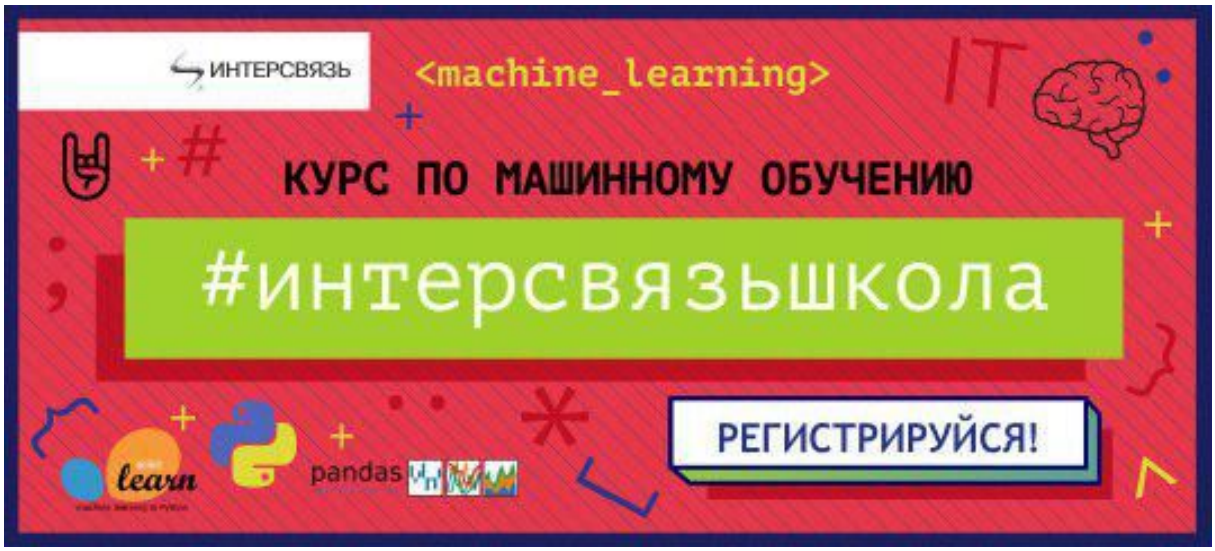

Отборочное задание

#ИнтерСвязьШкола, январь 2017

Отборочное задание состоит из трех задач: первой - основной - и двух дополнительных. Каждое задание проверяет различные навыки: умение разбираться в базовых вещах и навыки программирования; умение разбираться в более сложных вещах; навыки исследовательской деятельности. Основой для оценивания всего отборочного задания будет решение задачи №1, решения двух дополнительных задач - “бонусами”. Рекомендуется стремиться не к большому количеству решенных задач, а к качественному решению каждой задачи!

Решение всего отборочного задания необходимо предоставить в виде единого ZIP-архива, содержащего максимум три файла - по одному на каждую решенную задачи. В качестве имени файла с архивом следует использовать собственную фамилию - для легкости идентификации решений. Письмо с архивом необходимо отправить на адрес Школы is_school@intersvyaz.net, указав в тексте письма свое имя и фамилию.

При возникновении вопросов по отборочному заданию, задавайте их, пожалуйста, по электронной почте на адрес is_school@intersvyaz.net. Срок ответа - один рабочий день.

Задача №1 - “техническая”

Необходимо решить задачу о 8 ферзях с помощью генетического алгоритма. Следует использовать классический генетический алгоритм:

- Используется бинарное кодирование хромосом;
- Селекция особей выполняется с помощью “колеса рулетки”;
- Используется оператор одноточечного скрещивания.

При решении задачи требуется не только реализовать алгоритм, но и подобрать наилучшие параметры его работы (фитнес-функцию, размер популяции, величину вероятности скрещивания, величину вероятности мутации).

Требования к реализации

- Язык программирования Python 3;
- Следует стараться использовать только стандартные пакеты и модули языка;
- Решение необходимо оформить в виде модуля с именем “nqueens”, содержащего класс с именем “Solver_8_queens”, реализующий метод с именем “solve”;
- Проверяющая программа будет использовать класс Solver_8_queens аналогично скрипту solution.py:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
This script uses nqueens.py module for solving 8-queens problem by using
a genetic algorithm
"""
import sys
print('Python version:', sys.version)

import nqueens as nq

solver=nq.Solver_8_queens()
best_fit, epoch_num, visualization = solver.solve()
print("Best solution:")
print("Fitness:", best_fit)
print("Iterations:", epoch_num)
print(visualization)
```

- Конструктор класса Solver_8_queens должен быть определен с тремя аргументами:

