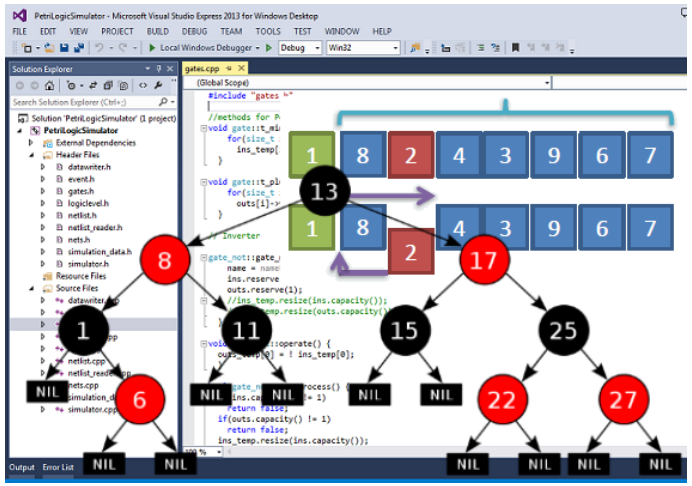
Национальный исследовательский университет «МИЭТ»

Институт интегральной электроники (группы ЭН-24-25, каф. ПКИМС)

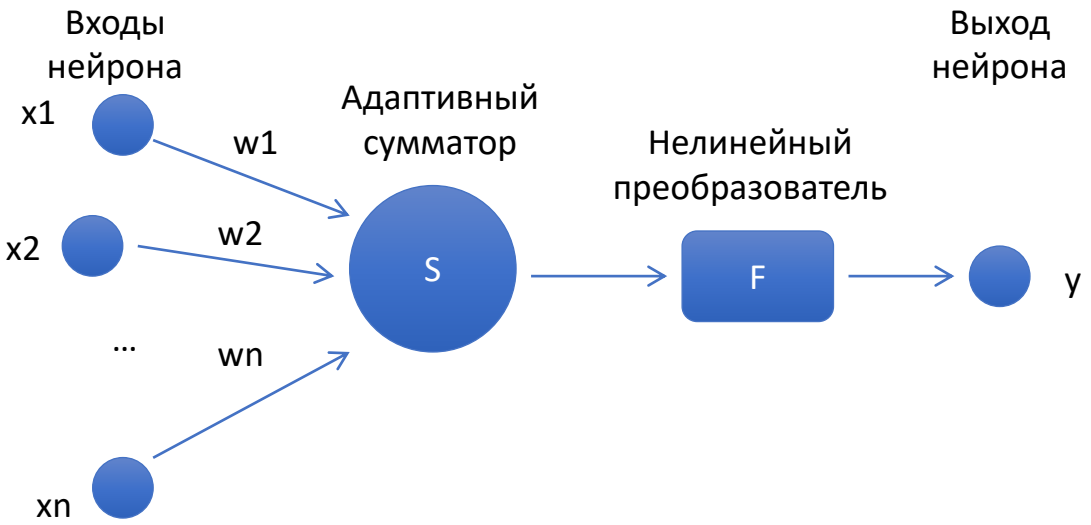
Теория алгоритмов



Лекция 8

Алгоритмы решения сложных задач

Нейронные сети: структура нейрона



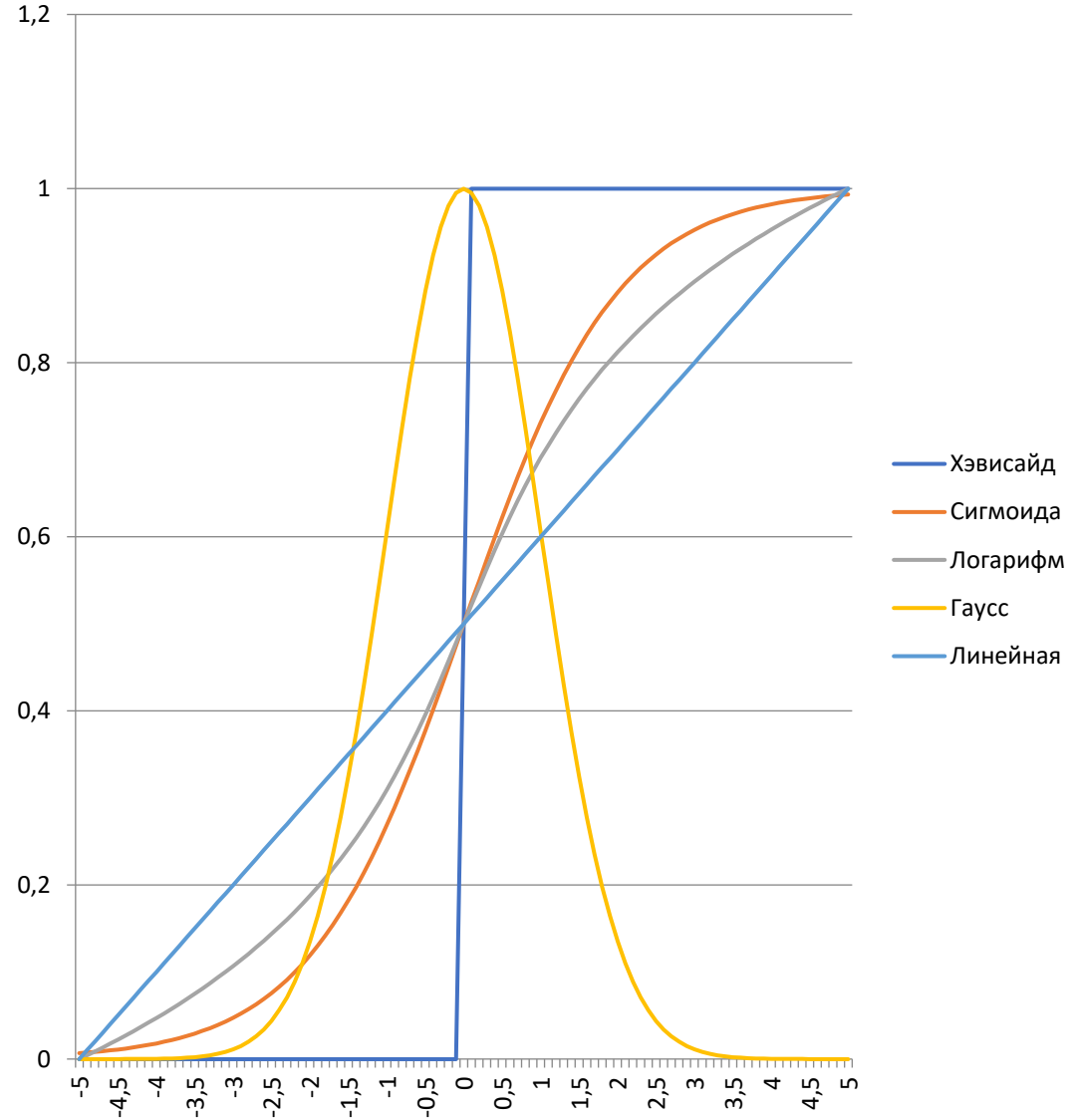
Фаза 1: фиксирование входных данных (взвешивание данных)

$$S = \sum_{i=1}^{n} x_i \cdot w_i$$

Фаза 2: активация нейрона (принятие решения)

$$y = F(S)$$

Нейронные сети: основные модели активации



Виды активационных функций:

