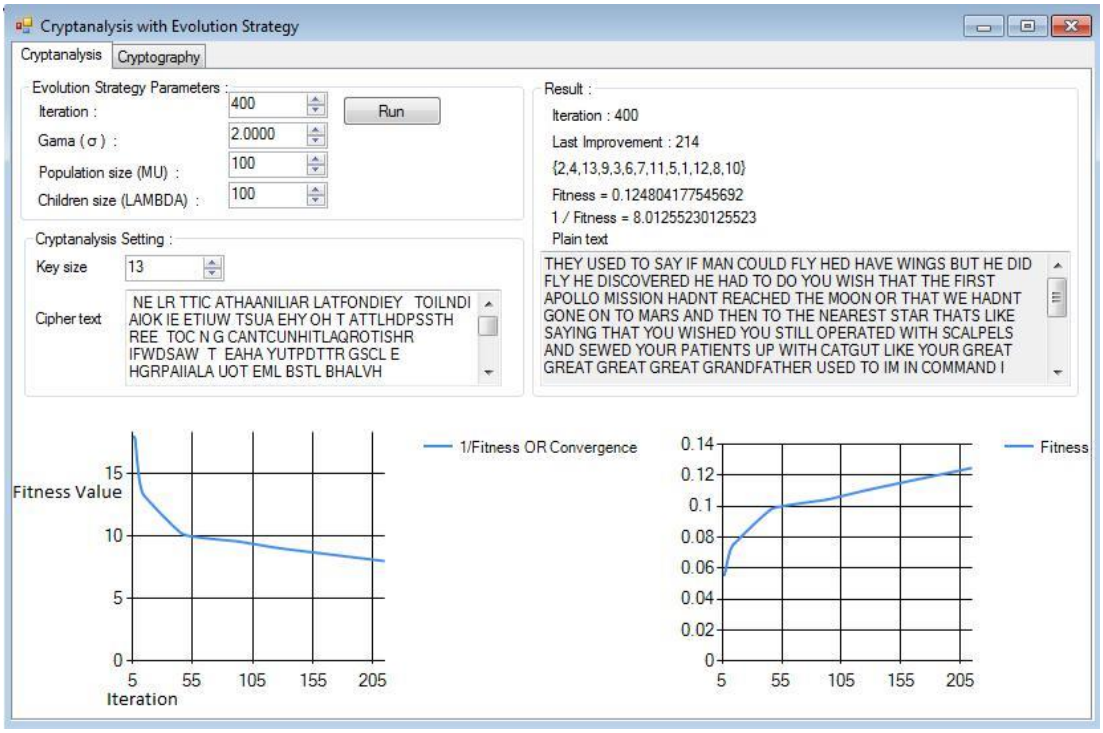
Spor ve yakınsaklık iki önemli parametre uygunluk değeri ve Şekil 3.6., 10'da gösterildiği gibi yineleme içermektedir. Buna rağmen, Yakınsama 1 / Spor'dur ve çözümün doğruluğunu gösterdiği için, Fitness ve yakınsama grafikleri tersine çevrilir. Örneğin, Şekil 3.4.'te uygunluk 0.12 ve yakınsama 1 / 0.12, yani 8.01'dir. Benzer şekilde, uygunluk ve yakınsama, uygunluk değerini içeren grafikleri tersine çevirmiş ve yineleme Şekil 3.6., 3.7.'de gösterilmiştir. Her bir şifrenin uygunluğu ve yakınsaması da şekil ve noktalarda değişiklik göstermektedir (Şekil 3.6., 3.7.).



Şekil 3.4. Deşifre edilen 700 harf gösterir ve son iyileştirme 182'dir.



