МИНОБРНАУКИ РОССИИ
Санкт-Петербургский государственный
электротехнический университет
«ЛЭТИ» им. В.И. Ульянова (Ленина)
Кафедра МО ЭВМ

ОТЧЕТ

по учебной практике (Вар. 14)

Тема: Реализация генетических алгоритмов с использованием GUI

Студент гр. 3388 _____ Беннер В.А.
Студент гр. 3388 _____ Сабалиров М.З.
Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2025

Оглавление

Глава 1

Введение

1.1 Условие задачи

Задача. Дана окружность радиуса R, необходимо в этой окружности расположить K квадратов одинаковых размеров, чтобы занимаемая ими площадь была максимальной. Квадраты не должны пересекаться.

Уточним исходный текст:

- $\mathbb{R}_+ \ni R > 0, K \in \mathbb{N}$.
- Стороны рассматриваемых квадратов $h \in \mathbb{R}_+$.
- Квадраты могут стоять вплотную.
- Основания квадратов параллельны оси абсцисс.

1.2 Стек технологий GUI

Для реализации графического интерфейса используются следующие python библиотеки:

1. PyQt6 – отрисовка интерфейса, обработка пользовательского ввода.
2. matplotlib – визуализация шагов решения и построение графиков.

1.3 Распределение обязанностей

	Беннер В.А	**Сабалиров М.З**
GUI	+	
Написание решения		+
Тестирование	+	+

Глава 2

Решение задачи

2.1 Целевая функция

$\exists R \in \mathbb{R}_+$ – радиус окружности;
Начало координат находится в центре окружности;
$[-R; R] \ni x$ – точка на оси абсцисс.

Лемма.
На хорде, перпендикулярной оси абсцисс и проходящей через точку $(x, 0)$, поместится ровно $f(x, h)$ целых отрезков длины $h \in \mathbb{R}, 0 < h < \sqrt{2}R$.

$$f(x, h) = \lfloor \frac{2 \cdot \sqrt{R^2 - x^2}}{h} \rfloor$$

$\exists \frac{\sqrt{4R^2-h^2}}{2} \leq r \leq R$
ρ – разбиение отрезка $[-r, R]$ с шагом h

Лемма.
На отрезке $\rho \ni [x_0, x_1]$ оси абсцисс в окружность поместится ровно $g(x_0, x_1, h)$ квадратов со стороной h.

$$g(x_0, x_1, h) = \begin{cases} f(x_0, h), & R - |x_0| \leq R - |x_1| \\ f(x_1, h), & R - |x_0| > R - |x_1| \end{cases}$$

$\exists K \in \mathbb{N}$ – необходимое количество квадратов;

Определение. Нижним псевдо-интегралом Дарбу назовем:

$$M_*(h, \rho) = \sum_{[x_0, x_1] \in \rho} g(x_0, x_1, h)$$

Определение. Целевой функцией назовем:

$$M(h,\rho) = \begin{cases} h^2K, & M_*(h,\rho) \geq K \\ 0, & M_*(h,\rho) < K \end{cases}$$

Заметим, что ген ρ представим в виде вещественного числа (первая точка разбиения). Таким образом, представители популяции являются носителями двух вещественнозначных генов. Задача сводится к максимизации целевой функции при заданных условиях.

2.2 Изменчивость

❑ Ген t у особи p мутирует с некоторой вероятностью P_m

Определение. Значение t после мутации t^* определяется, как нормальная случайная величина с средним t и стандартным отклонением σ, ограниченная допустимыми значениями гена

❑ Особь p становится родителем с некоторой вероятностью P_c

Определение. ❑ Выбраны два родителя $p_1, p_2; \alpha \geq 0$, тогда интервалы допустимых значений генов для их потомков определяются следующим образом:

$$T_t = [\max\{t_{\min}, p_{1_t} - \alpha(p_{2_t} - p_{1_t})\}; \min\{t_{\max}, p_{2_t} + \alpha(p_{2_t} - p_{1_t})\}]$$

Определение. Потомок определяется, как $p(p_1,p_2) := (h \in T_h(p_1,p_2), \rho \in T_\rho(p_1,p_2))$ (гены выбираются случайно).

