

Genetik Algoritma Desteđi İin Bir Java Kütüphanesi : JGAP

Elem Güzel¹, Diyar Saraođlu²

¹ Dokuz Eylül Üniversitesi, Bilgisayar Mühendisliđi Bölümü, İzmir

² Ege Üniversitesi, Uluslararası Bilgisayar Enstitüsü, İzmir

elemguzel@gmail.com, diyarsaracoglu@gmail.com

Özet: Genetik algoritmalar evrimsel hesaplama yöntemlerinden biridir. Süreksiz, zamanla deđişen problemlerin en uygun çözümlerini bulmada, özellikle arama problemlerinin çözümlerinde ve var olan çözümlerin iyileştirilmesinde etkin bir şekilde kullanılır. Genetik algoritmalar kullanılarak çözümlerin gerçekleştirilmesinde araç ve kütüphane kullanılması işleri oldukça kolaylaştırmaktadır. Bu çalışmada nesne yönelimli bir programlama dili olan Java, ortam bağımsızlığı ve hızlı yazılım geliştirilebilmesi nedeniyle tercih edilmiştir. Nesne yönelimli bir genetik algoritma kütüphanesi olan JGAP incelenmiş, örnek olarak bir doğrusal Diophantine eşitliđi ve bir simetrik gezgin satıcı problemi çözülmüştür.

Anahtar Sözcükler: Genetik Algoritmalar, Eniyileme, JGAP, Diophantine, Gezgin Satıcı Problemi

A Java Library For Genetic Algorithms Support: JGAP

Abstract: Genetic algorithms, a method of evolutionary computation, is used efficiently to find optimum solutions of discontinuous problems that changes in time, especially solving search problems and optimization of existing solutions. When implementing genetic algorithms for solving problems, using tools and libraries will ease our work. In this study an object oriented programming language Java was preferred, because of platform independence and fast software development. An object oriented genetic algorithm library JGAP explored and, a linear Diophantine equivalence problem and a symmetric travelling salesman problem solved as examples.

Keywords: Genetic Algorithms, Optimization, JGAP, Diophantine, Travelling Salesman Problem

1. Giriş

Evrimsel süreçle birlikte tüm canlılar, çevresel koşullara uyum sağlayarak hayatta kalmayı başarmıştır. Evrimin bilgisayarlı ve matematiksel simülasyonlarının geleneksel optimizasyon süreçlerine yeni yaklaşımlar getirebileceđi düşünölmüştür. Temel düşünce şudur ki; nasıl ki canlılar deđişen ortam şartlarına mükemmel uyum sağlayabilmek için mutasyon, çaprazlama gibi yöntemlerle deđişiyorsa problem çözümleri de ideal çözüm olabilmek amacıyla aynı yöntemlerle deđişebilir [5]. Evrimsel algoritmalar; uygunluk fonksiyonuyla kontrol edilen çözüme en yakın adaya sahip olmak için,

öncelikle çözüm adayları popölasyonu oluşturur ve bu popölasyonu en uygun adayı bulana kadar evrimleştirir. Her adayın uygunluk deđerleri hesaplanır ve deđerlendirilir, böylelikle yeni nesli oluşturacak atalar ve elenecek bireyler belirlenir[5].

Darwin'in evrim teorisinden yola çıkarak geliştirilen genetik algoritmalar, evrimsel hesaplama yöntemlerinden biridir.

Java; nesne yönelimli, ortamdandır bağımsız yüksek seviyeli bir dildir[10]. Bu çalışmada Java ile geliştirilmiş nesne yönelimli genetik

algoritma kütüphanesi JGAP (“Java Genetic Algorithms Package”) kütüphanesi incelenmiştir. Bu amaçla ikinci bölümde kısaca genetik algoritmalar, tarihsel gelişimi, işleyişi ve genetik algoritma operatörleri incelenmiş; üçüncü bölümde çeşitli genetik algoritma kütüphaneleri ve JGAP incelenmiş; dördüncü bölümde ise JGAP kütüphanesi kullanılmasına örnek olarak $a+2b+3c+4d=50$ doğrusal Diophantine eşitliği ve simetrik gezgin satıcı problemlerinden Berlin52¹ problemi çözülmüştür.

