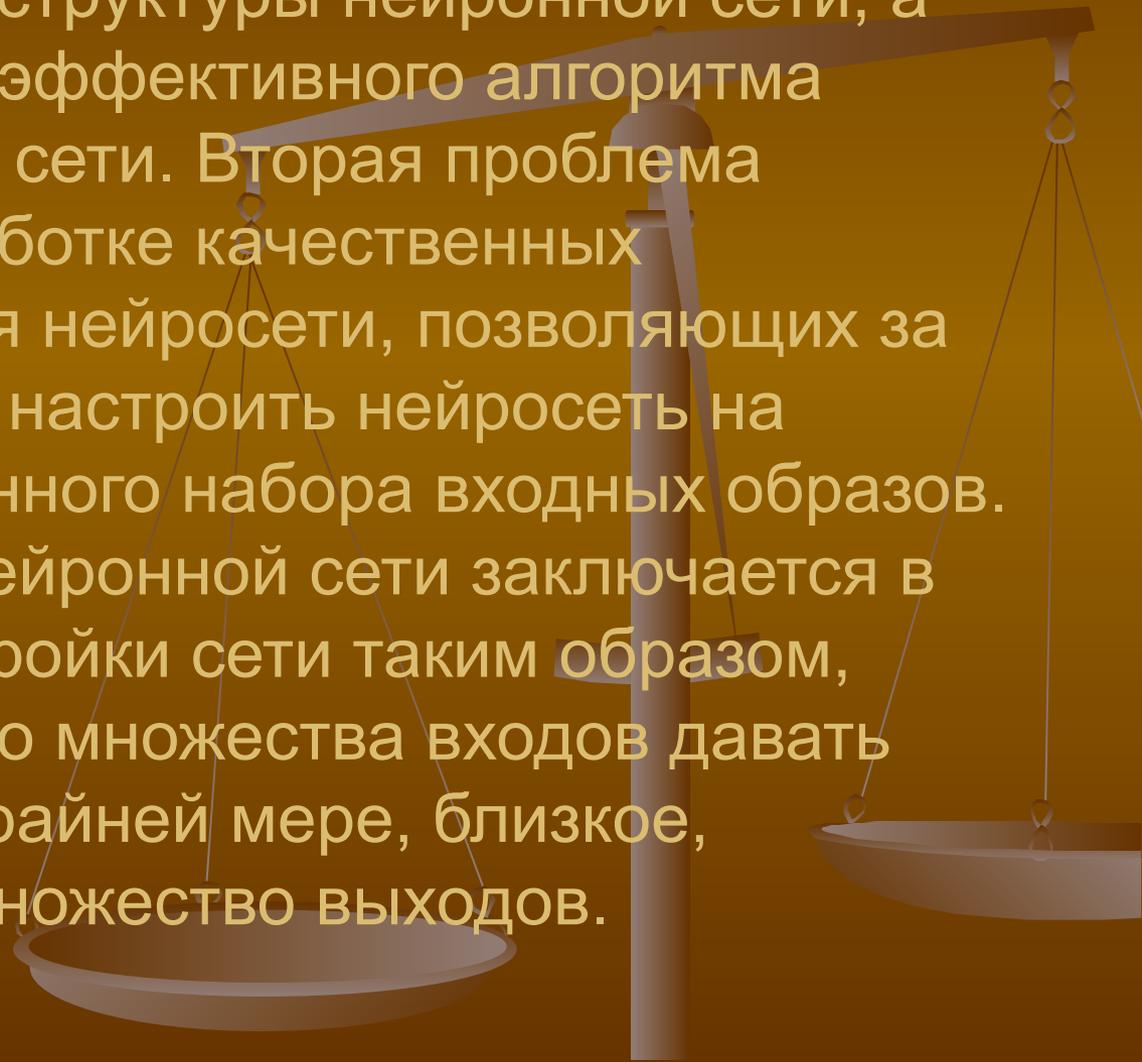


**ПРИМЕНЕНИЕ
ГЕНЕТИЧЕСКИХ
АЛГОРИТМОВ ДЛЯ
ОБУЧЕНИЯ НЕЙРОННЫХ
СЕТЕЙ**

В теории нейронных сетей существуют две актуальных проблемы, одной из которых является выбор оптимальной структуры нейронной сети, а другой - построение эффективного алгоритма обучения нейронной сети. Вторая проблема заключается в разработке качественных алгоритмов обучения нейросети, позволяющих за минимальное время настроить нейросеть на распознавание заданного набора входных образов. Процесс обучения нейронной сети заключается в необходимости настройки сети таким образом, чтобы для некоторого множества входов давать желаемое (или, по крайней мере, близкое, соответствующее с ним) множество выходов.