Алгоритм. ❑ Есть список особей из популяции, которые становятся родителями, тогда родители попарно скрещиваются, и из каждой пары в популяцию добавляется $c \in \mathbb{N}$ потомков.

2.3 Отбор

❑N – размер популяции в каждом новом поколении.
После скрещивания и мутаций особи ранжируются по значению целевой функции. В новое поколение проходит $N^* = \lfloor nN \rfloor (0 \leq n \leq 1)$ лучших особей. Далее происходит турнирный отбор $N - N^*$ особей (по $m \in \mathbb{N}$ представителей в раунде). N отобранных особей выживают, остальные погибают.

Начальная популяция формируется из комбинаций $\lfloor \sqrt{N} \rfloor$ равномерно распределенных значений h и ρ, итого $\approx N$ особей в популяции, недостающие особи выбираются

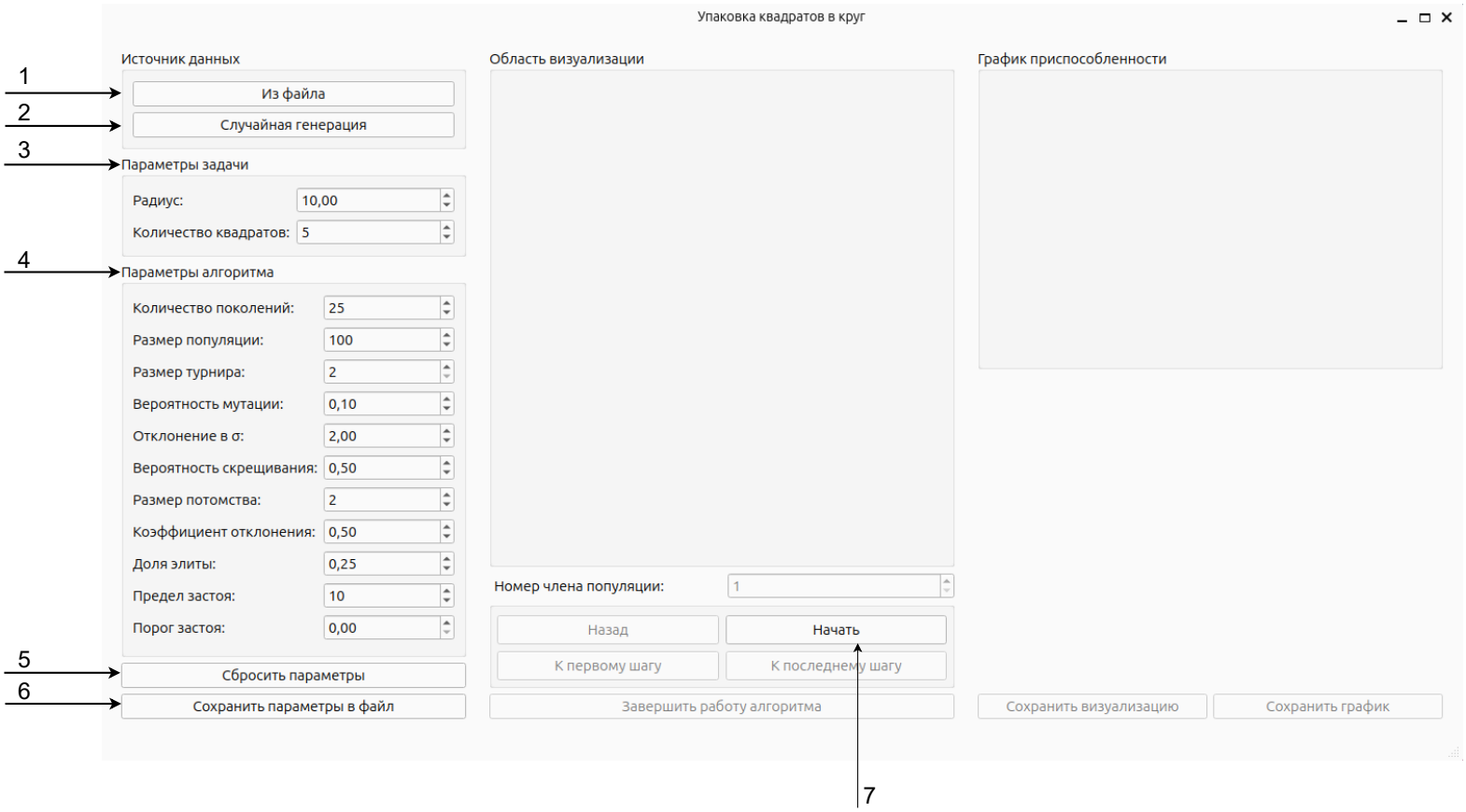
случайными допустимыми значениями генов.