- функция Хэвисайда

$$H(z) = \begin{cases} 0, z < 0 \\ 0.5, z = 0 \\ 1, z > 0 \end{cases}$$

- сигмоидальная

$$S(z) = \frac{1}{(1 + e^{-z})}$$

- логарифмическая

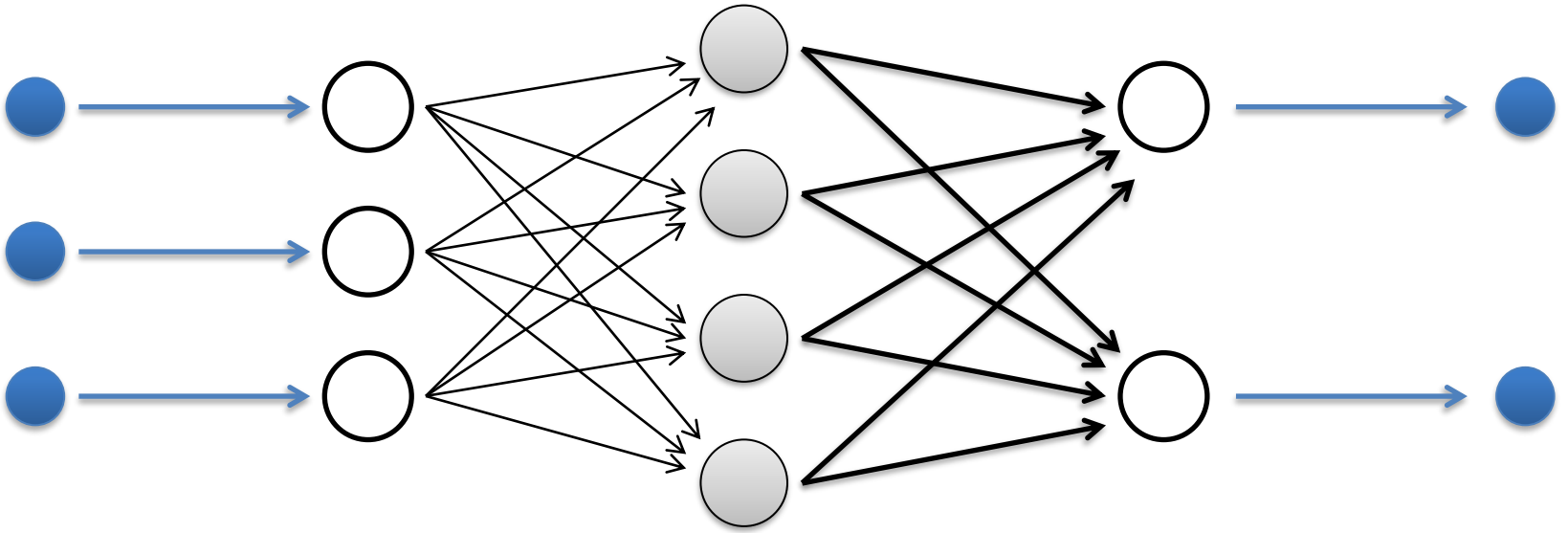
$$L(z) = \ln(z + \sqrt{z^2 + 1})$$

- гауссовская

$$G(z) = \frac{1}{e^{\frac{z^2}{2}}}$$

- линейная

Многослойные нейронные сети (1)



