Генетический алгоритм

Генетический алгоритм. Основные моменты

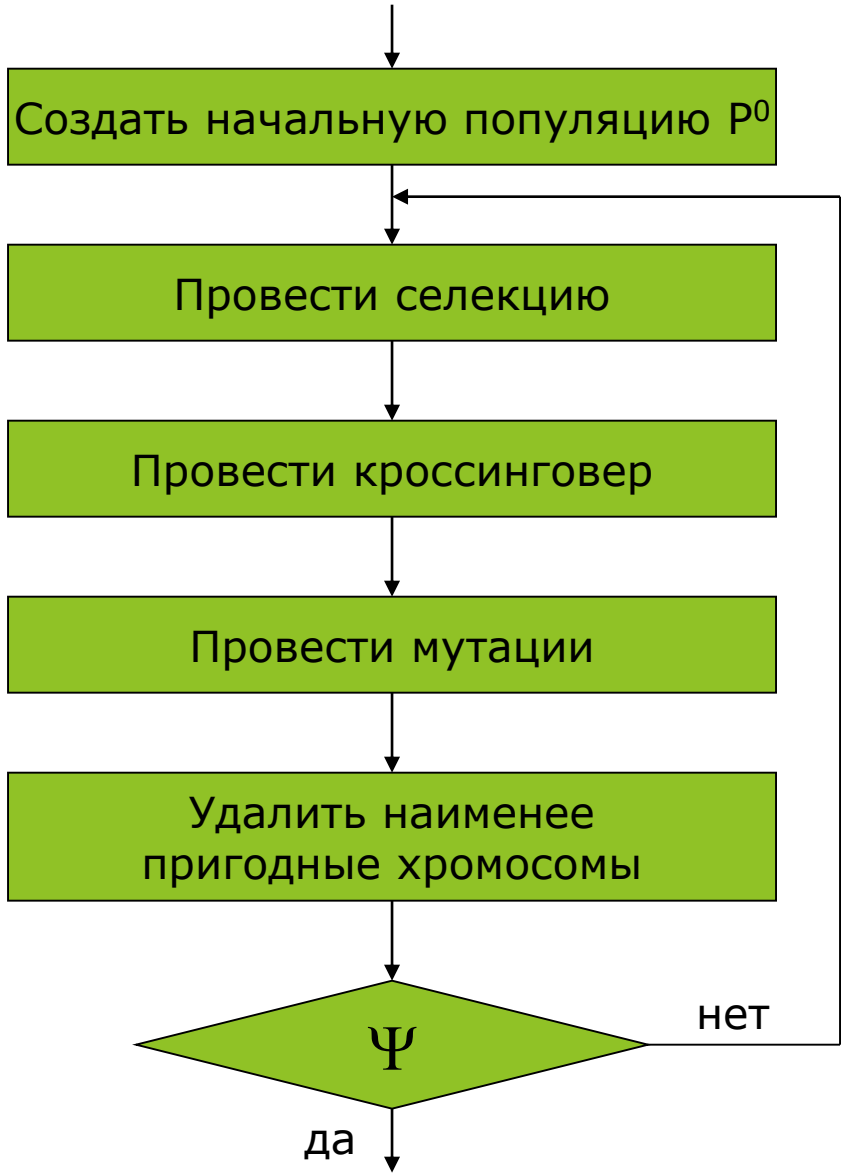
Задача формализуется таким образом, чтобы её решение могло быть закодировано в виде вектора («генотипа») генов.

Хромосома обычно представляет собой битовую строку, в которой закодированы все параметры решения.