2. Genetik Algoritmalar

Evrimsel hesaplama yöntemlerinden biri olan genetik algoritmalar Gregor Mendel'in genetik yasalarından ve Charles Darwin'in evrim teorisinden esinlenerek geliştirilmiştir. En kaba hatlarıyla belirtecek olursak evrimsel bir süreç takip edilerek, çözümün evrimleşmesiyle arama uzayı içerisinde en uygun çözüme erişmeye çalışılmaktadır [3].

Evrimsel hesaplama, 1960'lı yıllarda Ingo Rechenberg'in “Evrimsel Stratejileri” (“Evolution Strategies”) adlı eserinde ilk olarak ortaya atılmıştır. Genetik algoritmalar ise Michigan Üniversitesi'nde John Holland tarafından bulunmuştur. Daha sonra Holland'ın öğrencileri tarafından geliştirilmiştir. 1985 yılında Holland'ın doktora öğrencisi olarak tezini veren David E. Goldberg 'in gaz boru hatlarının döşenmesi ile ilgili çalışmasını duyurulmasıyla Genetik Algoritmaların sadece uygulaması yapılamayan bir araştırma konusu olmadığı ortaya çıkmıştır [6]. İlerleyen zamanlarda Genetik Algoritmaların kullanımı iyice yaygınlaştı ve iyileştirme problemlerinin çözümünde tercih edilen yöntemlerden biri olmuştur.

Genetik algoritmalarda her çözüm kümesi kromozomlarla temsil edilir. Bu temsiliyet çeşitli şekillerde olabilir. Her bir problemde

kullanılan kromozom kodlama yöntemleri probleme bağlı olarak farklılık gösterir. Kromozom kodlama yöntemlerinin arasında en yaygın olanı ikili karakter dizisi şeklinde kodlamadır. Bunun dışında tam sayı veya gerçek sayı şeklinde kodlamalar da yaygın olarak görülür.

Bir toplumdaki çözümler çeşitli işlemlerden geçerek yeni toplumların üretilmesini sağlarlar. Bu işlemler daha iyi çözümleri elde etmek için gerçekleştirilir. Yeni nesili oluşturacak çözümleri elde etmek için kullanılacak çözümler uygunluk fonksiyonu kullanılarak seçilir. Eğer bu çözümler daha uygun ise yeni neslin oluşturulmasında kullanılır. Bu süreç en iyiyi bulana kadar devam eder.

Genetik algoritmalarda en uygun çözümü bulma çaprazlama ve mutasyon gibi işlemlerden oldukça etkilenir. Çaprazlama yavruyu oluşturacak ataların üzerinde bir kesme noktası belirlenerek kesme noktasının solundaki herşeyi birinci atadan sağındaki herşeyi ise ikinci atadan alma işlemidir. Aslında çaprazlamanın farklı yolları mevcuttur. Birden fazla kesme noktası belirleyip kromozomların kodlanma şekline göre daha karmaşık çaprazlama işlemleri gerçekleştirmek mümkündür.

Mutasyon işlemi ile çaprazlanan yavru rastgele değiştirilir. Kromozomların kodlanma yöntemine göre mutasyon işlemi değişiklik gösterebilir.

Genetik algoritmaların işleyişine değinecek olursak basit bir genetik programlama taslağı şu şekilde olur :

- 1. Başlangıç:** n kromozom oluşan rastgele toplum oluşturulur (problemin olası çözümleri)
- 2. Uygunluk:** Toplumdaki her x kromozomu için $f(x)$ uygunluk değeri değerlendirilir.
- 3. Yeni Toplum:** Aşağıdaki adımlar izlenerek yeni toplum üretilir;
a.Seçim: Toplumdan

1 <http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/>

uygunluklarına göre iki ata seçilir (daha uygun olanın seçilme şansı daha fazladır)

b.Çaprazlama: Çaprazlama olasılığı ile ataları yeni yavru oluşturmak için birbirleriyle eşleştirilir. Eğer çaprazlama yapılmazsa, yavru ataların tıpatıp aynısı olacaktır.

c.Mutasyon: Mutasyon olasılığı ile yeni yavru üzerinde her yörünge için mutasyon işlemi yapılacaktır.

d.Kabul: Yeni yavru, yeni topluma eklenir.