Смена поколений происходит следующим образом: скрещивание $\rightarrow$ мутация $\rightarrow$ отбор. Алгоритм совершает $D \in \mathbb{N}$ итераций (D поколений), но если на протяжении $E \in \mathbb{N}$ поколений наилучшее значение целевой функции изменяется менее чем на $\varepsilon \geq 0$, то алгоритм досрочно завершает свою работу.

Глава 3

Описание GUI

Интерфейс при запуске программы:

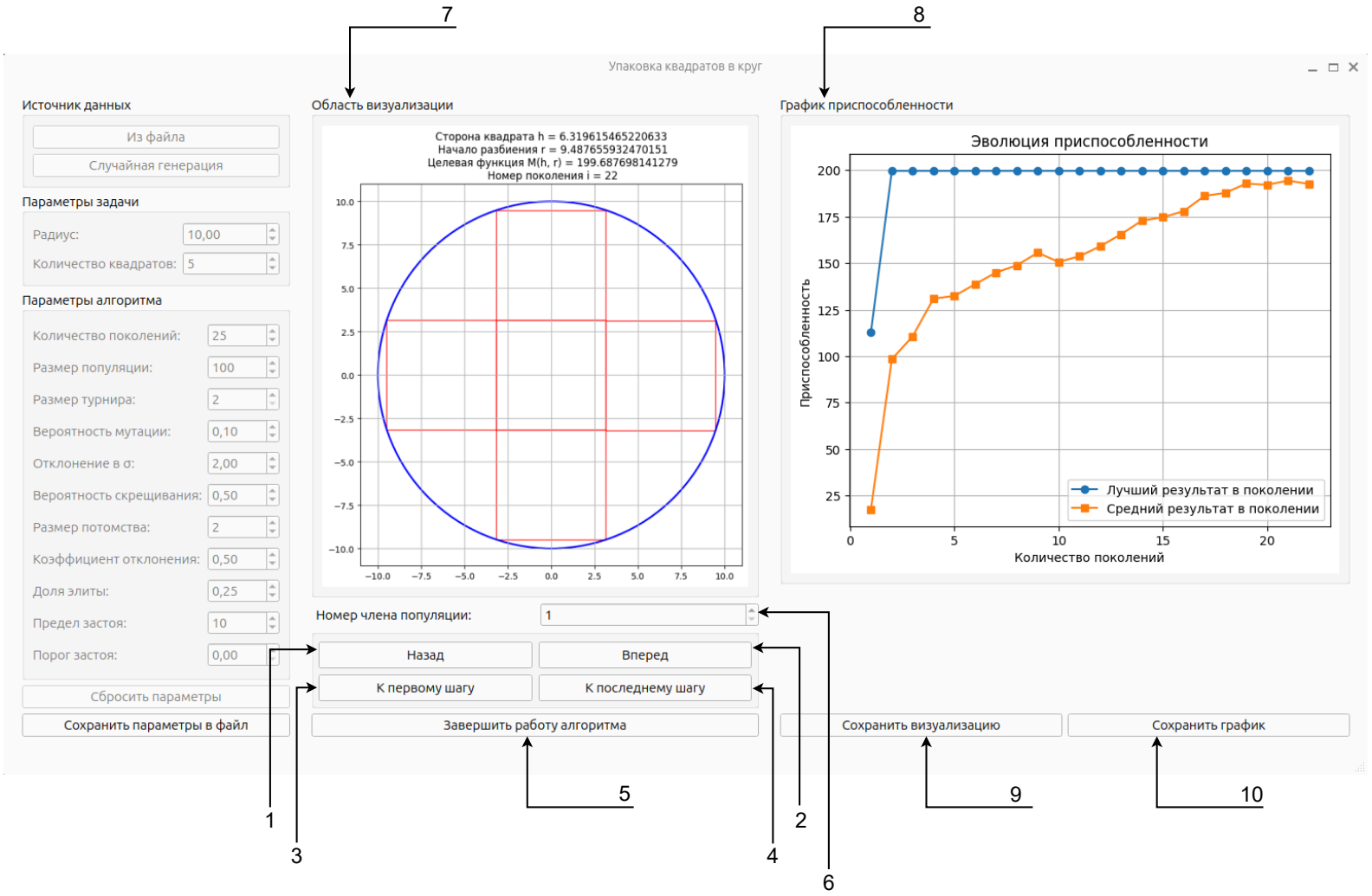


1. Позволяет загрузить параметры задачи и алгоритма из json-файла.
2. Случайно генерирует параметры в рамках допустимых значений.
3. Ввод начальных условий задачи.
4. Ввод параметров эволюции из предыдущей главы:

- Размер турнира – количество особей "сражающихся" за право перейти в следующие поколение.
- Вероятность мутации – вероятность мутации для всех генов.
- Отклонение в σ – на сколько σ-м значение мутированного гена может отклонятся при нормальном распределении.
- Вероятность скрещивания – вероятность того, что особь станет родителем.
- Коэффициент отклонение – на сколько потомок может отклонятся от родительского интервала (α в предыдущей главе).
- Доля элиты – какая часть лучших представителей популяции гарантировано переходит в следующие поколение.
- Предел застоя – максимальное количество поколений, в течении которых наилучшии особи отличаются менее чем на порог застоя. Если предел превышен, то алгоритм заканчивает свою работу.

5. Сбрасывает все параметры к начальным.
6. Сохраняет текущие параметры в json-файл.
7. Запускает алгоритм с текущими параметрами. Область изменения параметров становится неактивной.