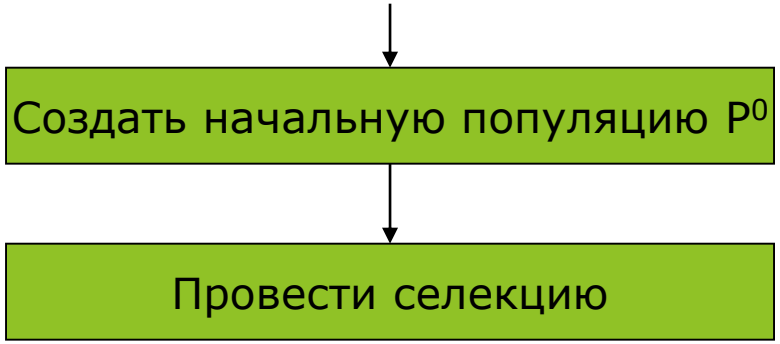
Популяция – набор решений (*хромосом*), рассматриваемых алгоритмом в конкретный момент.

Для каждого решения вычисляется функция полезности (*fitness*), которая говорит, насколько хорошо это решение удовлетворяет задаче.

Блок-схема работы генетического алгоритма



Блок-схема работы генетического алгоритма

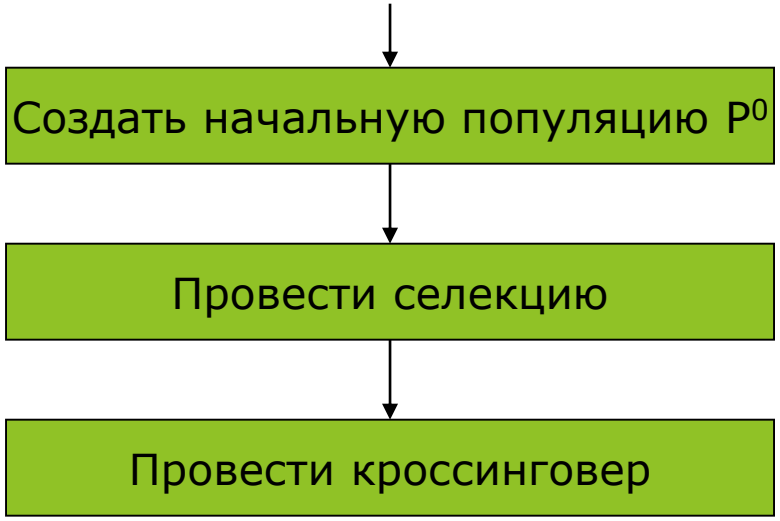


Из всей популяции выбираем определённую долю, которая останется «в живых» на этом этапе эволюции, причём вероятность выживания особи должна зависеть от значения функции приспособленности (функции полезности) **fitness**.

Про функцию fintess

- Задаёт цель оптимизации.
- Позволяет оценить степень приспособленности конкретных особей в популяции и выбрать из них наиболее приспособленных.
- Желательно, чтобы она требовала минимум ресурсов.

Блок-схема работы генетического алгоритма



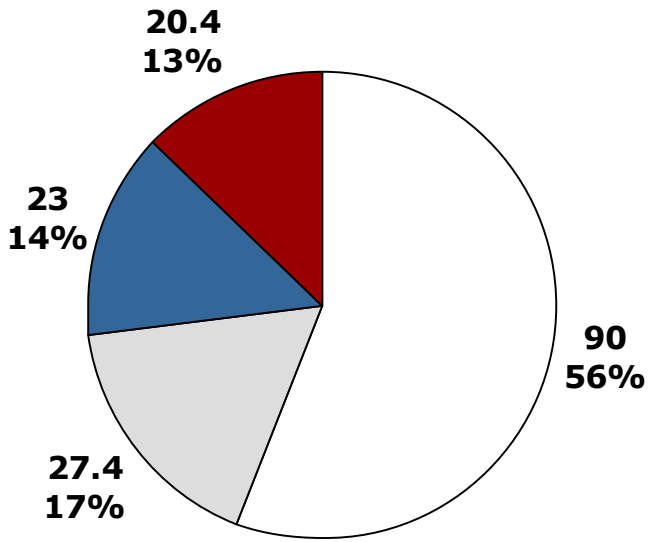
Кроссинговер (кроссовер) – скрещивание хромосом.