Входные данные

Входной слой нейронов

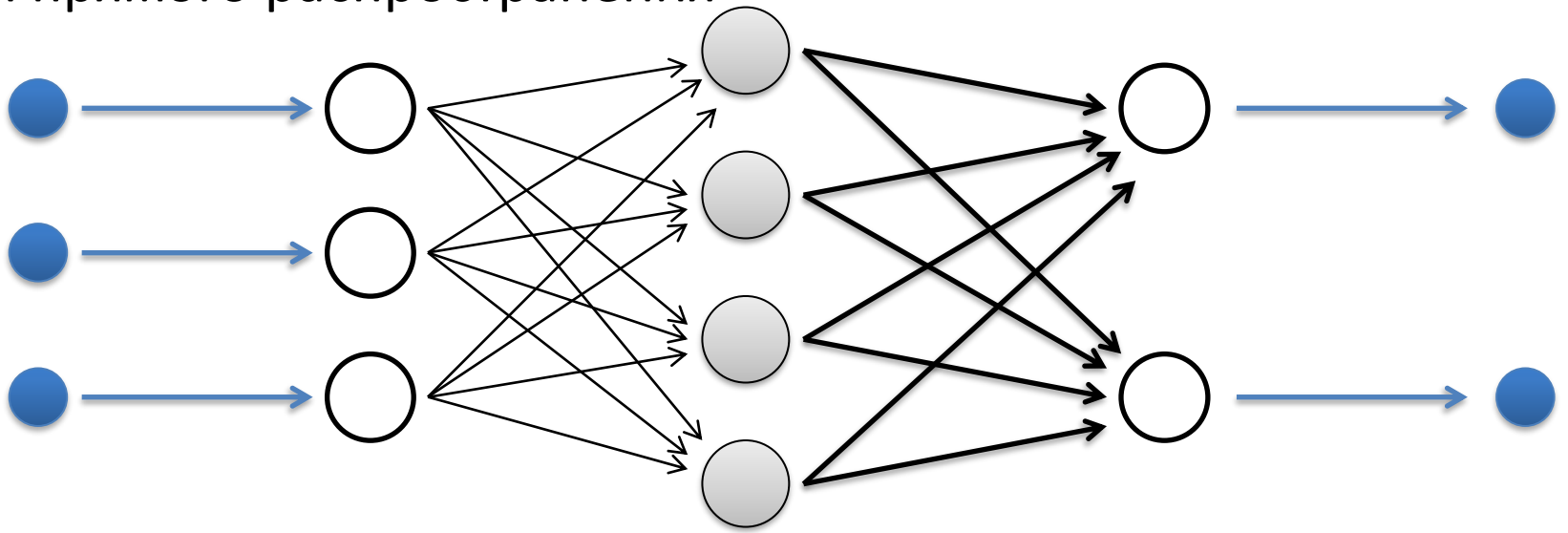
Скрытые слои нейронов

Выходной слой нейронов

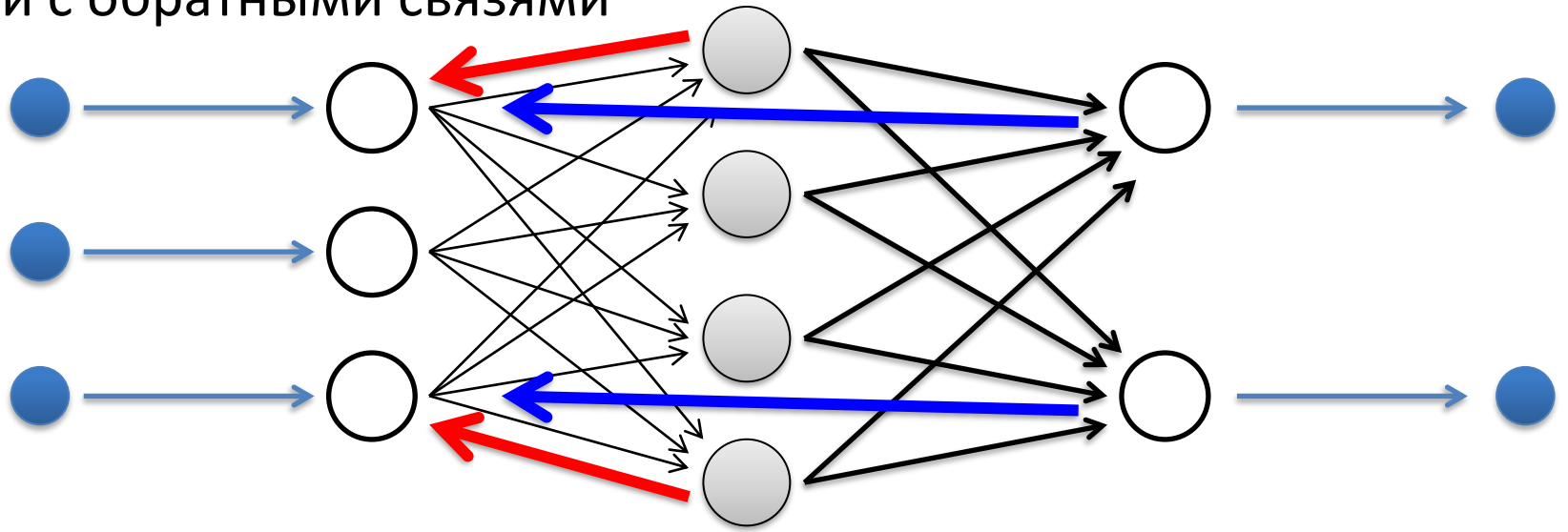
Выходные данные

Многослойные нейронные сети (2)

Сети прямого распространения



Сети с обратными связями



Нейронные сети с обучением

Задача обучения – подбор правильных значений весовых коэффициентов.

2 основных вида обучения.

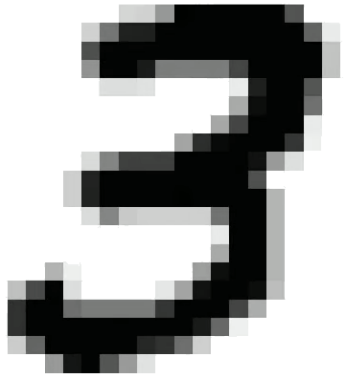
1. Обучение с учителем:

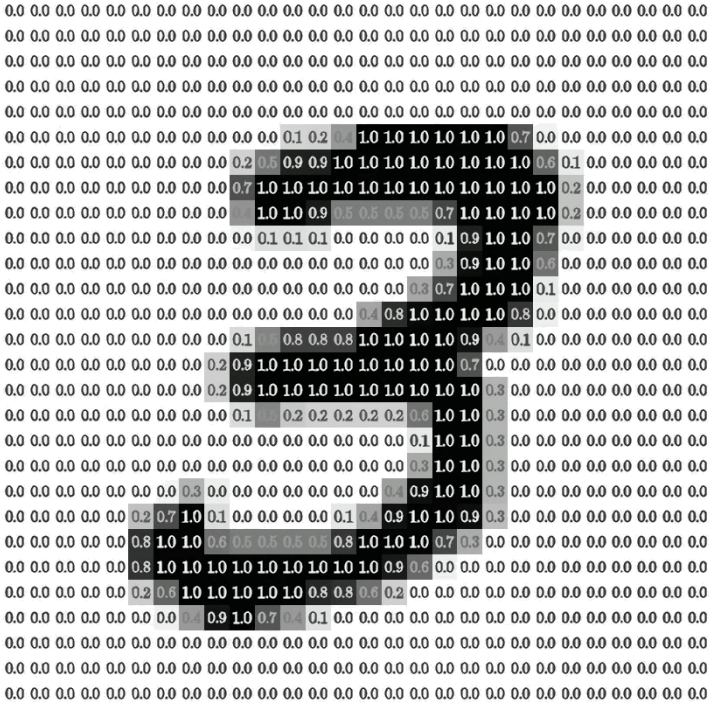
   на вход сети подаются тестовые комбинации, на выходы подаётся точный результат, который должен получиться.

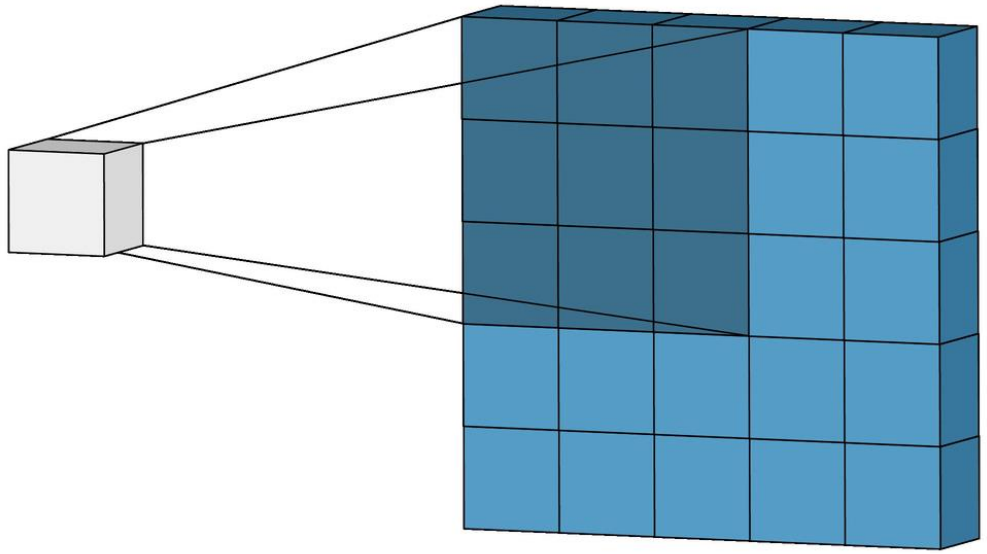
2. Обучение с поощрением:

   на вход сети подаются тестовые комбинации, на выходы результат не подаётся, но нейронной сети объясняется, правильно ли получился ответ или нет.

Нейронные сети: идентификация цифр







Эвристические алгоритмы: генетический алгоритм

Генетический алгоритм – эвристический алгоритм поиска, основанный на случайном подборе значений, комбинировании значений и параметров с использованием механизмов, имитирующих эволюционные процессы.