4. Değiştir: Yeni toplum algoritmanın tekrar işlenmesinde kullanılır.

5. Dene: Eğer bitiş durumu sağlandıysa, durup toplumdaki en iyi çözüm döndürülür.

6. Döngü: Adım 2'ye gidilir[4,8].

Genetik algoritmaların avantajlarından biri problem hakkında hiçbir bilgiye sahip olmadan problemi çözebilmesidir. Bir diğeri ise uygunluk fonksiyonunun süresiz, zamanla değişen bir fonksiyon olduğu veya birden çok yerel optimum değeri olan problemlerde iyi performans göstermesidir. Ayrıca arama uzayında birden fazla kromozom olduğu için paralel çalışmaya uygundur [9].

3. Genetik Algoritma Kütüphaneleri ve JGAP

Genetik algoritmalar ile problem çözümlerini gerçekleştirirken benzer işlemler kullanılır. Her problem çözümünde bu işlemleri yeniden yazmak yerine hazır olanlarını kullanmak ve iyileştirmek, geliştiriciler için problem çözümlerini kolaylaştıracaktır. Günümüzde farklı programlama dillerinde yazılmış birçok hazır genetik algoritma kütüphanesi bulunmaktadır.

3.1 Genetik Algoritma Kütüphaneleri

GAlib(Genetic Algorithms Library)² : C++ dilinde geliştirilen, eniyileme işlemleri için hazır fonksiyonlar bulunduran bir genetik

algoritma kütüphanesidir. Kütüphane içerisinde genetik algoritmalarla çözülmüş örnek problemleri de barındırmaktadır.

GAUL (Genetic Algorithm Utility Library)³ : C dilinde gerçekleştirimi yapılmıştır. Kütüphane paralel ve seri evrimsel algoritmaların çözümlerinin gerçekleştirilmesine olanak verir. Darwin'in, Lamarck'ın ve Baldwin'in evrimsel tasarımlarından yararlanılmıştır. Kütüphane içerisinde tepe tırmanma, benzetimli tavlama gibi bir çok yerel minimum algoritması hazır olarak bulunmaktadır.

JAGA (Java API for Genetic Algorithms)⁴ : Her tür genetik algoritma ve genetik programlama uygulamasını hızlı ve kolay bir biçimde gerçekleştirmeyi sağlayan bir Java uygulama geliştirmearayüzüdür. Çok sayıda hazır GA/GP kodlarını, metotlarını ve operatörlerini içermektedir.

GA Playground⁵ : Gerçekleştirimi Java'da yapılmıştır. Nesne yönelimli yaklaşımlar genetik algoritma çözümlerinde uygulanmıştır. Yerel minimumları aşabilmek için de özellikler eklenmiştir. Kütüphane içerisinde grafik arayüzü bulunmaktadır. Böylece TSP gibi problemlerin görselleştirilmesine olanak sağlamaktadır. Kütüphaneye applet uygulamalarından da erişim sağlamak mümkündür.

EO Evolutionary Computation Framework⁶: ANSİ-C++ dilinde gerçekleştirimi yapılmıştır. İçerisinde örnek problem çözümleri bulunmaktadır. Genetik algoritmanın, uygunluk fonksiyonuna ya da nesil sayısına göre sona erdirmeye kriterleri bulunmaktadır

Pyevolve⁷ : Python dili ile gerçekleştirimi yapılmıştır. Genetik algoritma çözümlerinde uygunluk fonksiyonun değişim grafiklerini gösterebilecek araçlara sahiptir. Problem çözümlerinin hızlı yapılmasına olanak verir.