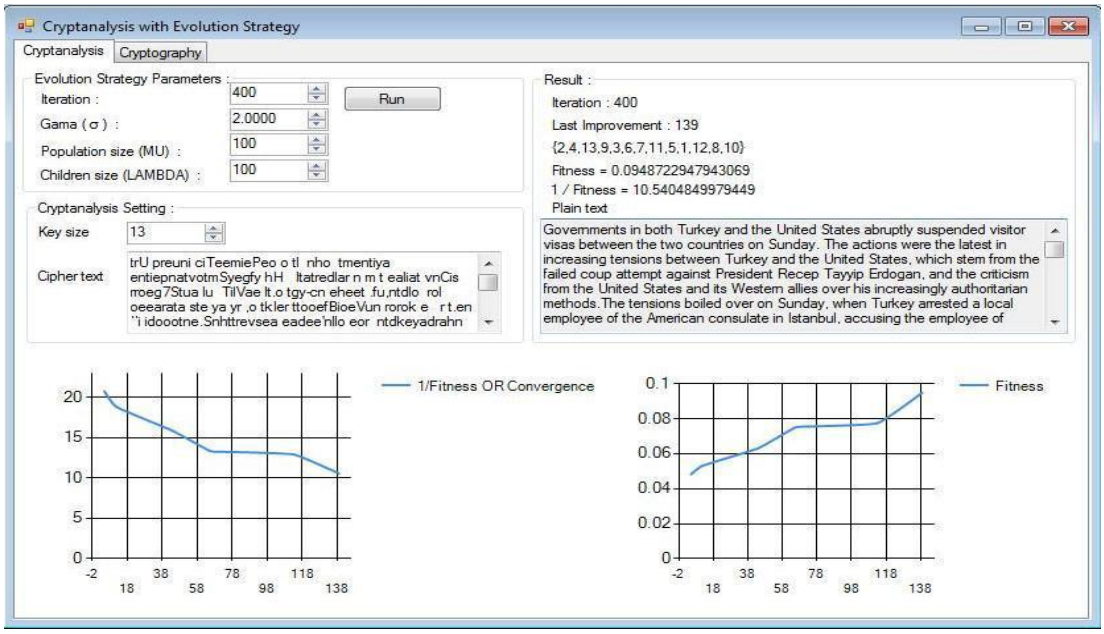
Şekil 3.5. Şifresini çözen 700 harfi gösterir ve şifre çözme son gelişme azalmıştır.



Şekil 3.6. 400 yinelemenin ve 700 kelimenin uygunluğunu ve yakınsamasını gösterir.

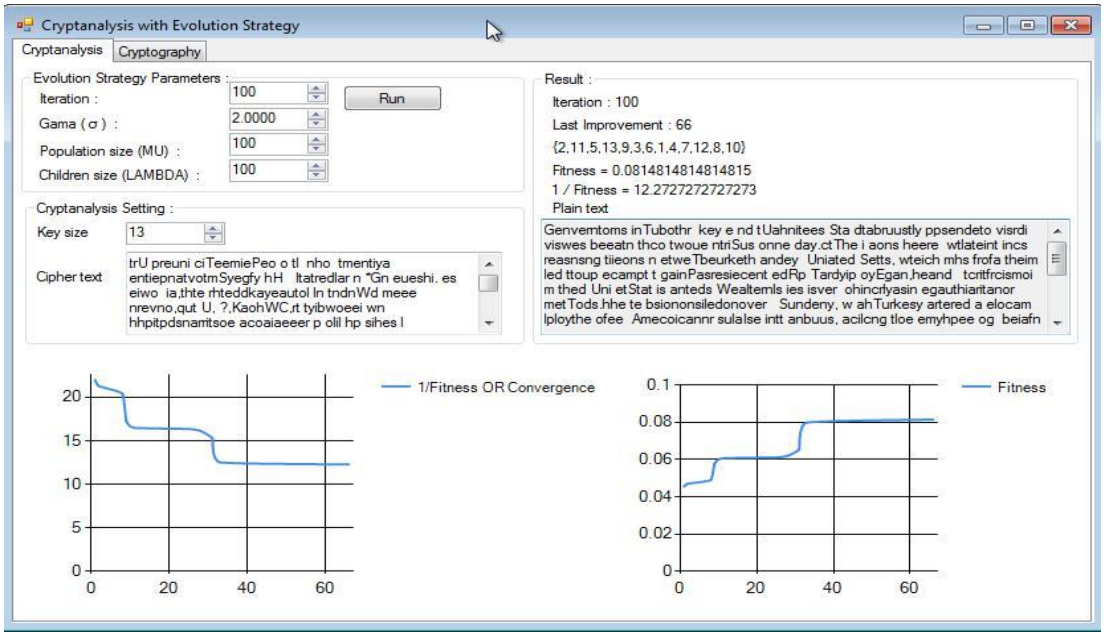
Şekil 3.7. 350 iterasyonun ve 700 harfin zindeliği ve yakınsaması.

Ayrıca, Fitness azalır ve yakınsama şifre metin arttıkça, örneğin 5000 harfler içeren şifre metni artıyor artışdır 0,09 Fitness ve 10,5 yakınsama Şekil 3.8. gösterildiği gibi, aynı zamanda içeren şifre metni 700 harfler 0,12 ve yakınsama 8 Şekil 3.6. gösterildiği gibi uygunluk vardır.