Хромосома – входные данные для алгоритма, набор конкретных значений (численных - для решаемой задачи, аналог приближения для итерационных методов в численных методах), на основе которых определяется функционирование особи (соответствие функции решению)

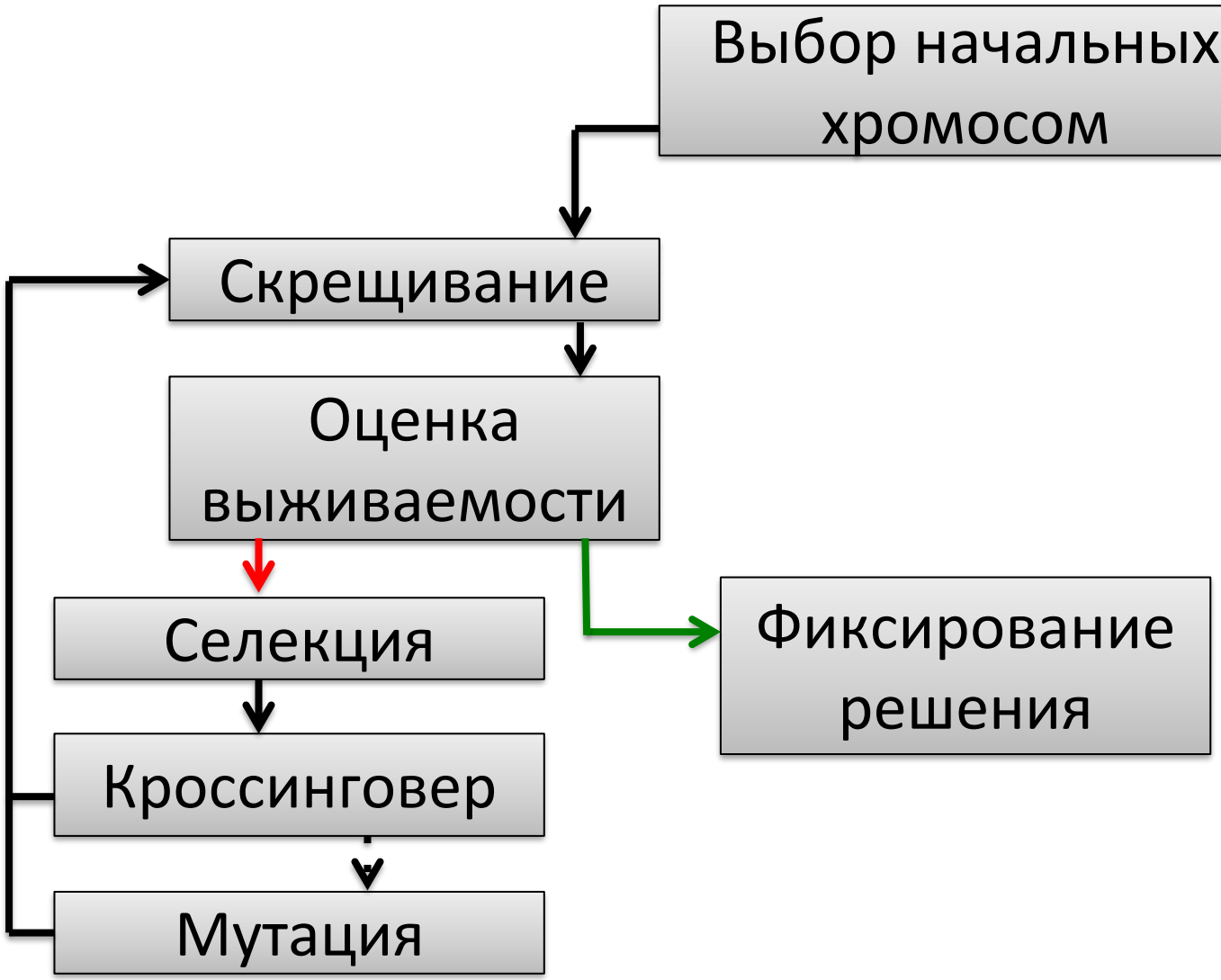
Скрещивание – получение особи (решения) на основе наборов хромосом

Селекция – определение наиболее приспособленных особей и отбор их хромосом для дальнейшей эволюции

Кроссинговер – перемешивание генов родителей для получения генов потомков (комбинация хромосом с учётом определённых правил, дающих новые комбинации числовых значений)

Мутация – генерация хромосом не на основе кроссинговера, а на основе некоторых преобразований, включая случайную генерацию значений.

Блок-схема генетического алгоритма и принцип эволюции



Выживать должны наиболее приспособленные особи, имеющие лучший набор хромосом.

Следующее поколение должно наследовать преимущественно лучшие гены.

Мутации позволяют приспосабливаться к внешним и внутренним условиям.

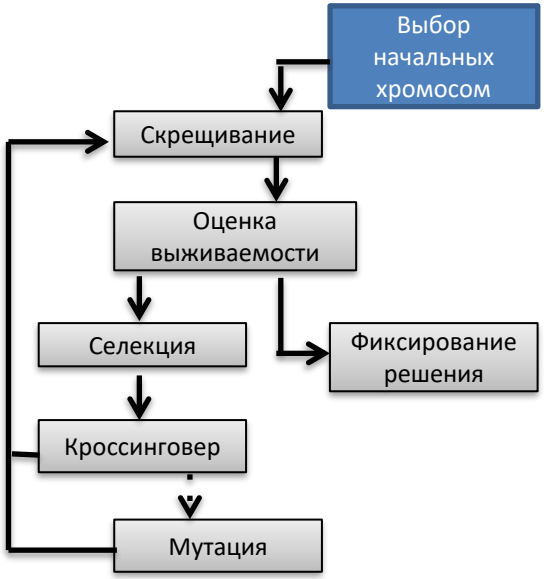
Генетические алгоритмы: выбор хромосом

Решаемое уравнение:

$$2x_1 + 4x_2 - 3x_3 = 6$$

Выбор начальных хромосом

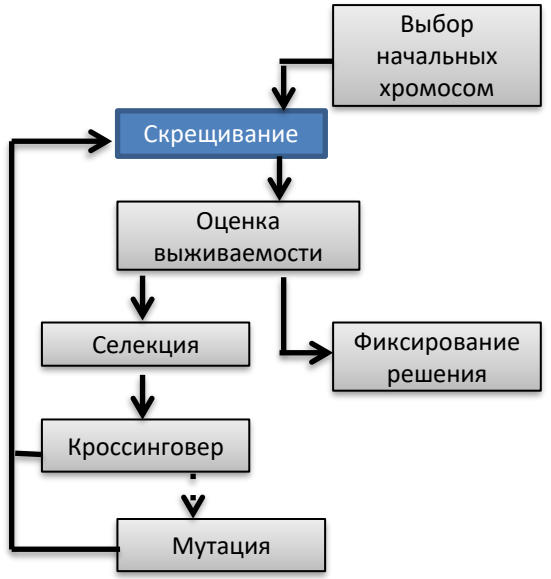
Номер хромосомы	Значение хромосомы
1	{ 1, 5, 2 }
2	{ 3, 6, 1 }
3	{ 4, 4, 4 }
4	{ 6, 2, 1 }



Генетические алгоритмы: скрещивание

Решаемое уравнение:

$$2x_1 + 4x_2 - 3x_3 = 6$$



Хромосомы:

№	Значение
1	{ 1, 5, 2 }
2	{ 3, 6, 1 }
3	{ 4, 4, 4 }
4	{ 6, 2, 1 }

Решения:

№	Значение	Решение
1	{ 1, 5, 2 }	2 + 20 – 6 = 16
2	{ 3, 6, 1 }	6 + 24 – 3 = 27
3	{ 4, 4, 4 }	8 + 16 – 12 = 12
4	{ 6, 2, 1 }	12 + 8 – 3 = 17