Главное требование к размножению — чтобы потомок или потомки имели возможность унаследовать черты обоих родителей, «смешав» их каким-либо способом.

Выбор хромосом для кроссинговера (выбор родителей)

Принцип Рулетки Монте-Карло: вероятность выбора особи тем вероятнее, чем лучше её значение функции приспособленности.

$$p_i = \frac{f_i}{\sum_{i=1}^{N} f_i}$$



Выбор хромосом для кроссинговера (выбор родителей)

Метод ранжинования (линейная нормализация): хромосомы сортируются по лучшему значению функции fitness, а затем допускаются к кроссинговеру с вероятностью, пропорциональной позиции в списке.

Таким образом, вероятность выбора зависит *от места в списке особей*, отсортированном по значению fitness.

$p_i = \frac{1}{N}\left(a - \frac{(a-b)(i-1)}{N-1}\right)$, где $a \in [1,2], b = 2 - a, i$ – порядковый номер особи в списке особей, отсортированном по значению функции fitness ($\forall i, \forall j > i, f_i \leq f_j$ -- если задача минимизировать значение fitness).

- Позволяет избежать проблем, если значения функции fitness близки или в популяции присутствует хромосома с очень большим значением fitness-функции.

Выбор родителей

- Panmixia (панмиксия) — оба родителя выбираются случайно, каждая особь популяции имеет равные шансы быть выбранной.
- Inbreeding (инбридинг) — первый родитель выбирается случайно, а вторым выбирается такой, который наиболее похож на первого родителя (сравнение по fitness или по генотипу).
- Outbreeding (аутбридинг) — первый родитель выбирается случайно, а вторым выбирается такой, который наиболее не похож на первого родителя.

Иногда на шаге селекции выбирают для размножения не только из «выживших», но вообще из всех особей. Такой подход вынуждает хранить всех существовавших ранее особей, что увеличивает вычислительную сложность задачи.

Оператор обмена информацией

Кроссинговер:

1. Выбрать два индивидуума A и B из промежуточной популяции
2. Сгенерировать точку скрещивания (crossover point) -- равномерно распределенное случайное число k из интервала [0, m].
3. Поменять местами части выбранных индивидуумов, правее точки скрещивания.
4. Поместить выбранные хромосомы в новую популяцию.
5. Повторить шаги 1-5 до заполнения новой популяции.

$$
\begin{array}{ll|l}
A: & a_1 a_2\ a_3 \ldots\ a_k & a_{k+1} \ldots\ a_{m-1} a_m \\
B: & b_1 b_2\ b_3 \ldots\ b_k & b_{k+1} \ldots\ b_{m-1} b_m \\
\hline
A1: & a_1 a_2\ a_3 \ldots\ a_k & b_{k+1} \ldots\ b_{m-1} b_m \\
B1: & b_1 b_2\ b_3 \ldots\ b_k & a_{k+1} \ldots\ a_{m-1} a_m
\end{array}
$$

Оператор обмена информацией

2. Двухточеный кроссинговер

Выбираются две хромосомы и случайные позиции k и l. Хромосомы разрезаются по этим позициям и обмениваются средними частями:

