

## Лабораторная работа 2

### Введение в эволюционные вычисления

Выполнил Домницкий Е.А. М4130

#### Цель работы

Целью данной лабораторной работы является получение студентом представления об возможностях применения эволюционных алгоритмов для решения различных классов задач и программных средств для их разработки.

#### Оборудование и программное обеспечение

- Java JDK версии 1.8 и выше
- Watchmaker framework версии 0.7.1

#### BitsCount

Суть задачи в том, чтобы оптимизировать сумму элементов битовой строки заданной величины (максимизировать, иными словами, получить в результате эволюции популяцию строк с максимальным числом единиц). Сходиться должно к строке целиком из единиц. Было проанализировано количество итераций (поколений), необходимых для достижения оптимального решения на строках длиной 20, 50 и 100. Результаты представлены в таблице ниже.

Размерность	Run 1	Run 2	Run 3	Run 4	Run 5	Среднее
20	38	30	121	18	42	49.8
50	2772	3108	2882	1331	4362	2891
100	10866284	4511672	7889341	3256778	4796123	6264039.6

#### Задача коммивояжёра

Необходимо минимизировать путь, проходящий хотя бы 1 раз через все узлы графа (точки на карте).

##### Стандартные параметры:

*[Evolution (pop: 300, gen: 100, elite: 3, Truncation Selection (50%))]*

*ROUTE: Vienna -> Berlin -> Helsinki -> Stockholm -> Copenhagen -> Amsterdam -> London -> Dublin -> Lisbon -> Madrid -> Paris -> Brussels -> Luxembourg -> Rome -> Athens -> Vienna*

*TOTAL DISTANCE: 10976.0km*

*(Search Time: 1.116 seconds)*

*[Evolution (pop: 500, gen: 100, elite: 3, Truncation Selection (50%))]*

#### Популяция 500:

*ROUTE: London -> Dublin -> Lisbon -> Madrid -> Rome -> Athens -> Vienna -> Berlin -> Helsinki -> Stockholm -> Copenhagen -> Amsterdam -> Brussels -> Luxembourg -> Paris -> London*

*TOTAL DISTANCE: 10494.0km*

*(Search Time: 0.093 seconds)*

#### Поколений 300:

*[Evolution (pop: 300, gen: 300, elite: 3, Truncation Selection (50%))]*

*ROUTE: Amsterdam -> Brussels -> Luxembourg -> Paris -> London -> Dublin -> Lisbon -> Madrid -> Rome -> Athens -> Vienna -> Berlin -> Helsinki -> Stockholm -> Copenhagen -> Amsterdam*

*TOTAL DISTANCE: 10494.0km*

*(Search Time: 0.145 seconds)*

#### Элитарных элементов 12:

*[Evolution (pop: 300, gen: 100, elite: 12, Truncation Selection (50%))]*

*ROUTE: Lisbon -> Dublin -> London -> Paris -> Luxembourg -> Brussels -> Amsterdam -> Copenhagen -> Stockholm -> Helsinki -> Berlin -> Vienna -> Athens -> Rome -> Madrid -> Lisbon*

*TOTAL DISTANCE: 10494.0km*

*(Search Time: 0.042 seconds)*

#### Популяция 500, поколений 300, элитарных элементов 12:

*[Evolution (pop: 500, gen: 300, elite: 12, Truncation Selection (50%))]*

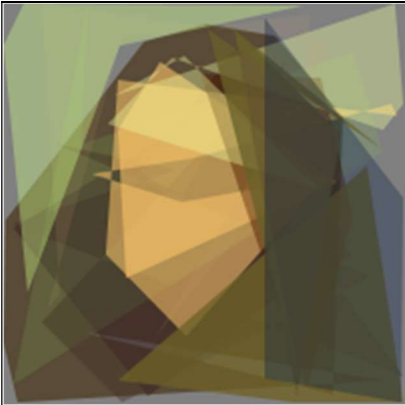


*ROUTE: Copenhagen -> Amsterdam -> Brussels -> Luxembourg -> Paris -> London -> Dublin -> Lisbon -> Madrid -> Rome -> Athens -> Vienna -> Berlin -> Helsinki -> Stockholm -> Copenhagen*

*TOTAL DISTANCE: 10494.0km*

*(Search Time: 0.189 seconds)*

## **Мона Лиза**

В этой задаче необходимо аппроксимировать картинку с Моной Лизой при помощи цветных полигонов разных цветов, размеров и поворотов.

Решение	Итерация	Фитнесс	Кол-во полигонов и углов	Рисунок
плохое	5467	333079	19 п, 90 у	
среднее	17789	233730	37 п, 195 у	
хорошее	18945	204574	25 п, 225 у	

### Ответы на вопросы:

- Типы структур решений:
  - Bits Counts — бинарный
  - Traveling salesman problem — комбинаторный
  - Mona Lisa — комбинаторный
- Решения в задачи коммивояжера закодированы в виде списка городов в определенном порядке (последовательность).
- В задаче Mona Lisa генотип — закодированное множество полигонов (список объектов ColouredPolygon), фенотип — отрисованное изображение