Стратегия обучения с учителем предполагает, что есть обучающее множество $\{X, Y\}$.

Обучение осуществляется путем последовательного предъявления векторов обучающего множества, с одновременной подстройкой весов в соответствии с определенной процедурой, пока ошибка настройки по всему множеству не достигнет приемлемого низкого уровня.

$$Err = \sum_{i=1}^N |\bar{Y}_i - Y_i| \rightarrow \min$$

Зависимость реального выходного сигнала Y от входного сигнала X можно записать в виде: $Y = F(W, X) + Err$, где:

- $F(W, X)$ - некоторая функция, вид которой задается алгоритмом обучения нейронной сети;
- W - множество параметров, позволяющих настроить функцию на решение определенной задачи распознавания образов (количество слоев сети, количество нейронов в каждом слое сети, матрица синаптических весов сети);
- Err - некоторая ошибка, возникающая из-за неполного соответствия реального значения выходного сигнала нейронной сети требуемому значению, а также погрешности в вычислениях.

Генетический алгоритм

- Генетический алгоритм является самым известным на данный момент представителем эволюционных алгоритмов и по своей сути является алгоритмом для нахождения глобального экстремума многоэкстремальной функции. Он заключается в параллельной обработке множества альтернативных решений. При этом поиск концентрируется на наиболее перспективных из них. Это говорит о возможности использования генетических алгоритмов при решении любых задач искусственного интеллекта, оптимизации, принятия решений.

Генетические алгоритмы для подстройки весов скрытых и выходных слоев используются следующим образом. Каждая хромосома (решение, последовательность, индивидуальность, "родитель", "потомок", "ребенок") представляет собой *вектор из весовых коэффициентов* (веса считываются с нейронной сети в установленном порядке - слева направо и сверху вниз).

Хромосома $a=(a_1, a_2, a_3, \dots, a_n)$ состоит из генов a_i , которые могут иметь числовые значения, называемые "аллели".

Популяцией называют набор хромосом (решений).

Эволюция популяций - это чередование поколений, в которых хромосомы изменяют свои признаки, чтобы каждая новая популяция наилучшим способом приспособилась к внешней среде. Заметим, что в нашем случае каждый "ген" в хромосоме - реальное число, а не бит.

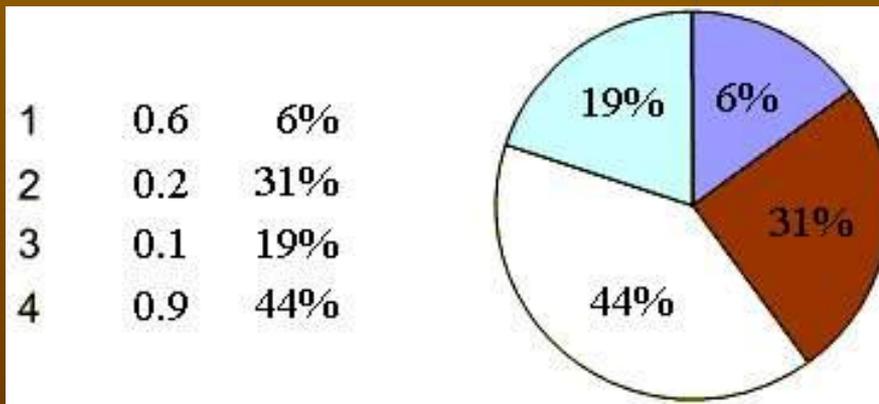
Начальная популяция выбирается случайно, значения весов лежат в промежутке $[-1.0, 1.0]$. Для обучения сети к начальной популяции применяются простые операции: **селекция, скрещивание, мутация**, - в результате чего генерируются новые популяции.

У генетического алгоритма есть такое свойство как вероятность. Т.е. описываемые операторы не обязательно применяются ко всем хромосомам, что вносит дополнительный элемент неопределенности в процесс поиска решения. В данном случае, неопределенность не подразумевает негативный фактор, а является своеобразной "степенью свободы" работы генетического алгоритма.