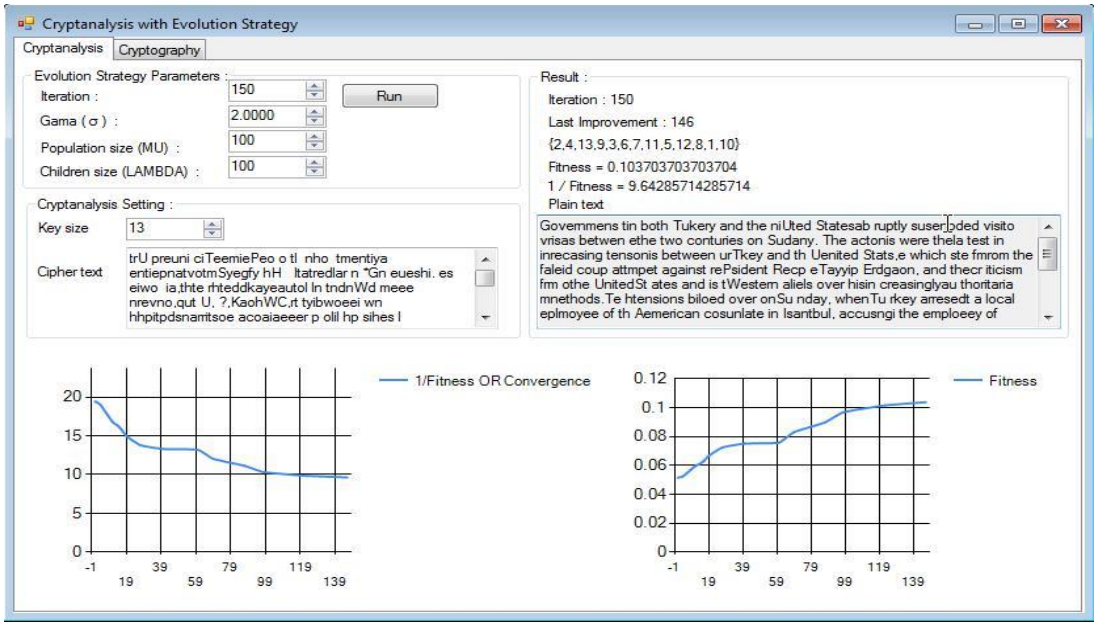


Şekil 3.8. 0.09 uyum ve 10 yakınsama olan 5000 harfli şifre metni.

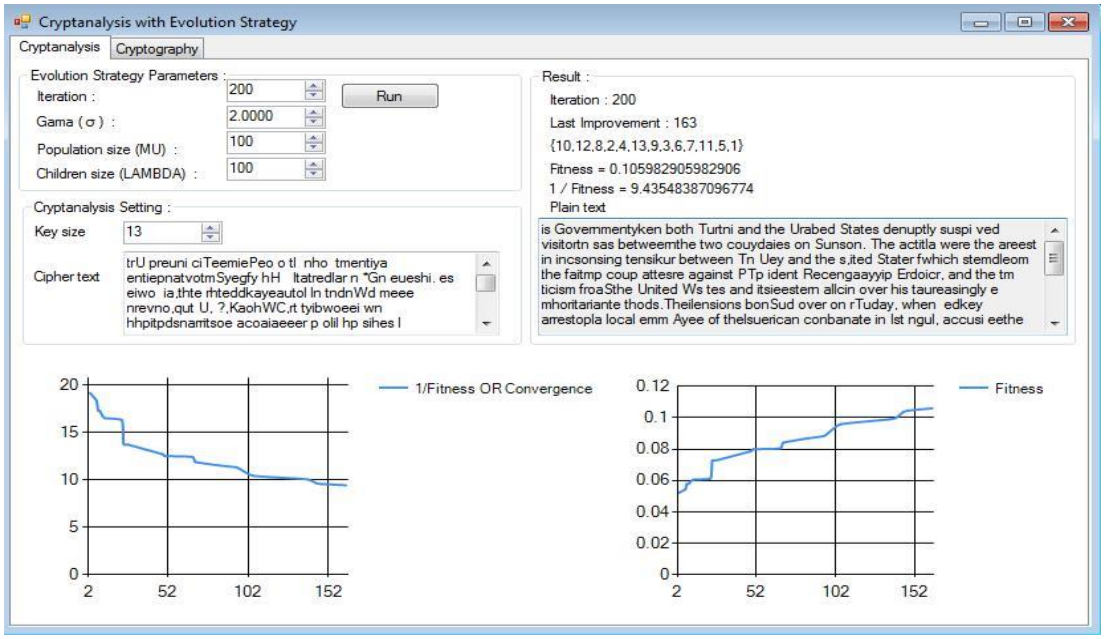
Daha önce de açıklandığı gibi, araştırmanın sunduğu başka bir iyi düşünce vardır; bu yineleme, şifre metnini arttırır ve düz metin, Şekil 3.6.,3.7. ve 11'de gösterildiği gibi mükemmel olarak alınır ve 350 ve 400 iterasyon gösterir. Öte yandan, iterasyon azaltılırsa, şifre metni ve tuşları, Şekil 3.9., 3.13. ve 3.14.'te gösterildiği gibi düzgün bir şekilde getirilememektedir, çünkü uygulamanın kullandığı minimum yineleme 400'tür, ancak tümü 350 iterasyon uygulandığında bunları alabilir.