3 <http://gaul.sourceforge.net/>

4 <http://www.jaga.org/>

5 <http://www.aridolan.com/ga/gaa/gaa.html>

6 <http://eodev.sourceforge.net/>

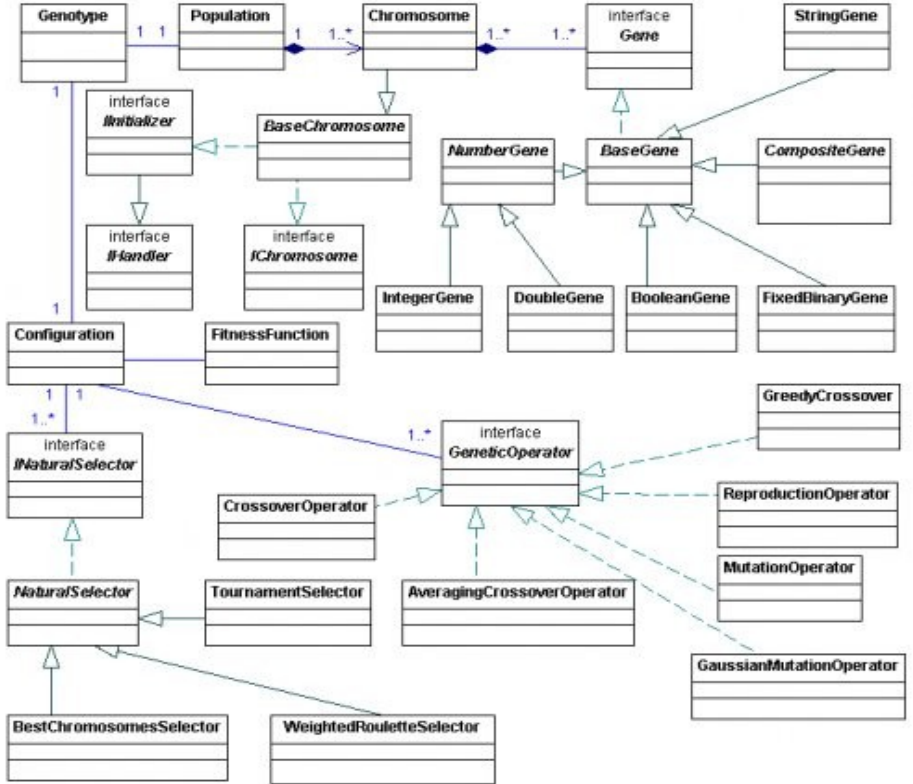
7 <http://pyevolve.sourceforge.net/>

2 <http://lancet.mit.edu/ga/>

İki boyutlu kromozom gerçekleştirmelerine olanak vermektedir.

3.2. JGAP (Java Genetic Algorithms Package)

çözümleri bulunmaktadır. Bu da kromozomların içerisindeki gen sayısını ve genler içinde temsil edilecek bilgiyi belirlemek anlamına gelmektedir. Problem tipine göre genler içerisinde temsil edilecek



Resim 1: JGAP'da önemli genetik programlama yapılarının sınıf diyagramı (V3.1)

JGAP⁸ Java dili ile geliştirilmiş bir genetik algoritma paketidir. Paket geliştirimi devam etmektedir.

JGAP'da nesne yönelimli yaklaşım genetik algoritmalara uyarlanmıştır.

Problem Çözümü

Temelde genetik algoritmalar kullanılarak problem çözümü 5 adımda gerçekleştirilmektedir; **Kromozom**

Yapılandırılması : İlk olarak kromozom yapısını düzenlememiz gerekmektedir. Kromozom içerisinde problemin olası

bilgiler değişebilmektedir.

1. Uygunluk Fonksiyonunun Uyarlanması :

Genetik algoritmalarla çok sayıda problemin çözülebilmesi sebebiyle uygunluk fonksiyonları da çeşitlilik göstermektedir. JGAP içerisinde bazı problemler için hazır uygunluk fonksiyonları bulunmaktadır. Diğer problemler için ise kendi uygunluk fonksiyonlarının yazılması gerekmektedir. Hazırlanan uygunluk fonksiyonu sınıfına

8 <http://jgap.sourceforge.net>

org.jgap.FitnessFunction çağrımını ekleyerek JGAP'ın sağladığı özelliklerden yararlanabilmek mümkündür.

2. **Konfigürasyon Nesnesinin Ayarlanması** : JGAP'da genetik algoritma işleçlerinin yapılandırılması konfigürasyon aracılığıyla yapılmaktadır. Konfigürasyon sınıfından varsayılan özelliklere sahip bir konfigürasyon nesnesi oluşturulabileceği gibi istenilen özelliklere sahip konfigürasyon nesnesi de yaratılabilir. Yaratılan nesnenin seçim, mutasyon, çaprazlama ve diğer işleçlerini JGAP içerisinde hazır bulunan işleçlerden uygun olanı seçilerek verilebilir ya da yeni işleçler yazılabilir. Paket içerisinde birçok işleç hazır olarak bulunmaktadır.
3. **İstenilen Çözümler İçin Populasyonların Yaratılması** : Çözümek istenilen problem türüne göre oluşturulacak populasyon büyüklüğü değişebilmektedir. Önceden belirlenen populasyon büyüklüğüne göre populasyon yaratılabilir ya da JGAP'ın rastgele populasyon oluşturma özelliklerinden yararlanılabilir. Burada ilgili metoda parametre olarak oluşturulan konfigürasyon nesnesinin verilmesi gerekmektedir.
4. **Populasyonların Evrimleştirilmesi** : Bütün ilgili ayarlamalar yapıldıktan sonra populasyonların evrimleştirilmesi sağlanabilir. Populasyonların evrimi önceden belirlenen evrim sayısına, zaman süresine göre ya da uygunluk fonksiyonunun durumuna göre sonlandırılabilir.