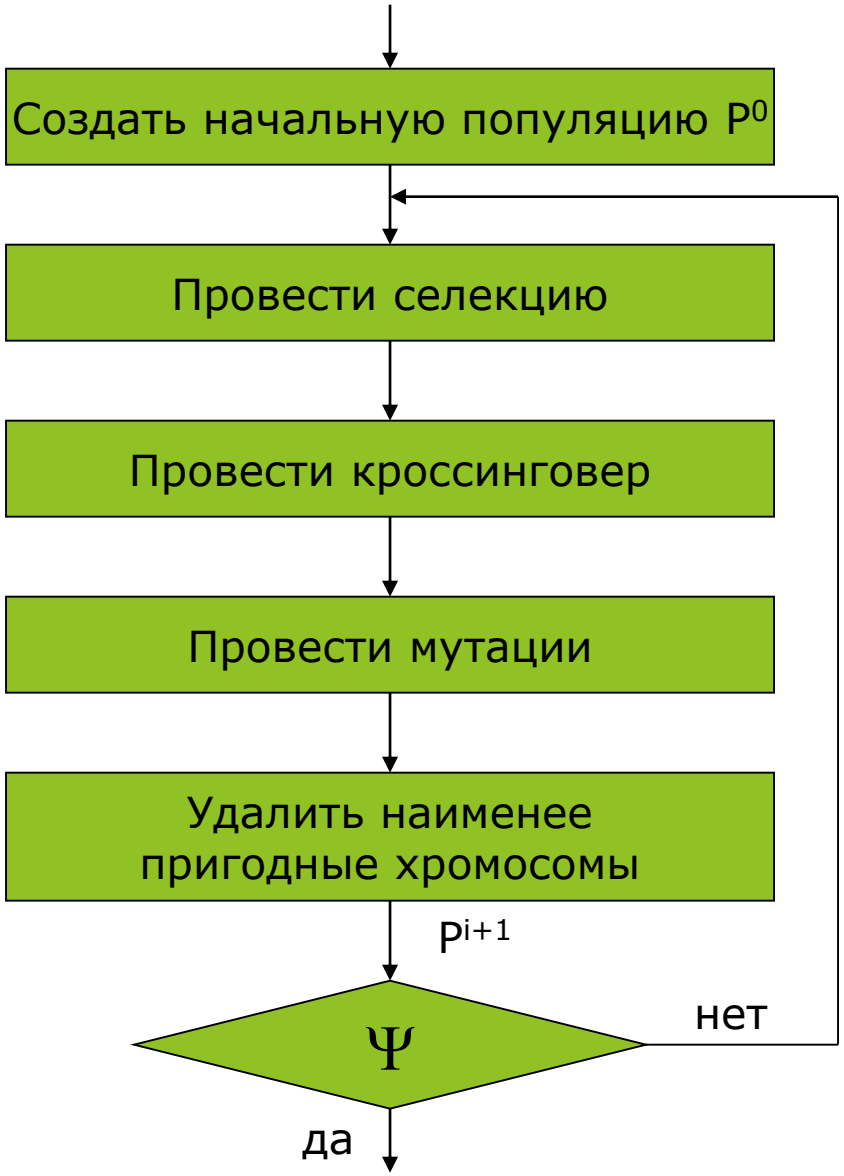
A:	a_1a_2	$a_3\ldots$	a_k	$a_{k+1}\ldots$	a_l	$a_{l+1}\ldots$	$a_{m-1}a_m$
B:	b_1b_2	$b_3\ldots$	b_k	$b_{k+1}\ldots$	b_l	$b_{l+1}\ldots$	$b_{m-1}b_m$
A1:	a_1a_2	$a_3\ldots$	a_k	$b_{k+1}\ldots$	b_l	$a_{l+1}\ldots$	$a_{m-1}a_m$
B1:	b_1b_2	$b_3\ldots$	b_k	$a_{k+1}\ldots$	a_l	$b_{l+1}\ldots$	$b_{m-1}b_m$