Şekil 3.9. 100 iterasyon, şifre metni ve anahtarı bulunamıyor.



Şekil 3.10. 150 tekrarlama, şifre metni ve tuşu kurtarılamaz



Şekil 3.11. 150 iterasyon, şifre metni ve tuşu kurtarılamaz

Buna ek olarak, Evrim Stratejisi algoritması, Tablo 3.1. ve Bölüm 2'de gösterildiği gibi, diğer algoritmalara kıyasla şifreyi hızlı bir şekilde deşifrelemektedir.

Table 3.1. Farklı şifre metnine geri kazanılan süre.

Iterasyon	Şifre metni uzunluğu	Anahtar Uzunluğu.	Kurtarılan Zaman (s)
400	1000 harf	13	21
400	2000 harf	13	60
400	5000 harf	13	300

Ayrıca, tuşlar sipariş edildiğinde uygulama şifre metnini çabucak kırıyor ancak tablo 3.2.'de gösterildiği gibi sırasız tuşları kullanırken bir süre alır.

Tablo 3.2. Sipariş edilen ve sıralanmamış anahtarlar arasındaki farkın (zaman) karşılaştırılması.

Iterasyon	Şifre metni uzunluğu	Anahtar Uzunluğu.	Kurtarılan Zaman (s)
400	700 harf	sirali	12
400	700 harf	sirasiz	14
400	1000 harf	sirali	20
400	1000 characters	sirasiz	23

Tablo 3.3. Uygulanan algoritmaların karşılaştırılması

Algoritmalar	Hızlı	Yavaş	Hepsini kırar	Decıpher less	Bazılarını çözer	Birçok parametre	Çok algoritma ile uygulanır.	Az algoritma ile uygulandı
Genetik algoritmalar		X		X	X	X		X
Cuckoo Arama Algoritması.		X		X	X	X		X
Evrim stratejisi.	X						X	
ROT-13, Transpozion Speraları ile birlikte		X		X	X	X		X
Uygunluk fonksiyonu	X		X				X	

Yukarıdaki tablo, çalışmada uygulanan algoritmaların avantajlarını, dezavantajlarını ve etkinliğini açıklamaktadır.

BÖLÜM 4. BÜTÜNLE MÜCADELE VE DİĞER GÖRÜŞMELER

Bu araştırmada, uygulanan Evrim Stratejisi Algoritması, araştırmada kullanılan tüm parametrelere en iyi çözümü bulmuştur. Aynı şekilde, C # 'da yerleşik olan iki farklı bileşen, şifreleme ve kriptan analizi de dahil olmak üzere her iki faktör de mükemmel çalışıyor; şifreleme düz metinleri şifreleyip şifreleyebilen Sütun Transkripsiyon Şifrelemelerini kullanıyordu. Benzer şekilde, şifreleme analizi bölüm 5'de analiz edildiği gibi tüm şifre metinlerini deşifre etmektedir; ayrıca, Evrim Stratejisi ile birleştirilen fitness fonksiyonu deşifre etmek ve sonuçta mükemmel bir çözüm getirmek için şifre metni harfleri düzenlemektedir.