birinin değişmesi problemlere yol açabilmektedir. JGAP içerisinde “supergenes” yapısı ile bu durumun önüne geçilmiştir. Bu yapı ile birbirleriyle sıkı ilişki içerisindeki genler, mutasyon gibi süreçlerde beraber değişebilmektedirler. Beraber işlem görebilmenin kontrolü, gen kümelerinin belirli diziler oluşmasıyla yapılmaktadır.

Kütüphane içerisinde genetik algoritmalarla çözülmüş bazı örnek problemler de yer almaktadır. Çözümü yapılmış problemlere örnek olarak Fibonacci Dizisi Problemi, Karınca Kuyruğu Problemi, Mona Lisa Problemi ve Robocode verilebilir. Özellikle Robocode örneği ilgi çekicidir. Robocode projesinin sonradan özel bir proje olarak geliştirilmesine devam edilmiştir. Geliştirilen birçok projede genetik algoritma gerçekleştirmeleri JGAP kullanılarak yapılmıştır. Bunlar arasında insan kaynakları yönetimi projeleri, bilgisayar ağlarına sızmaların tespiti, fiber optik ağların eniyileme problemleri, veritabanı optimizasyonu projelerini saymak mümkündür. JOONEGAP projesinde JGAP kullanılarak sinir ağları gerçekleştirimine olanak veren JOONE paketinde eniyileme yapılarak melez bir sistem gerçekleştirilmiştir.

4. JGAP Kullanım Örnekleri

4.1. Diophantine Eşitliği

Diophantine ismi Helen dönemi matematikçilerinden Diaphantus'dan gelmektedir. Diaphantus çözümü tamsayılar kümesinde aranan basit belirsiz polinom denklemleri üzerinde çalışmıştır. Daha sonraları Baudhayana, Apastamba gibi Hindistanlı matematikçilerinde benzer eşitlikler üzerinde çalıştığı varsayılmaktadır. Fransız matematikçi Pierre de Fermat diophantine eşitlikleri üzerinde çalışmıştır. Euler Fermat'ın çözüm sağlayamadığı birçok eşitliği çözmüştür.

Matematik dünyasından birçok insanın çözüm bulmaya çalıştığı Fermat'ın son

Birbirleriyle ilişkili olan genlerden sadece

teoremi ise Andrew Wiles tarafından 1994 yılında çözülmüştür.

Diophantine eşitliklerinin lineer, üstel gibi bir çok değişik varyasyonu bulunmaktadır.

$a+2b+3c+4d=50$ lineer diophantine eşitliğinin çözümü genetik algoritmalar ile JGAP paketi kullanılarak gerçekleştirilmiştir.

Diophantine Eşitliği Çözümü : Problemin çözüm adımları;

Kromozom Yapılandırılması : İlk olarak genler oluşturulmuştur. Genlerin 0 ile 10 arasında rastgele değerler alması rastgele sayı üretici kullanarak gerçekleştirilmiştir. Belirtilen problemde 4 adet değişken olduğundan 4 adet gen oluşturularak örnek kromozom yapısına verilmiştir. Böylece 4 gene sahip kromozomlar oluşturulmuştur.

Uygunluk Fonksiyonunun Uyarlanması : Genlerin aler değerleri alınarak ilgili katsayı değerleri çarpılmıştır. Çarpımı gerçekleştirilen bu değerler toplanarak beklenen değer olan 50 sayısından çıkartılarak bulunan fark değeri minimize edilmeye çalışılmıştır.