Оператор селекции (reproduction, selection) осуществляет отбор хромосом в соответствии со значениями их функции приспособленности. Здесь применим **метод рулетки (roulette-wheel selection)**, с помощью которого осуществляется отбор. Колесо рулетки содержит по одному сектору для каждого члена популяции. Размер *i*-го сектора пропорционален соответствующей величине $P_{sel}(i)$ вычисляемой по формуле:

$$P_i = \frac{f_i(x)}{\sum f(x)}$$

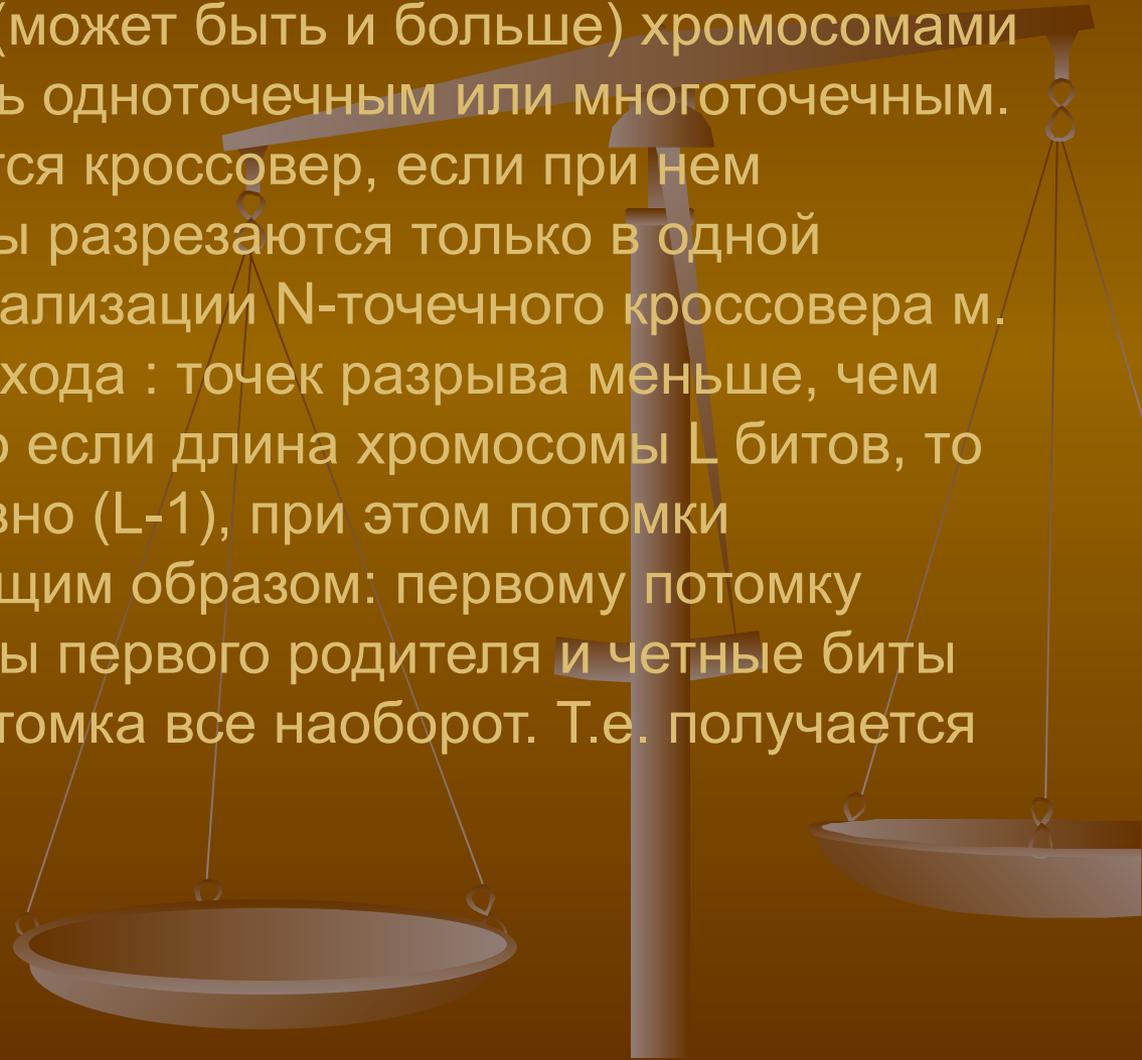
$f_i(x)$ - значение целевой функции *i*-й хромосомы в популяции; ;
 $\sum f(x)$ - суммарное значение целевой функции всех хромосом в популяции;