Buna uygun olarak, uygulama fitness fonksiyonunu ve birbirlerine ters çalıştıkları yakınsamayı gösteriyor. Uygulamanın karşılaştığı ve gözden geçirilmesi ve çözülmesi gereken bazı zorluklar olmasına rağmen, örneğin şifre metni ve iterasyon azaltıldığında, alınan anahtarlar ve düz metin doğru olmayacaktır, buna bağlı olarak uygulama sabit tekrarlama (400), ancak yineleme 400 ve daha fazla olursa, hem düz metin hem de tuşlar geri yüklenir. Son olarak, uygulama aslında daha iyi ve diğerlerinden daha hızlı çalışıyor; daha önce 2. bölümde tanımlanan çeşitli algoritmaları ve aynı uygunluk fonksiyonunu uygulayan kişiler.

KAYNAKLAR

[1] Henk C.A. Van Tilborg, Fundamentals of cryptology, Kluwer academıc publıshers,10-50, 2002.

[2] Ross anderson, Security Engineering: A guide to building dependable distributed systems (vol. 2). Wiley. 2002.

[3] William stallings, Cryptography and network Security,(vol.4), McGraw-Hill), 2005.

[4] Dan boneh, Victor shoup, A graduate course in Applied cryptography, Standford, 6-25, 2015.

[5] Sans institute, Understanding and selecting a data loss prevention solution, 5-11, 2014.

[6] Holland, J. H, Adaptation in natural and artificial systems. Ann arbor, University of Machigan Press. 10-37, 1975.

[7] Goldberg, D, Genetic algorithms in search optmization and machine learning, Addison-wesley, 1989.

[8] Xin-she Yang, Nature inspired Metaheuristic algorithms, University of Cambridge, 2-15, 2010.

[9] Sean luke, Essentials of metaheuristics, George mason University, (vol.2), 9-15, 2015.

[10] Rechenberg, Cybernetic solution path of an experimental problem, Technical report, Minstry of aviation, 1965.

[11] Rechenberg, Evolutions strategie, Optimierung technischer Systeme nach Prinzipien der Biologischen, Evolution. Frammann-Holzboog Verlag, Stuttgart,1973.

[12] Ingo rechenberg, Evolutionäre technik evolutionsstrategie, Doi: 10.1002/biuz.19950250620,100-130, 1995.

[13] H.-P. Schwefel: Numerische optimierung von computer-modellen mittels der Evolutionsstrategie (Birkh¨auser, 1977).

[14] Hans-Paul, Schwefel, Evolution and optimum seeking, Wiley, 11-50, 1995.

[15] L. J. Fogel, A. J. Owens, M. J. Walsh: Arti- ficial Intelligence through Simulated evolutio, Wiley, 1996.

[16] Lishan kang, Advances in Computing and intelligence, third international symposium, Springer, 780-799, 2008.

[17] Malay B. Pramanik. Implementation of Cryptography Technique using Columnar Transposition. International Journal of Computer Applications, 19-23. 2014.

[18] Quist-Aphetsi Kester, A hybrıd cryptosystem based on vıgenère cıpher and columnar transposıtıon cıpher, International Journal of Advanced Technology & Engineering Research, 141-147, 2013.

[19] Omar Alkathiry ve diğerleri, A Powerful Genetic Algorithm to Crack a Transposition Ciphers. International Journal of Future Computer and Communication, 395-399.

[20] Xin-she Yang ve diğerleri, Cuckoo Search via Levy Flights. World Congress on Nature & Biologically Inspired Computing, 210-214. 2009.

[21] Xin-she Yang ve diğerleri, Engineering optimization by cuckoo search, International journal of Mathematical modelling and numerical optimization, 330-343, 2010.

[22] Morteza Heydari ve ve diğerleri, Automated Cryptanalysis of Transposition Ciphers Using Cuckoo search algorithm. International Journal of Computer Science and Mobile Computing, 140-149.

[23] Mark Stamp, Applied Cryptanalysis: Breaking Ciphers in the Real World. Wiley Press, 2007.

[24] Rechenberg, Cybernetic Solution path of an experimental problem, Royal Aircraft establishment, Springer, 1965.

[25] Rechenberg. Evolutionsstrategie: Optimierung technischer Systemenach Prinzipien der Biologischen Evolution. Frammann-Holzboog Verlag, 1975.

[26] Schwefel, P. Hanso, Evolutionsstrategie und numerische Optimierung. PhD thesis, Technical University Berlin, 1975.

[27] Andries P. Engelbrecht, An Introduction to Computational Intelligence. In Evolution Strategies. Wiley, 2150-216, 2007.