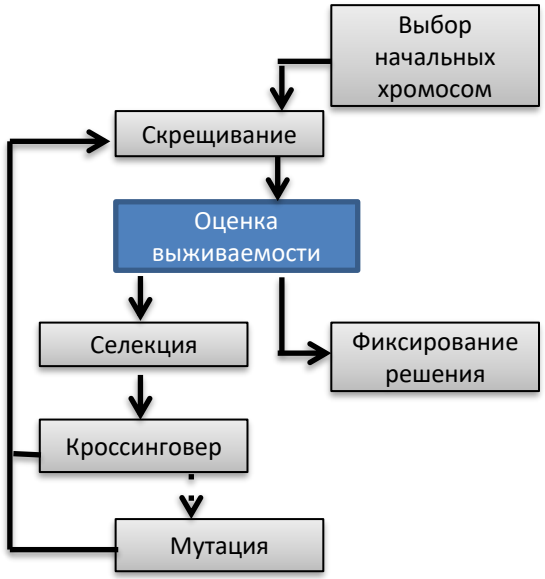
Генетические алгоритмы: оценка выживаемости (1)

Решаемое уравнение:

$$2x_1 + 4x_2 - 3x_3 = 6$$

Решения:

№	Значение	Решение
1	{ 1, 5, 2 }	2 + 20 – 6 = 16
2	{ 3, 6, 1 }	6 + 24 – 3 = 27
3	{ 4, 4, 4 }	8 + 16 – 12 = 12
4	{ 6, 2, 1 }	12 + 8 – 3 = 17



Вычисление коэффициента близости:

$$K1 = |solution - exact\_solution|$$

Вычисление коэффициента выживаемости:

$$K2 = \frac{1}{|solution - exact\_solution|}$$

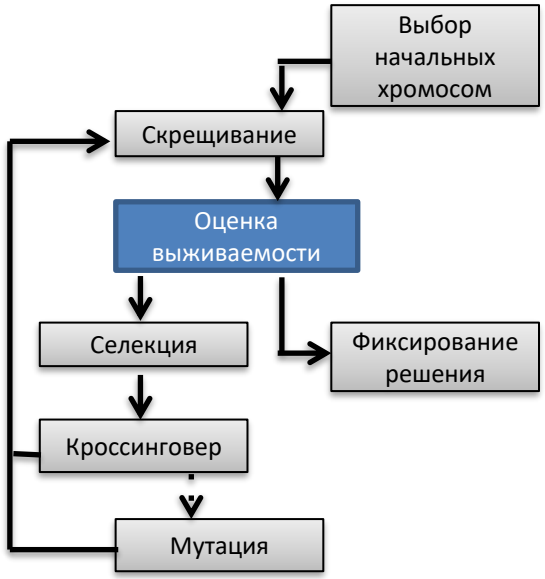
Генетические алгоритмы: оценка выживаемости (2)

Решаемое уравнение:

$$2x_1 + 4x_2 - 3x_3 = 6$$

Решения:

№	Значение	Решение
1	{ 1, 5, 2 }	2 + 20 – 6 = 16
2	{ 3, 6, 1 }	6 + 24 – 3 = 27
3	{ 4, 4, 4 }	8 + 16 – 12 = 12
4	{ 6, 2, 1 }	12 + 8 – 3 = 17



Коэффициенты близости:

№	Значение	Решение	К1
1	{ 1, 5, 2 }	2 + 20 – 6 = 16	10
2	{ 3, 6, 1 }	6 + 24 – 3 = 27	21
3	{ 4, 4, 4 }	8 + 16 – 12 = 12	6
4	{ 6, 2, 1 }	12 + 8 – 3 = 17	11

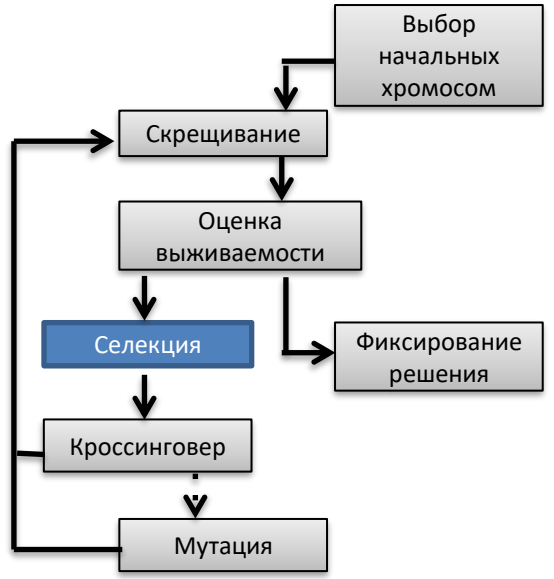
Генетические алгоритмы: селекция

Решаемое уравнение:

$$2x_1 + 4x_2 - 3x_3 = 6$$

Коэффициенты выживаемости:

№	Значение	Решение	К2
1	{ 1, 5, 2 }	2 + 20 – 6 = 16	0.1
2	{ 3, 6, 1 }	6 + 24 – 3 = 27	0.048
3	{ 4, 4, 4 }	8 + 16 – 12 = 12	0.17
4	{ 6, 2, 1 }	12 + 8 – 3 = 17	0.091



Критерии выбора наиболее подходящих потомков:

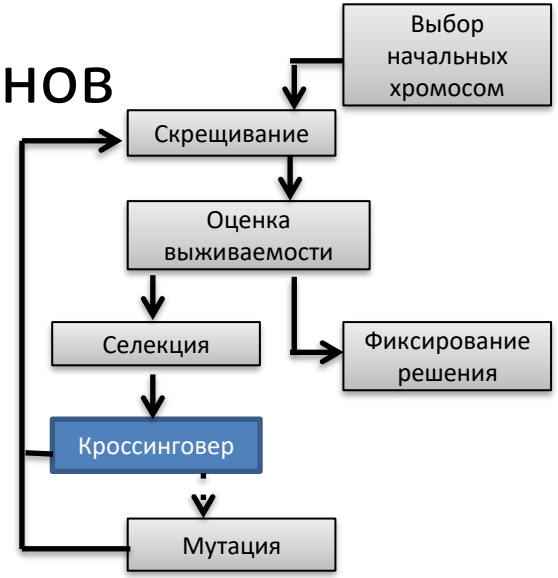
1. отсеивание
2. вероятностный подход

Вероятность выживания:

№	Значение
1	24 %
2	12 %
3	42 %
4	22 %

Генетические алгоритмы: варианты реализации кроссинговера

Задача кроссинговера – смешивание генов



- Одноточечный кроссинговер:

  ABCD EF**GH** ABGH

- Многоточечный кроссинговер:

  AB**C**D E**F**G**H** AFCH

- Арифметический кроссинговер:

  с участием обычной и модулярной арифметики, побитовых операций

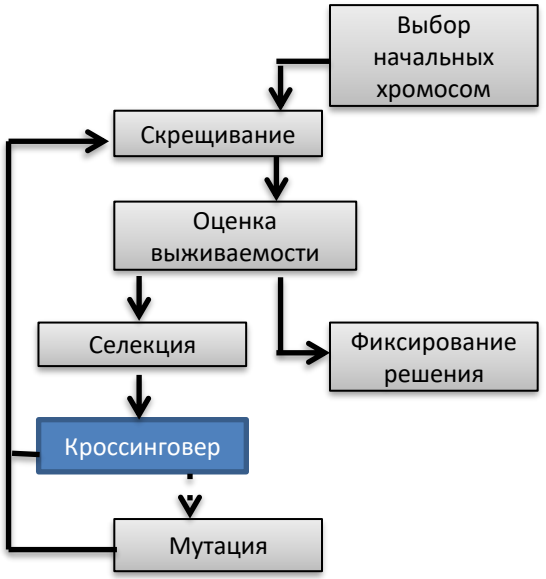
Генетические алгоритмы: кроссинговер

Номера родителей:

Родитель 1	Родитель 2
1	3
3	1
1	4
3	4

Гены родителей:

Родитель 1	Родитель 2
{ 1, 5, 3 }	{ 4, 4, 4 }
{ 4, 4, 4 }	{ 1, 5, 2 }
{ 1, 5, 2 }	{ 6, 2, 1 }
{ 4, 4, 4 }	{ 6, 2, 1 }



Кроссинговер:

Родитель 1	Родитель 2	Потомок
{ 1, 5, 2 }	{ 4, 4, 4 }	{ 1, 5, 4 }
{ 4, 4, 4 }	{ 1, 5, 2 }	{ 4, 5, 2 }
{ 1, 5, 2 }	{ 6, 2, 1 }	{ 1, 5, 1 }
{ 4, 4, 4 }	{ 6, 2, 1 }	{ 4, 2, 1 }

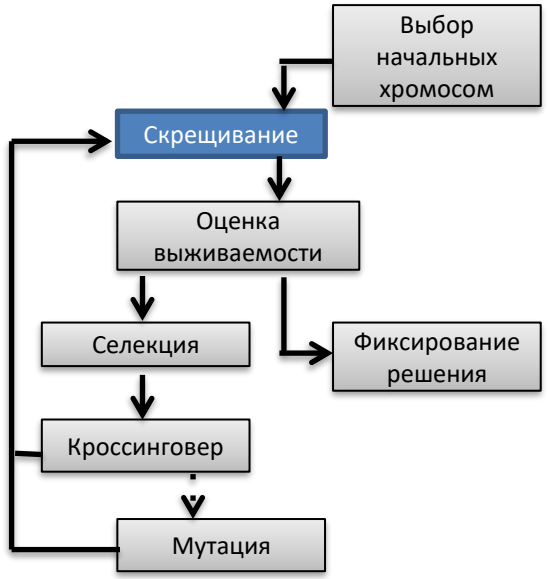
Генетические алгоритмы: скрещивание

Решаемое уравнение:

$$2x_1 + 4x_2 - 3x_3 = 6$$

Хромосомы:

№	Значение
1	{ 1, 5, 4 }
2	{ 4, 5, 2 }
3	{ 1, 5, 1 }
4	{ 4, 2, 1 }



Решения:

№	Значение	Решение
1	{ 1, 5, 4 }	2 + 20 – 12 = 10
2	{ 4, 5, 2 }	8 + 20 – 6 = 22
3	{ 1, 5, 1 }	2 + 20 – 3 = 19
4	{ 4, 2, 1 }	8 + 8 – 3 = 13

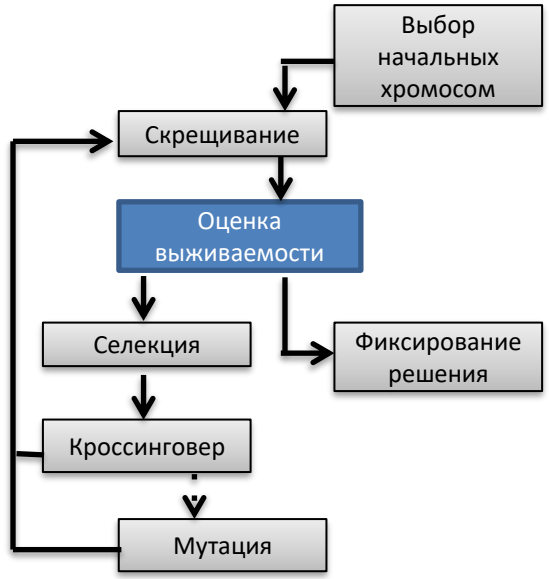
Генетические алгоритмы: оценка выживаемости (1)

Решаемое уравнение:

$$2x_1 + 4x_2 - 3x_3 = 6$$

Решения:

№	Значение	Решение
1	{ 1, 5, 4 }	2 + 20 – 12 = 10
2	{ 4, 5, 2 }	8 + 20 – 6 = 22
3	{ 1, 5, 1 }	2 + 20 – 3 = 19
4	{ 4, 2, 1 }	8 + 8 – 3 = 13



Коэффициенты близости:

№	Значение	Решение	К1
1	{ 1, 5, 4 }	2 + 20 – 12 = 10	4
2	{ 4, 5, 2 }	8 + 20 – 6 = 22	16
3	{ 1, 5, 1 }	2 + 20 – 3 = 19	13
4	{ 4, 2, 1 }	8 + 8 – 3 = 13	10

Вероятность выживания:

№	Значение
1	51 %
2	12 %
3	16 %
4	21 %

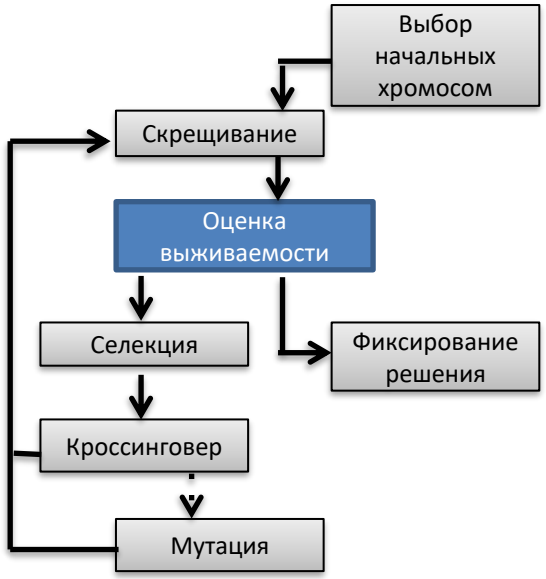
Генетические алгоритмы: оценка выживаемости (2)

Решаемое уравнение:

$$2x_1 + 4x_2 - 3x_3 = 6$$

Коэффициенты близости родителей:

№	Значение	Решение	K2	%
1	{ 1, 5, 2 }	2 + 20 – 6 = 16	**0,1**	**24 %**
2	{ 3, 6, 1 }	6 + 24 – 3 = 27	**0,048**	**12 %**
3	{ 4, 4, 4 }	8 + 16 – 12 = 12	**0,17**	**42 %**
4	{ 6, 2, 1 }	12 + 8 – 3 = 17	**0,09**	**22 %**



Коэффициенты близости потомков:

№	Значение	Решение	K2	%
1	{ 1, 5, 4 }	2 + 20 – 12 = 10	**0,25**	**51 %**
2	{ 4, 5, 2 }	8 + 20 – 6 = 22	**0,06**	**12 %**
3	{ 1, 5, 1 }	2 + 20 – 3 = 19	**0,08**	**16 %**
4	{ 4, 2, 1 }	8 + 8 – 3 = 13	**0,1**	**21 %**