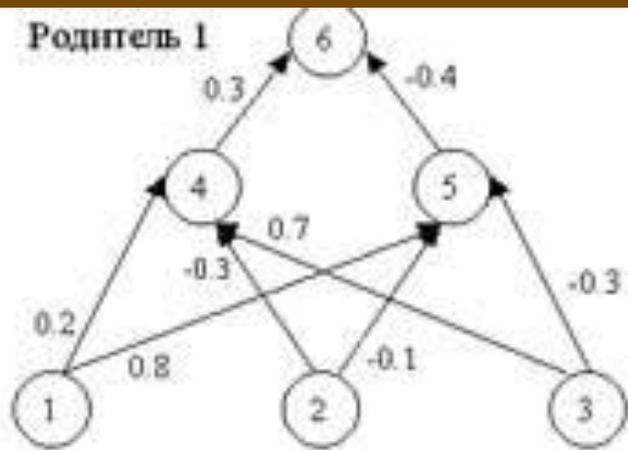
Реализация работы оператора селекции

Номер хромосомы	Начальная популяция (генерируется случайно) - величина погрешности f_i	Нормализованная величина p_i	Нормализованная величина $k_i=2p_i^{cp}-p_i$	Ожидаемое кол-во копий хромосомы в след. поколении	Реальное кол-во копий хромосомы в след. поколении
1	0.8	0.5	0	0	0
2	0.2	0.125	0.375	1.5	1
3	0.5	0.3125	0.1875	0.75	1
4	0.1	0.0625	0.4375	1.75	2
Суммарная функция (sumf(x))	1.6	1.00	1.00	4.00	4
Среднее значение функции $f_{cp}(x)$	0.4	0.25	0.25	1.00	1
Min значение функции $f(x)$	0.1	0.0625	0	0	0

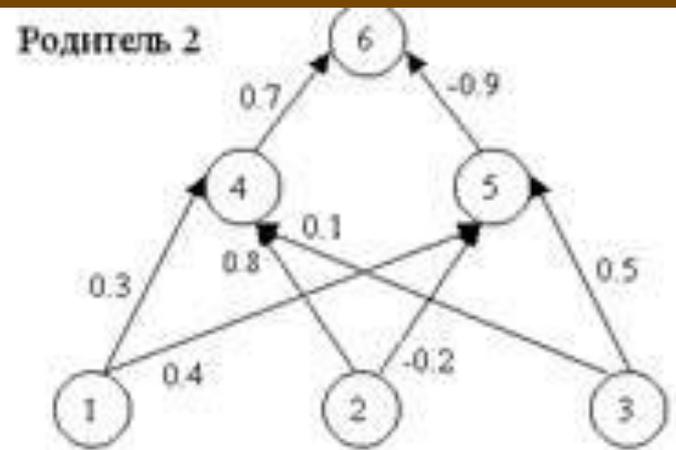
Оператор кроссовера (crossover operator) иногда называемый кроссинговером, является основным генетическим оператором, за счет которого производится обмен частями хромосом между двумя (может быть и больше) хромосомами в популяции. Может быть односточечным или многоточечным. Односточечным называется кроссовер, если при нем родительские хромосомы разрезаются только в одной случайной точке. Для реализации N-точечного кроссовера м. б. использовано два подхода : точек разрыва меньше, чем генов в хромосоме, либо если длина хромосомы L битов, то число точек разрыва равно $(L-1)$, при этом потомки наследуют биты следующим образом: первому потомку достаются нечетные биты первого родителя и четные биты второго; у второго же потомка все наоборот. Т.е. получается как бы "расческа".