[28] Garg, P. and A.M. Sherry, An Improved. Cryptanalytic Attack on Knapsack Cipher using Genetic Algorithm, International Journal of Information and Communication Engineering 3(6): 444-451. 2007.

EKLER

EK1: Algoritma Kodları

Proje çeşitli sınıflar içeriyor ve her birinin kendi kodlaması var, bu nedenle algoritmalar uygun şekilde kodlanıyor.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace Cryptanalysis
{
public class RowTransposition : SecurityAlgorithm
{
readonly int[] key;
public RowTransposition(int[] key)
{
this.key = key;
}
#region Public Methods
public override string Encrypt(string plainText)
{
int columns = 0, rows = 0;
Dictionary<int, int> rowsPositions = FillPositionsDictionary(plainText, ref
columns, ref rows);
char[,] matrix2 = new char[rows, columns];
```

```
//Fill Matrix
int charPosition = 0;
for (int i = 0; i < rows; i++)
{
for (int j = 0; j < columns; j++)
{
if (charPosition < plainText.Length)
{
matrix2[i, j] = plainText[charPosition];
}
else
{
matrix2[i, j] = '*';
}
charPosition++;
}
}
string result = "";
for (int i = 0; i < columns; i++)
{
for (int j = 0; j < rows; j++)

{
result += matrix2[j, rowsPositions[i + 1]];
}
}
return result;
}
public override string Decrypt(string cipherText)
{
int columns = 0, rows = 0;
```

```
Dictionary<int, int> rowsPositions = FillPositionsDictionary(cipherText,
ref columns, ref rows); char[,] matrix = new char[rows, columns];
//Fill Matrix
int charPosition = 0;
for (int i = 0; i < columns; i++)
{
for (int j = 0; j < rows; j++)
{
matrix[j, rowsPositions[i + 1]] = cipherText[charPosition];
charPosition++;
}
}
string result = "";
foreach (char c in matrix)
{
if (c != '*')
{
result += c;
}
}
return result;
}
#endregion
#region Private Methods
private Dictionary<int, int> FillPositionsDictionary(string token, ref int columns, ref
int rows)
{
var result = new Dictionary<int, int>();
columns = key.Length;
rows = (int)Math.Ceiling((double)token.Length / (double)columns); /* we need
something to tell where to start
*           4 3 1 2 5 6 7          Key
```

```
*               0 1 2 3 4 5 6        Value
*/
//attack postponed until two am xyz
for (int i = 0; i < columns; i++)
{
result.Add(key[i], i);
}
return result;
}
#endregion
}
}
```

BigramFrequncy class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace Cryptanalysis.ModelFrequency
{
public class Bigram
{
public string bigram;
public double frequency;
public double Score;
public Bigram(string bigram, double frequency, double Score)

{
this.bigram = bigram;
this.frequency = frequency;
```

```
this.Score = Score;
}
}
public class BigramFrequency
{
public          Dictionary<int, Bigram> bigrams = new Dictionary<int, Bigram>();
public BigramFrequency()
{
bigrams.Add(0, new Bigram("TH", 2.71, 2));
bigrams.Add(1, new Bigram("HE", 2.33, 1));
bigrams.Add(2, new Bigram("IN", 2.03, 1));
bigrams.Add(3, new Bigram("ER", 1.78, 1));
bigrams.Add(4, new Bigram("AN", 1.61, 1));
bigrams.Add(5, new Bigram("ED", 1.08, 1));
}
}
}
```

TrigramFrequency class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace Cryptanalysis.ModelFrequency
{
public class Trigram
{
public string trigram;
public double frequency;
public double Score;
```

```
public Trigram(string trigram, double frequency, double Score)
{
this.trigram = trigram;
this.frequency = frequency;
this.Score = Score;
}
}
public class TrigramFrequency
{
public Dictionary<int, Trigram> trigrams = new Dictionary<int, Trigram>();
public TrigramFrequency()
{
trigrams.Add(0, new Trigram("THE", 1.81, 5)); trigrams.Add(1, new
Trigram("ING", 0.72, 5)); trigrams.Add(2, new Trigram("AND", 0.73, 5));
trigrams.Add(3, new Trigram("EEE", 0.00001, -5)); }