После запуска алгоритма интерфейс выглядит следующим образом:



1. Отправляет на предыдущий шаг алгоритма.
2. Отправляет на следующий шаг алгоритма.
3. Отправляет на первый шаг алгоритма.
4. Отправляет на последний шаг алгоритма.
5. Завершает работу алгоритма, интерфейс переходит к начальному состоянию.
6. Номер члена популяции в текущем поколении (1-нумерация, популяция отсортирована по убыванию значений целевой функции).
7. Отрисовка члена популяции под указанным номером в текущем поколении.
8. График максимального и среднего значения целевой функции от первой до текущей популяции.

9. Локально сохраняет текущую визуализация члена популяции.
10. Локально сохраняет текущий график приспособленности.

Выводы

В ходе работы были изучены основные принципы построения генетических алгоритмов, а также разработка приложений с GUI.

Успешно реализован генетический алгоритм для решения задачи упаковки квадратов в круг, включающий селекцию, скрещивание, мутацию и элитизм. Алгоритм справляется с поставленной задачей, большое количество параметров, позволяет производить тонкую настройку эволюции, для улучшения результата.

Программа обладает гибкостью благодаря настраиваемым параметрам, поддержке загрузки и сохранения настроек, а также случайной генерации, что делает её удобным инструментом для исследований. Графический интерфейс на Qt6 обеспечивает интерактивность, включая пошаговое выполнение, визуализацию упаковки и график приспособленности.

Для дальнейшего улучшения можно ввести зависимость вероятности скрещивания от целевой функции, а также усовершенствовать стратегии селекции.