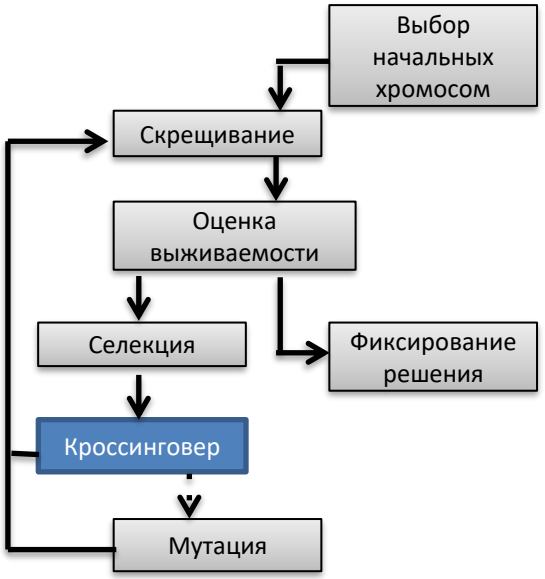
Генетические алгоритмы: кроссинговер

Номера родителей:

Родитель 1	Родитель 2
3	4
4	3
3	2
2	3

Гены родителей:

Родитель 1	Родитель 2
{ 4, 4, 4 }	{ 6, 2, 1 }
{ 6, 2, 1 }	{ 4, 4, 4 }
{ 4, 4, 4 }	{ 3, 6, 1 }
{ 3, 6, 1 }	{ 4, 4, 4 }



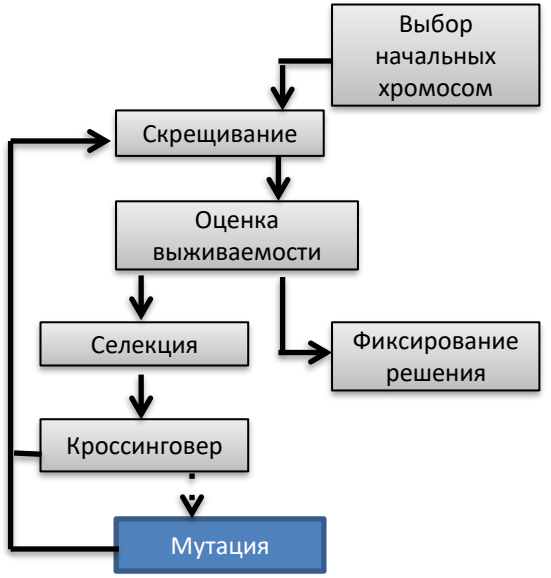
Кроссинговер:

Родитель 1	Родитель 2	Потомок
{ 4, 4, 4 }	{ 6, 2, 1 }	{ 4, 2, 1 }
{ 6, 2, 1 }	{ 4, 4, 4 }	{ 6, 2, 4 }
{ 4, 4, 4 }	{ 3, 6, 1 }	{ 3, 4, 1 }
{ 3, 6, 1 }	{ 4, 4, 4 }	{ 3, 4, 1 }

Генетические алгоритмы: варианты мутации

Варианты мутации:

- произвольная генерация нового гена;
- модификация имеющегося гена;
- модификация на основе статистических данных;
- ...



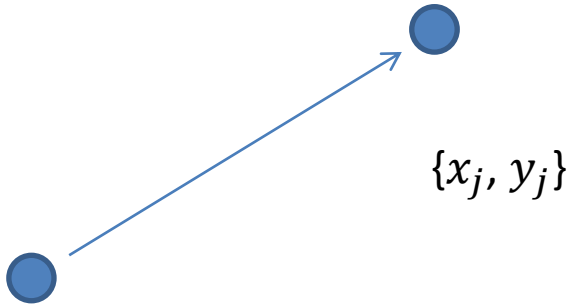
Результат кроссовера:

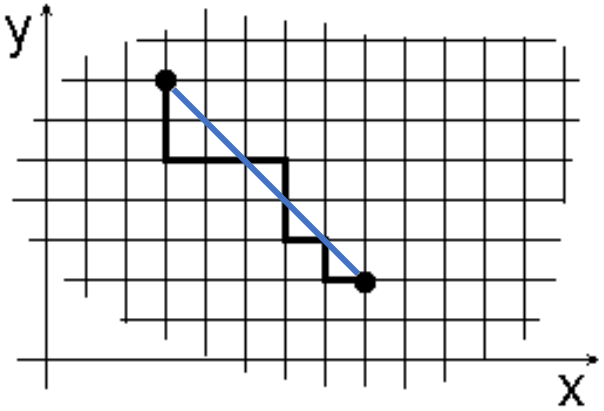
Родитель 1	Родитель 2	Потомок
{ **4**, 4, 4 }	{ 6, **2**, **1** }	{ 4, 2, 1 }
{ **6**, **2**, 1 }	{ **4**, 4, **4** }	{ 6, 2, 4 }
{ 4, **4**, 4 }	{ **3**, 6, **1** }	{ 3, 4, 1 }
{ **3**, 6, **1** }	{ 4, **4**, 4 }	{ 3, 4, 1 }

Результат мутации:

Родитель 1	Родитель 2	Потомок
{ **4**, 4, 4 }	{ 6, **2**, **1** }	{ 4, 2, 1 }
{ **6**, **2**, 1 }	{ **4**, 4, **4** }	{ 6, 2, 4 }
{ 4, **4**, 4 }	{ **3**, 6, **1** }	{ 3, 4, 1 }
		{ 1, 5, 2 }