}
}
```

Encryption class

```
using Cryptanalysis.ModelFrequency;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Text.RegularExpressions;

using System.Threading.Tasks;
namespace Cryptanalysis
{
```

```
public class EncryptionKey : IEquatable<EncryptionKey>,
IComparable<EncryptionKey> {

public int[] key;
public double Gama;
private double fitness;
public string                Plaintext="";
public static Random random=new Random();
public double Fitness
{
get { return fitness; }
set { fitness=value;}
}
public EncryptionKey()
{
Gama = Population.Gama;
GenreateRandom();
FitnessFunc();
}
public EncryptionKey(EncryptionKey S)
{

this.key = new int[Population.KeySize];
S.key.CopyTo(this.key, 0);
this.Gama = S.Gama;
this.fitness = S.fitness;
}
///             <summary>
///             Clone the Genotype
///             </summary>
public virtual EncryptionKey Clone()
{
```

```
return new EncryptionKey(this);
}
public void GenreateRandom()
{
key = new int[Population.KeySize];
List<int> tempKey = new List<int>(Population.keyGeneritor); int end =
Population.KeySize;
for (int j = 0; j < Population.KeySize; j++)
{
int m = EncryptionKey.getRandomNumber(0, --end);
if (m <= tempKey.Count)
{
key[j] = tempKey[m];
tempKey.RemoveAt(m);
}
}
}
public static int[] GenreateRandomKey(int keySize )
{
int[] keyTemp = new int[keySize];
int[] keyGeneritor=new int[keySize];
for (int i = 1; i <= keySize; i++)
keyGeneritor[i-1] = i;
List<int> tempKey = new List<int>(keyGeneritor); int end = keySize;
for (int j = 0; j < keySize; j++)
{
int m = EncryptionKey.getRandomNumber(0, --end);
if (m <= tempKey.Count)
{
keyTemp[j] = tempKey[m];
tempKey.RemoveAt(m);
}
```

```
}
return keyTemp;
}
public double FitnessFunc()
{
RowTransposition securityModel = new RowTransposition(key); Plaintext
= securityModel.Decrypt(Population.CipherText); string UpperplainText =
Plaintext.ToUpper();
double sumBigram = 0;
BigramFrequency bigramFrequency = new BigramFrequency(); for (int i = 0; i
< bigramFrequency.bigrams.Count; i++) {
                double C =
Regex.Matches(UpperplainText,bigramFrequency.bigrams[i].bigram).Count /
(UpperplainText.Length - 1.0);
sumBigram += C *
bigramFrequency.bigrams[i].Score; }
double sumTrigram = 0;
TrigramFrequency trigramFrequency = new TrigramFrequency(); for (int i = 0; i
< trigramFrequency.trigrams.Count; i++) {
                double D =
Regex.Matches(UpperplainText,trigramFrequency.trigrams[i].trigram).Count /
(UpperplainText.Length - 1.0);
sumTrigram += D * trigramFrequency.trigrams[i].Score;
}
fitness = 0.7 * sumBigram + 0.3 * sumTrigram; return fitness;
}
#region IEquatable<Genotype> Members
public bool Equals(EncryptionKey other)
{
                if (other == null || !(object.ReferenceEquals(this,
other))||this.key.Length!=other.key.Length) {
```

```
return false;
}
else if (object.ReferenceEquals(this, other))
{
return true;
}
else
{
bool isEquals = true;
for (int i = 0; i < this.key.Length; i++)
{
if (this.key[i] != other.key[i])
{
isEquals = false;
break;
}
}
return isEquals;
}
}
#endregion
#region IComparable<Genotype> Members
///             <summary>
///             Compare two chromosomes
///             </summary>
public int CompareTo(EncryptionKey other)
{
double f = other.fitness;
return (fitness == f) ? 0 : (fitness < f) ? 1 : -1;
}
#endregion
public static int                getRandomNumber(int min=0,int max=0)
```

