МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ
(МИНОБРНАУКИ)

ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ «САНКТ-ПЕТЕРБУРГСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ
И ОПТИКИ»
(СПбГУ ИТМО)

УТВЕРЖДАЮ
Ректор СПбГУ ИТМО,
докт. техн. наук, профессор
В. Н. Васильев

____ _____ 2008 г.

ПРОГРАММНОЕ СРЕДСТВО 3GENETIC

ПРОГРАММНЫЙ МОДУЛЬ GA.ISLAND
ТЕКСТ ПРОГРАММЫ

ЛИСТ УТВЕРЖДЕНИЯ

7.190.00001-01 12 04-ЛУ

Инв.N подл.	Подп. и дата	Взам.инв.N	Инв.N дубл.	Подп. и дата

Декан факультета «Информационные
технологии и программирование»
докт. техн. наук, профессор
_____ В. Г. Парфенов

Руководитель темы
заведующий кафедрой «Технологии программирования»,
докт. техн. наук, профессор
_____ А. А. Шалыто

2008

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ
(МИНОБРНАУКИ)

ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ «САНКТ-ПЕТЕРБУРГСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ
И ОПТИКИ»
(СПбГУ ИТМО)

УТВЕРЖДЕНО
7.190.00001-01 12 04-ЛУ

ПРОГРАММНОЕ СРЕДСТВО 3GENETIC

ПРОГРАММНЫЙ МОДУЛЬ GA.ISLAND
ТЕКСТ ПРОГРАММЫ

7.190.00001-01 12 04

Листов 7

Инв. N подл.	Подп. и дата	Взам. инв. N	Инв. N дубл.	Подп. и дата

2008

АННОТАЦИЯ

В данном документе приводится текст модуля GA.Island программного средства 3GENETIC, реализующего отсровной вариант генетического алгоритма.

СОДЕРЖАНИЕ

1. МОДУЛЬ GA.ISLAND

Модуль GA.island является подключаемой частью программного средства 3GENETIC и представляет реализацию островного варианта генетического алгоритма. Исходный текст модуля хранится в 2-х файлах:

1. ga/simple/IslandGA.java – реализация классического генетического алгоритма;
2. ga/simple/IslandGALoader.java – загрузчик simpleGA.

1.1. Пакет ga.island

1.1.1. ga/island/IslandGA.java

```
package ga.island;

import ga.simple.SimpleGA;
import laboratory.common.ga.GA;
import laboratory.common.ga.Individual;
import laboratory.common.ga.IndividualFactory;

import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Random;

public class IslandGA implements GA {

   private final GA[] islands;

   private int indexGeneration;

   private final int numberSwap;

   private final int periodSwap;

   private final int sizeElite;

   private final int periodBigMutation;

   private final double percentKilledIslands;

   private Random r;

   public List<Individual> getGeneration() {
      List<Individual> gen = new ArrayList<Individual>();
      for (GA ga : islands) {
         gen.addAll(ga.getGeneration());
      }
      Collections.sort(gen);
      return gen;
   }

   public IslandGA(int numberIslands, int sizeIsland, double sizeElite, double probabilityMutations,
```

```
            IndividualFactory factory, double numberSwap, int periodSwap, int periodBigMutation,
            double percentKilledIslands) {
        indexGeneration = 0;
        islands = new GA[numberIslands];
        for (int i = 0; i < numberIslands; i++) {
            islands[i] = new SimpleGA(sizeIsland, sizeElite, probabilityMutations, factory);
        }
        this.numberSwap = (int) (numberSwap * sizeIsland);
        this.periodSwap = periodSwap;
        this.sizeElite = (int) (sizeElite * sizeIsland);
        this.periodBigMutation = periodBigMutation;
        this.percentKilledIslands = percentKilledIslands;
        r = new Random();
    }

    public void nextGeneration() {
        indexGeneration++;
        for (GA ga : islands) {
            ga.nextGeneration();
        }
        if (indexGeneration % periodSwap == 0) {
            int ni = islands.length;
            List<Individual>[] generation = new List[ni];
            for (int i = 0; i < ni; i++) {
                generation[i] = islands[i].getGeneration();
            }
            int size = generation[0].size();
            for (int i = 0; i < ni; i++) {
                for (int j = 0; j < numberSwap; j++) {
                    generation[i].set(size - j - 1, generation[r.nextInt(ni)].get(r.nextInt(sizeElite)));
                }
            }
            for (int i = 0; i < ni; i++) {
                Collections.sort(generation[i]);
            }
        } else if (indexGeneration % periodBigMutation == 0) {
            bigMutation();
        }
    }

    public void bigMutation() {
        for (GA ga : islands) {
            if (r.nextDouble() < percentKilledIslands) {
                ga.bigMutation();
            }
        }
    }

    public Individual getBest() {
        Individual best = islands[0].getBest();
        for (int i = 1; i < islands.length; i++) {
            Individual b = islands[i].getBest();
```

```
            if (b.fitness() > best.fitness()) {
                best = b;
            }
        }
        return best;
    }
}
```

1.1.2. ga/island/IslandGALoader.java

```
package ga.island;

import laboratory.common.Loader;
import laboratory.common.ga.GA;
import laboratory.common.ga.IndividualFactory;
import laboratory.util.Parser;

import java.io.IOException;
import java.util.Properties;
import java.util.jar.JarEntry;
import java.util.jar.JarFile;

public class IslandGALoader implements Loader<GA> {
    private final Parser properties;

    public IslandGALoader(JarFile file) {
        Properties in = new Properties();
        try {
            JarEntry ent = new JarEntry("islandGA.conf");
            in.load(file.getInputStream(ent));
        } catch (IOException e) {
            e.printStackTrace();
        }
        properties = new Parser(in);
    }

    public GA load(Object... args) {
        return new IslandGA(properties.getInt("numberIslands"), properties.getInt("sizeIsland"),
                properties.getDouble("sizeElite"), properties.getDouble("probabilityMutation"),
                (IndividualFactory) args[0], properties.getDouble("numberSwap"),
                    properties.getInt("periodSwap"),
                properties.getInt("periodBigMutation"), properties.getDouble("percentKillIsland"));
    }

    public Properties getProperties() {
        return properties.getProperties();
    }
}
```

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

Номера листов (страниц)					*Всего листов (страниц) в докум.*	*№ докумен-та*	*Входящий № сопрово-дительного докум. и дата*	*Подп.*	*Дата*
Изм.	*Измененных*	*Замененных*	*Новых*	*Аннулированных*					