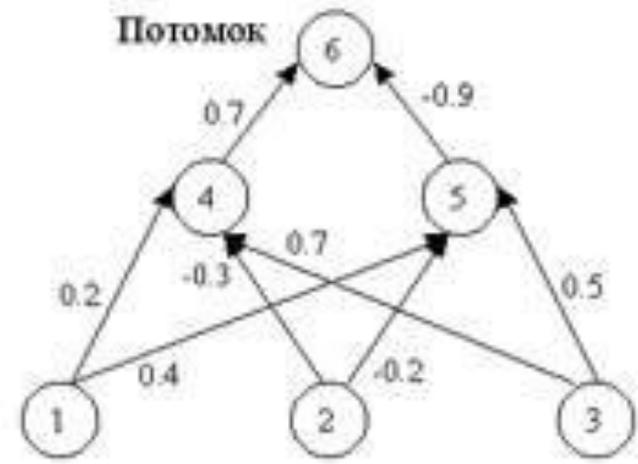
Оператор скрещивания



(0.3 -0.4 0.2 0.8 -0.3 -0.1 0.7 -0.3)

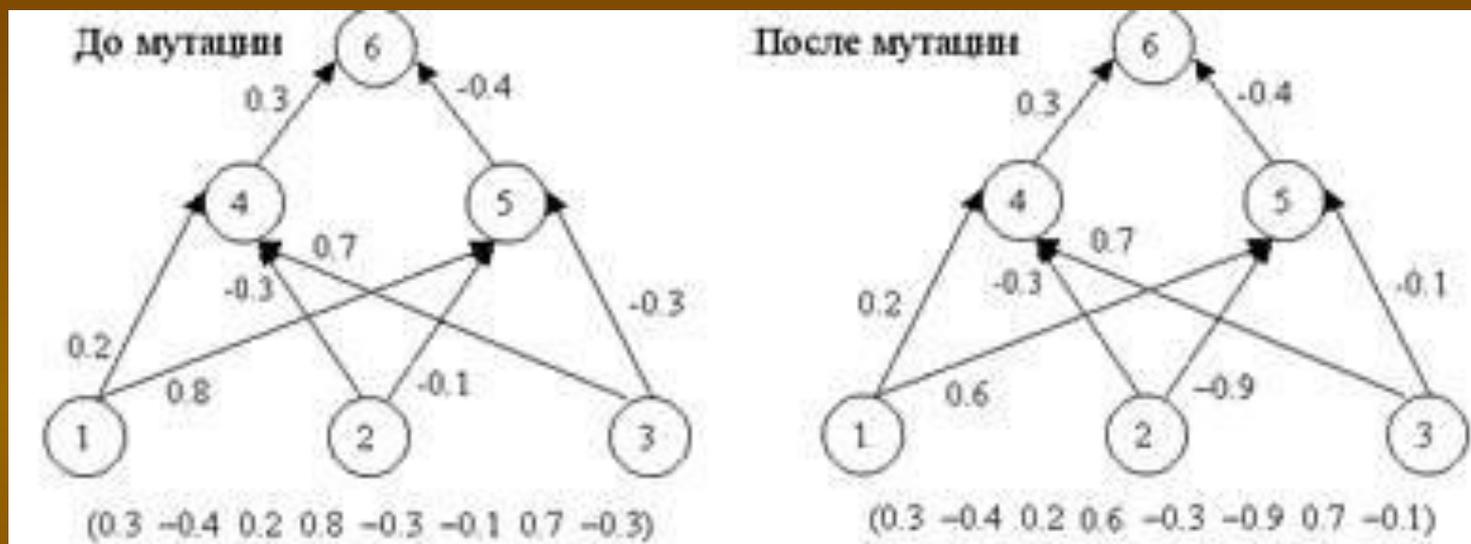


(0.7 -0.9 0.3 0.4 0.8 -0.2 0.1 0.5)