Konfigürasyon Nesnesinin Ayarlanması : Varsayılan konfigürasyon tipinde bir konfigürasyon oluşturularak ilgili işleçler

İstenilen Çözümler İçin Populasyonların Yararlanması : Çeşitli boyutlarda populasyonlar oluşturulmuştur.(20, 30, 40)

Populasyonların Evrimleştirilmesi : Bir önceki adımda oluşturulan populasyon belirlenen evrim sayılarına göre evrimleştirilerek uygunluk fonksiyonunun ulaşması gereken değer elde edilmeye çalışılmıştır.

5. Sonuç

Genetik algoritmalar eniyileme problemleri gibi çözüm uzayının çok geniş olabileceği problemlerde iyi sonuçlar vermektedir. Diğer birçok yöntemle göre (istatistiksel vb) daha hızlı sonuçlar elde edebilmek böylece mümkün olmaktadır.

Genetik algoritma çözümleri için birçok kez farklı platformlarda yeni kütüphaneler, yeni paketler yazılmıştır. Yapılan bu çalışmalar problem çözümlerini kısa zamanda gerçekleştirebilmek için son derece kolaylık sağlamaktadır. Geliştirilen bazı paketler/kütüphaneler son derece gelişmiş olup bazıları ise sadece basit problem çözümleri için daha iyi performans göstermektedir.

Nesne yönelimli yaklaşımların, hazırlanan genetik algoritma kütüphanelerinde uygulanmasıyla problem çözümlerinin gerçekleştirilmesi değişik bir boyut kazanmıştır. JGAP'da son yıllarda geliştirilen

Seçilim	Evrimsayısı	PopulasyonBüyüklüğü	Mutasyon	Sonuç
Turnuva(20,1.0d)	1024	128	1/200	10089.466375376205
Turnuva(40,1.0d)	1024	128	1/200	9268.871154956929
Turnuva(100,1.0d)	512	128	1/200	8901.751244073472
Turnuva(100,1.0d)	1024	128	1/200	8291,02

Tablo 1: Elde edilen bazı sonuçlar

konfigürasyona verilmiştir. Verilen işleçlerin değişimine göre elde edilen sonuçların ve sonuç elde edim hızının değiştiği gözlemlenmiştir.

paketler içerisinde gelişmiş özellikleriyle ön plana çıkmaktadır.

JGAP içerisinde hazır bulunan işleçler ile yeni işleç yazılması gereksinimini minimuma

indirmiştir. Örnek problem çözümlerindeki çeşitlilik ise benzer problem çözümlerinin gerçekleştirimini oldukça kolaylaştırmıştır.

JGAP içerisinde yer alan bazı genlerin beraber evrim süreçlerine girebilme mantığı da genetik programlama için oldukça yeni bir yaklaşımdır.

JGAP paketi kullanılarak gerçekleştirimi yapılan iki problem aracılığıyla da çözüm aşamaları gözlemlenmiş olup, problem çözümleri için iyi sonuçlar elde edilmiştir.

Gezgin satıcı probleminin çözümünde bilinen en iyi sonuca yaklaşılmakla beraber en iyi sonuca ulaşamamıştır. En iyi sonuca ulaşabilmek için yerel minimum algoritmalarının kullanılması gerekmektedir.

6. Kaynakça

[1] Hordijk W., “An Introduction to Evolutionary Computation”

[2] Karaboğa D., 2004, ”Yapay Zeka Eniyileme Algoritmaları”

[3] Kalaycı T. Emre, 2006, “Yapay Zeka Teknikleri Kullanan Üç Boyutlu Grafik Yazılımları İçin 'Extensible 3D' (X3D) ile Bir Altyapı Oluşturulması Ve Gerçekleştirimi”

[4] Thomas A., 2001, “Solving The TSP Sales Man Problem using a Genetic Algorithm”

[5] Er H., Çetin M. K., İpekçi Çetin E., “Evolutionary Algorithmic Approaches in Finance : Applications of Genetic Algorithms”

[6] Saraçoğlu D., 2009, “Web Tabanlı Bir Genetik Algoritma Öğreticisi”

[7] Genetic Algorithms and Network Intrusion Detection-Marc McFadden

[8]“Genetic Algorithms”,
<http://www.obitko.com/tutorials/genetic-algorithms/>

[9] “Genetic Algorithms”
<http://www.cs.chalmers.se/Cs/Grundutb/Kurs/algsem/Projects2007/GeneticAlgorithms.pdf>

[10] “Learn About Java Technology”,
<http://java.com/en/about/>