```
class Solver_8_queens:

    def __init__(self, pop_size=100, cross_prob=0.50, mut_prob=0.25):
```

  - **pop_size** - размер популяции;
  - **cross_prob** - вероятность скрещивания (в интервале от 0 до 1);
  - **mut_prob** - вероятность мутации (в интервале от 0 до 1);
  - Для каждого аргумента необходимо подобрать значение по умолчанию таким образом, чтобы генетический алгоритм решал задачу наилучшим образом.

- Метод solve класса Solver_8_queens должен быть определен с двумя параметрами (критерии останова):

```
122
123     def solve(self, min_fitness=0.9, max_epochs=100):
124
```

  - **min_fitness** - пороговое значение фитнес-функции наиболее приспособленной особи в популяции, при превышении которого алгоритм останавливается;
  - **max_epochs** - максимальное количество эпох (итераций) генетического алгоритма, при превышении которого он останавливается;
  - Если оба аргумента имеют значения отличные от None, то алгоритм должен останавливаться при срабатывании любого из критериев останова;
  - Если только один аргумент имеет значение отличное от None, то алгоритм должен останавливаться только при срабатывании этого критерия.
- Метод solve класса Solver_8_queens должен возвращать кортеж (tuple) из трех значений:

```
11
12 solver=nq.Solver_8_queens()
13 best_fit, epoch_num, visualization = solver.solve()
14
```

  - **best_fit** - значение фитнес-функции для наиболее приспособленной особи в популяции;
  - **epoch_num** - количество эпох (итераций), прошедших до остановки генетического алгоритма;
  - **visualization** - текстовая строка, содержащая интерпретацию (визуализацию шахматной доски) решения, соответствующего наиболее приспособленной особи;
  - Следует придерживаться стиля отображения при котором пустые клетки шахматной доски отображаются символом “+”, а ферзи - символом “Q”:

```
Best solution:
Fitness: 1.0
Iterations: 15
++Q+++++
+++++Q++
+++Q++++
+Q++++++
+++++++Q
++++Q+++
++++++Q+
Q+++++++
```

- Решение задачи необходимо предоставить в виде единого Python-модуля, соответствующего перечисленным выше требованиям.

Критерий оценивания

При оценивании решения задачи №1 будет учитываться следующие критерии (в порядке убывания приоритета):

1. Умение работать с источниками информации, т.е. правильность понимания отличительных особенностей классического генетического алгоритма;
2. Корректность реализации классического генетического алгоритма;
3. Уверенность владения языком Python;
4. Общий стиль программирования;
5. Скорость работы программы.

Задача №2 - “творческая”

Необходимо реализовать модифицированный генетический алгоритм. Изменения могут коснуться:

- Способа кодирования хромосом, например, с помощью кода Грея;
- Способа селекции хромосом, например, заменить “колесо рулетки” на ранговую или турнирную селекцию;
- Генетического оператора скрещивания, например, заменить одноточечный оператор на многоточечный.

Требования к реализации

- Все изменения должны быть реализованы в рамках класса Solver_8_queens из задачи №1. Другими словами, класс Solver_8_queens должен стать гибким, приобрести возможность работы в различных режимах;
- Разрешается добавлять добавлять дополнительные аргументы в конструктор класса Solver_8_queens, а также в интерфейсный метод solve. При этом необходимо указывать значения по умолчанию в определении указанных методов;
- Решение задачи необходимо предоставить в виде единого Python-модуля с названием “nqueens_mod”, соответствующего требованиям задач №1 и №2.

Критерии оценивания

При оценивании решения задачи №2 будет учитываться следующие критерии (в порядке убывания приоритета):

1. Корректность реализации изменений в генетическом алгоритме;
2. Гибкость реализации и поведения класса Solver_8_queens для модифицированного генетического алгоритма;
3. Количество и сложность модификаций.

Задача №3 - “исследовательская”

Необходимо выполнить сравнительное тестирование скорости работы классического и модифицированного генетического алгоритма при решении задачи с большей размерностью: N=16, 32, …

Требования к решению

Решение задачи №3 необходимо предоставить в виде текстового отчета в произвольной форме, содержащего описание проведенных тестов, фактические данные в виде таблиц и графиков, а также собственные выводы относительно отличительных особенностей работы модифицированного генетического алгоритма в сравнении с классическим. Отчет должен быть предоставлен в виде единого PDF-файла.

Критерии оценивания

При оценивании решения задачи №3 будет учитываться следующие критерии (в порядке убывания приоритета):

1. Умение анализировать экспериментально полученные данные и делать выводы на их основе;
2. Количество и объем проведенных экспериментов;
3. Наглядность отчета.