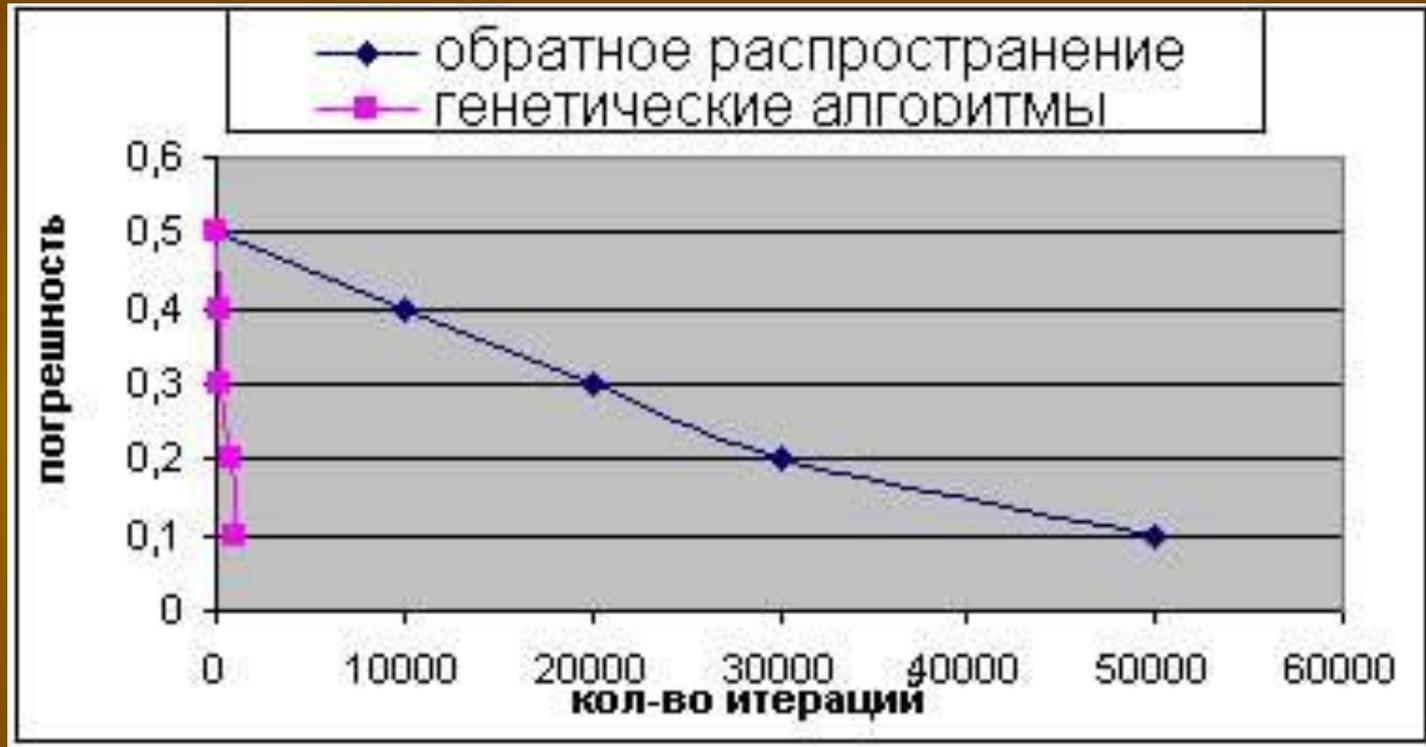
(0.7 -0.9 0.2 0.4 -0.3 -0.2 0.7 0.5)

Оператор мутации (mutation operator) – стохастическое изменение части хромосом. Он необходим для "выбивания" популяции из локального экстремума и способствует защите от преждевременной сходимости.