```
{
double m= random.NextDouble() * (max - min) + min; return (int)Math.Floor(m);
}
}
}
```

Population class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace Cryptanalysis
{
class Population
{
public static double Gama = 1.5;
public static string CipherText;
public static int KeySize;
public static int MAXChange = (int)(KeySize * 0.8); public static
List<int> keyGeneritor = new List<int>();
        public static int MU    = 100;       /* Population size */
                                LAMBDA =
        public static int     100;                /* No. of Offspring */
        public static double    succeedRatio = 0;
        public static   int   MAXGenerations = 400;     /* max. number of generations */
                        Rando random =
        public static   m       new             Random();

/* genotype (GT), a member of the population */
```

```
public List<EncryptionKey> population = new List<EncryptionKey>();
public List<EncryptionKey> newpopulation = new List<EncryptionKey>();
public Population()
{
MAXChange = (int)(KeySize * 0.8);
int j;
keyGeneritor.Clear();
for (j = 1; j <= KeySize; j++)
keyGeneritor.Add(j);
/* Initialise variables within the bounds */
for (j = 0; j < MU; j++)
{
population.Add(new EncryptionKey());
}
population.Sort();
}
///             <summary>
///             The most basic form of the transformation looks like:
///             y1 = sqrt( - 2 ln(x1) ) cos( 2 pi x2 )
///             y2 = sqrt( - 2 ln(x1) ) sin( 2 pi x2 )
///             We start with two independent random numbers, x1 and x2, which
come from a
///uniform distribution (in the range from 0 to 1).
///             </summary>
///             <param name="mean"></param>
///             <param name="stdDev"></param>
///             <returns></returns>
public double NextGaussian(double mean = 0, double stdDev = 1)
{
double u1 = random.NextDouble(); //these are uniform(0,1) random doubles
double u2 = random.NextDouble();
double randStdNormal = Math.Sqrt(-2.0 * Math.Log(u1)) *
```

```
Math.Sin(2.0 * Math.PI * u2); //random normal(0,1)
double randNormal =
mean + stdDev * randStdNormal; //random normal(mean,stdDev^2) return
randNormal;
}
///             <summary>
///             Mutation: A variable selected for mutation is replaced by The
///             previous value of this variable added to (Gama of this variable
///             multiplied by Gaussian noise).
///             </summary>
private void mutate()
{
int succeedNo = 0;
newpopulation.Clear();
for (int i = 0; i < LAMBDA; i++)
{
int index = i % MU;
EncryptionKey offspring = population[index].Clone(); //Moving by using Gama
time by Gaussian Random
int changes = (int)Math.Floor(MAXChange * Population.Gama * NextGaussian();
if (changes < 1)
changes = 1;
if (changes > MAXChange)
changes = MAXChange;
for (int j = 0; j < changes; j++)
{
lt:
int i1 = EncryptionKey.getRandomNumber(0, offspring.key.Length); int i2
= EncryptionKey.getRandomNumber(0, offspring.key.Length); if (i1 == i2)
goto lt;
int aux = offspring.key[i1];
offspring.key[i1] = offspring.key[i2];
```

```
offspring.key[i2] = aux;
}
offspring.Fitness = offspring.FitnessFunc();
newpopulation.Add(offspring);
if (offspring.Fitness > population[index].Fitness)
succeedNo++;
}
succeedRatio = (double)succeedNo / MU;
}
void Select_NewPopulation()
{
List<EncryptionKey> pool = new List<EncryptionKey>();
// /* Put Parents and Offspring into one `pool' */
pool.AddRange(population); pool.AddRange(newpopulation);
```

```
// Sort the pool. pool.Sort(); population.Clear(); // Select The
best MU from the pool
population.AddRange(pool/*.Distinct<Point>().ToList()*/.GetRange(0, MU));
}
///             <summary>
///             Main function: Each generation involves selecting the best
///             members, performing mutation then evaluating the resulting
///             population, until the terminating condition is satisfied.
///             </summary>

public void run()
{
int generation = 0;
int lastimprovment = 0;
double bestFitness = population[0].Fitness;
while (generation < MAXGenerations)
{
generation++;
mutate();
/* Select offspring through tournaments */
Select_NewPopulation();
if (population[0].Fitness > bestFitness)
{
lastimprovment = generation;
bestFitness = population[0].Fitness;
}
if (generation % 5 == 0)
{
if (succeedRatio > 0.2)
{
Population.Gama /= 0.85;
```

```
}
else
{
Population.Gama *= 0.85;
}
}
}
population.Sort();
}
public void runOneEpoch()
{
mutate();
/* Select offspring through tournaments */
Select_NewPopulation();
}
}
}
```

ÖZGEÇMİŞ

Abdihalim Adam ABDIRAHMAN, 10.10.1988'de Somali'de doğdu. İlk, orta ve lise eğitimini Somali'de tamamladı. 2007 yılında Ilays Lisesi'nden mezun oldu. 2007 yılında başladığı Nugaal Üniversitesi Bilgisayar Mühendisliği Bölümü'nü 2011 yılında bitirdi. 2012 ve 2014'te kadar Nugaal Üniversitesi'nde araştırma görevlisi olarak çalışmaya başladı. 2014 yılında Sakarya Üniversitesi yüksek lisans başladı.