Блок-схема работы генетического алгоритма

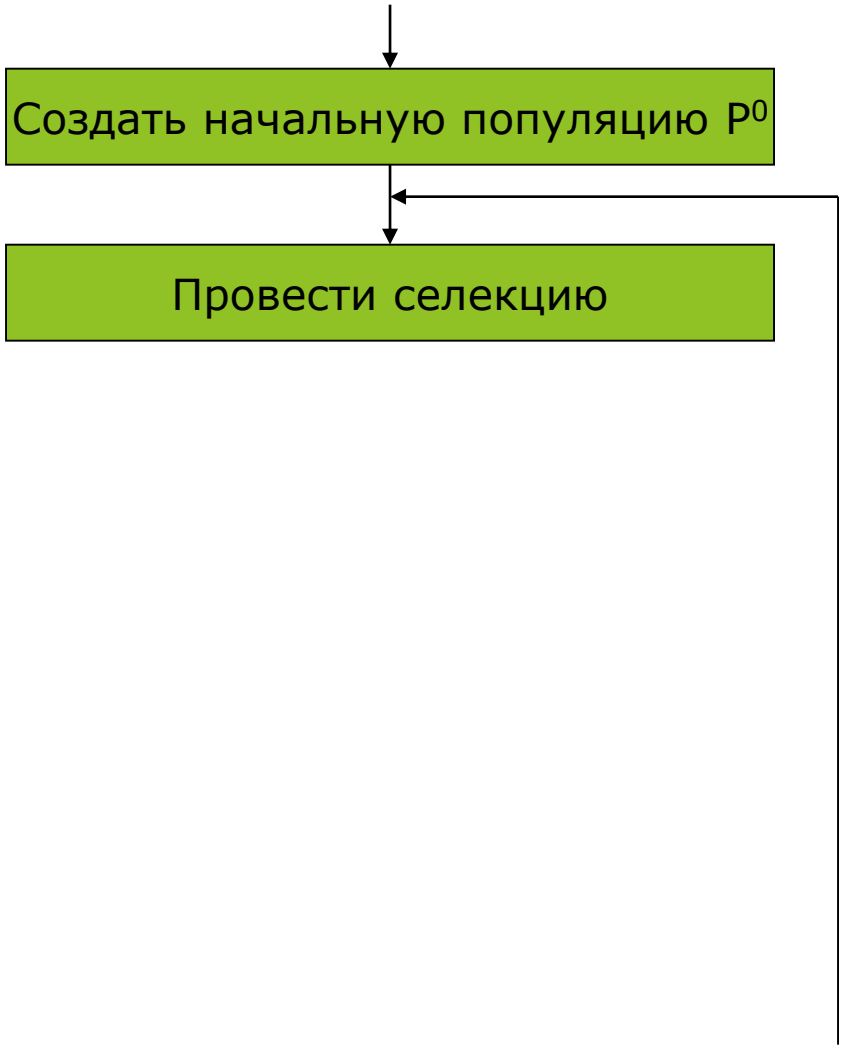
Мутация - случайное изменение каких-то особей. Этот механизм позволяет привнести что-то новое в генетическое разнообразие.

Оператор мутации определяет вероятность, с которой происходит инвертирование бита в случайно выбранной позиции

Блок-схема работы генетического алгоритма



Блок-схема работы генетического алгоритма



Создание нового поколения (селекция)

- **Без элитизма**

В новое поколение попадают только хромосомы, полученные путём рекомбинаций и мутаций из старого, которое забываем целиком.

- **Элитизм (elitism)**

Лучшие хромосомы из старого поколения сохраняются в новом

- **Случайный отбор (принцип рулетки)**

- **Устойчивое состояние (Steady state)** – выбираем либо родителей, либо новое поколение.

На каждом шаге генерируется только несколько новых хромосом и определяется, кто остается в популяции:

- потомок сохраняется, если он лучше родителей
- случайный выбор
- запрет на одинаковые хромосомы
- …

Критерии остановки

1. В популяции остаётся одно оптимальное решение
2. По числу итераций
3. По замедлению улучшения полученной фитнесс-функции
4. По допустимому времени работы
5. …

Преимущества и недостатки

-

• Много расчётов, во многих случаях классические алгоритмы оказываются эффективнее

+

• Хорошие возможности распараллеливания
• Очень низкие требования к виду оптимизируемой функции
• Может избегать локальных экстремумов