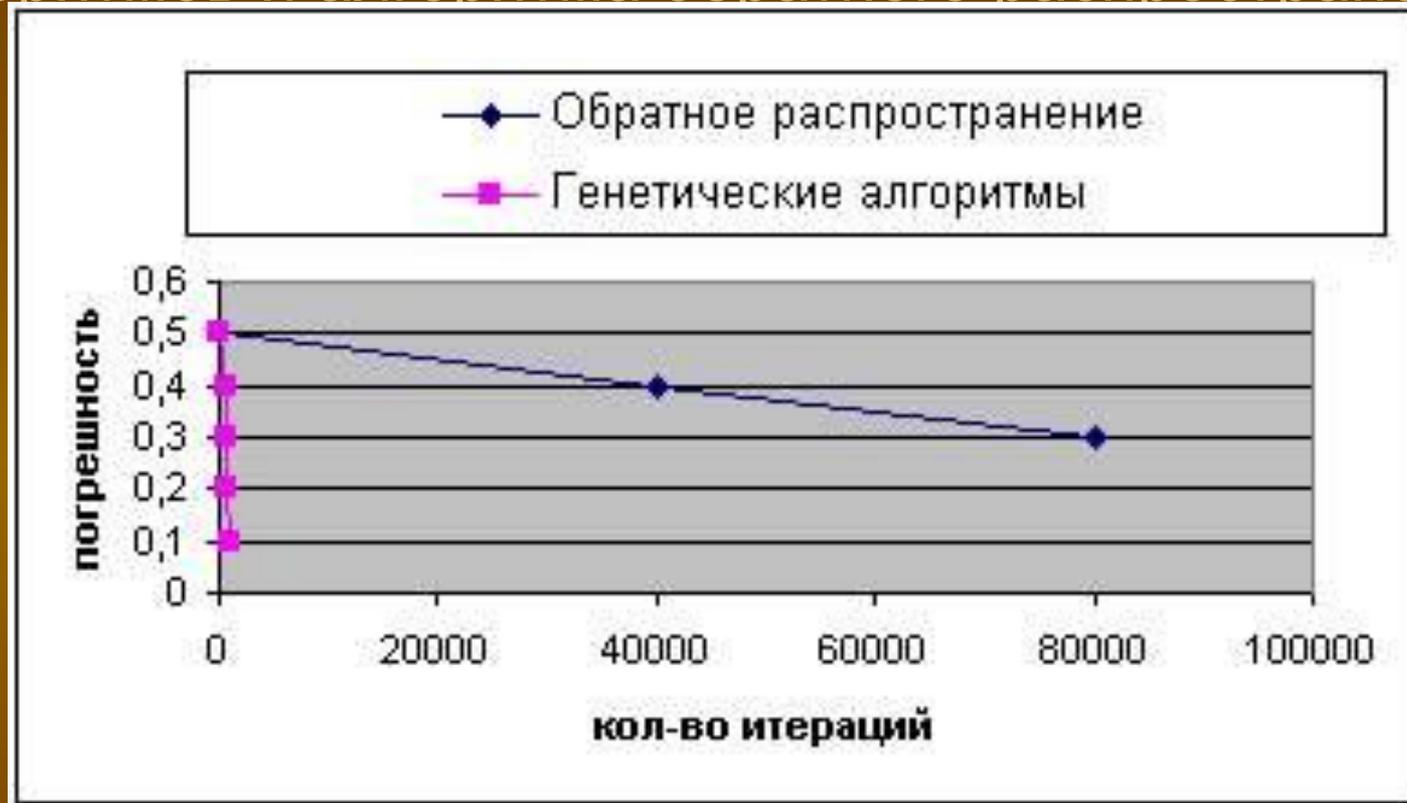
Достигаются это счет того, что каждый ген строки, которая подвергается мутации, с малой вероятностью P_{mut} меняется на другой ген (добавляется случайная величина между -1.0 и 1.0 к весу). Так же как и кроссовер, мутация проводится не только по одной случайной точке. Можно выбирать некоторое количество точек в хромосоме для изменения, причем их число также может быть случайным. Вероятность мутации значительно меньше вероятности кроссовера и редко превышает 1%.

Сравнение результатов работы генетических алгоритмов и алгоритма обратного распространения



- В случае двухслойной нейронной сети с характеристиками: 4 входных сигнала, 3 нейрона скрытого слоя, 1 выходной сигнал

Сравнение результатов работы генетических алгоритмов и алгоритма обратного распространения



- В случае трехслойной нейронной сети с характеристиками: 5 входных сигналов, 3 нейрона в первом скрытом слое, 2 нейрона во втором скрытом слое, 2 выходных сигнала.

Сеть в обоих случаях являлась полносвязной, таким образом, между нейронами было в первом случае 15 связей, а во втором 25. На вход НС подавалось обучающее множество, представляющее собой последовательность 0 и/или 1. Выполнялось обучение нейронной сети с помощью алгоритма обратного распространения ошибки и с помощью генетических алгоритмов. Обучение в обоих случаях заканчивается, когда на всем обучающем реальным выходной сигнал был равен требуемому выходному сигналу. Для обучения нейронной сети достаточно было выполнить 500 генераций с помощью генетических алгоритмов. Алгоритмам обратного распространения требовалось во много раз большее число итераций - около 500000 итераций, где одна итерация - это полный пересчет всех обучающих данных (весовые коэффициенты, погрешность, значения выходов нейронной сети).