Метрики измерения расстояний



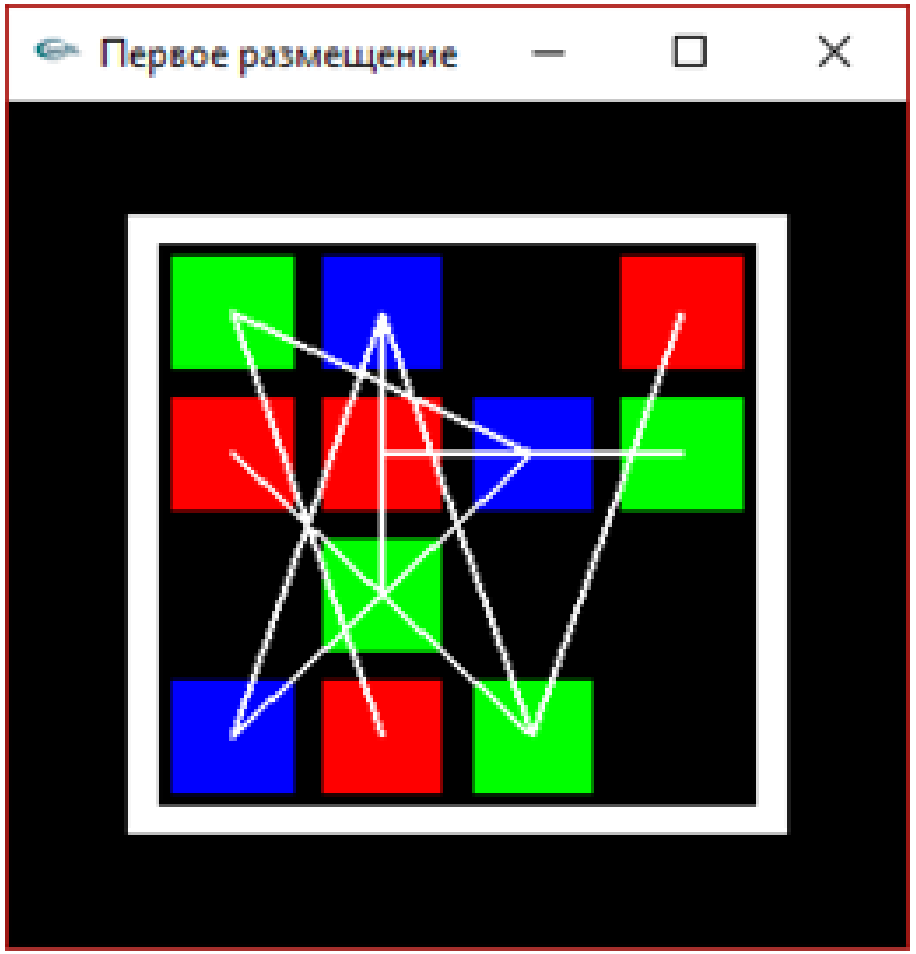


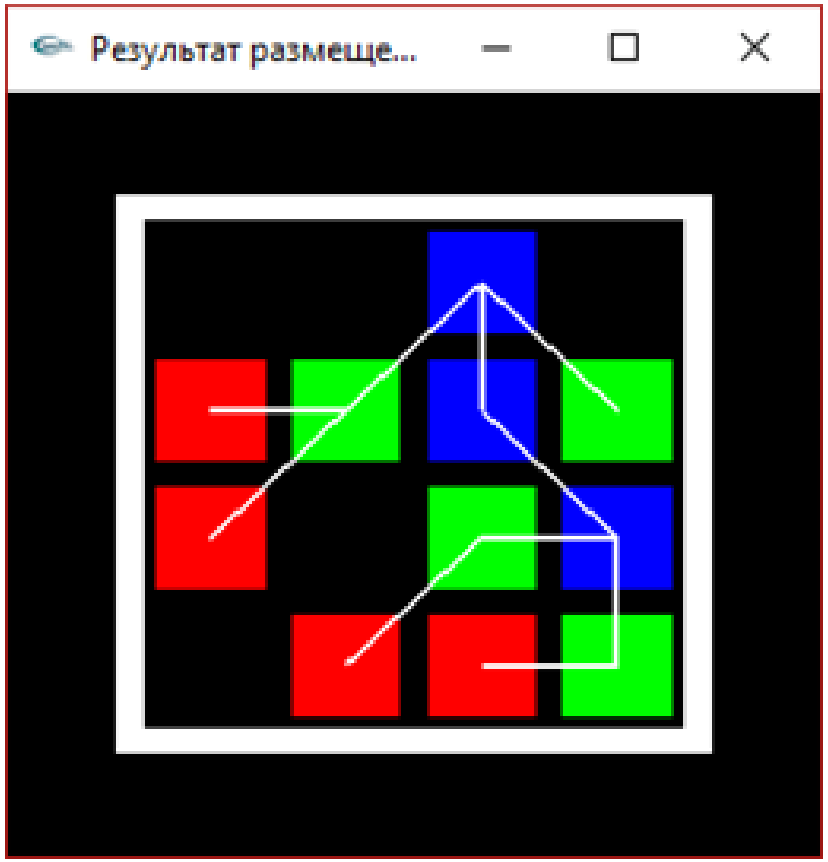
$$dEuclid = \sqrt[2]{(x_i - x_j)^2 + (y_i - y_j)^2}$$

$$dManhattan = |x_i - x_j| + |y_i - y_j|$$

ГА в САПР: решение задачи размещения

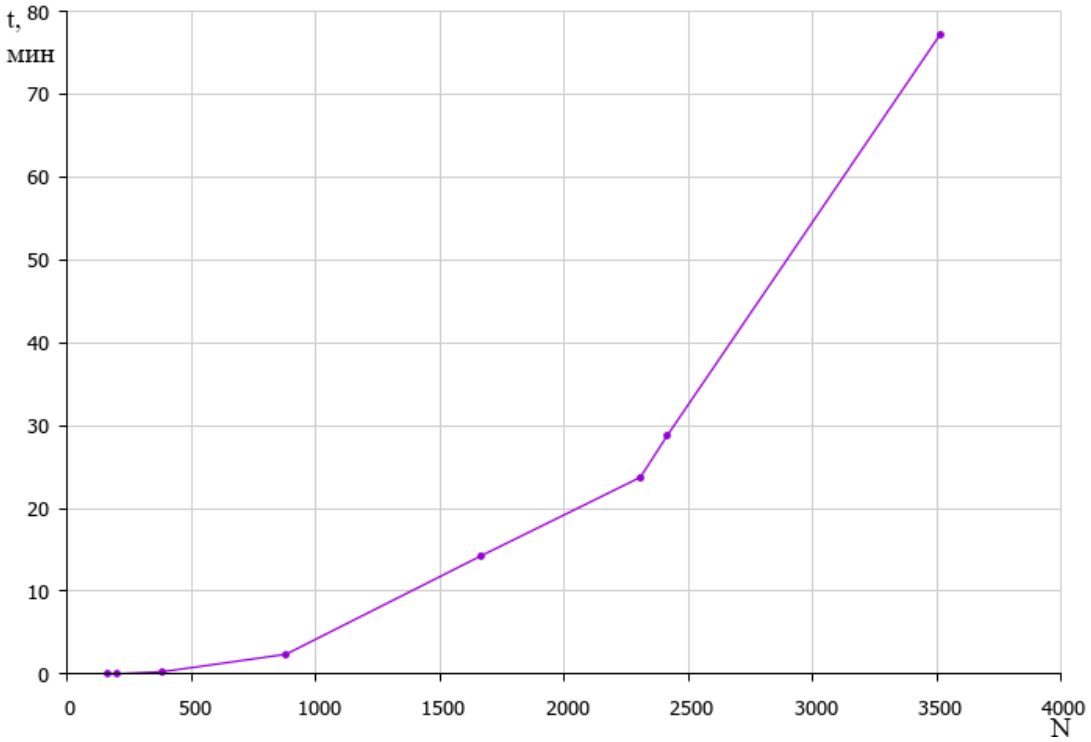
```
module test (out, in1, in2,
             in3, in4, in5,
             in6, in7, in8 );

      input in1, in2, in3, in4,
            in5, in6, in7, in8;

      output out;

      wire w1, w2, w3, w4, w5,
           w6, w7, w8, w9, w10;

      not not_1 (w1, in1);
      not not_2 (w2, in2);
      nand nand_1 (w6, in3, in4);
      not not_3 (w3, in5);
      nand nand_2 (w7, w3, in6 );
      not not_4 (w4, in7);
      nand nand_3 (w8, w4, in8);
      nand nand_4 (w5, w1, w2);
      nor nor_1 (w9, w5, w6);
      nor nor_2 (w10, w7, w8);
      nor nor_3 (out, w9, w10);

endmodule
```





ГА в САПР: характеристики

Зависимость времени работы от числа элементов



Наименование схемы	*ΔL при различном количестве итераций, %*				
	5	*10*	*15*	*20*	*25*
c432	2,58	5,37	6,36	8,88	15,10
c499	3,61	5,56	7,96	9,25	9,28
c880	4,84	5,93	6,64	8,44	8,52

Схема	$L_{начальное}$	$L_{конечное}$	*ΔL, %*	$S_{кристалла}$	*Количество элементов*	*Количество итераций*	*Время, мин*
c17	16	14	12, 5	9	6	6	0
c1908	61847	57575	6,91	900	880	10	2,42
c2670	87763	83282	5,11	1296	1269	9	4,03
c3540	177206	171465	3,24	1681	1669	10	14,35
c432	4858	4504	7,29	169	160	10	0,04
c499	10054	9416	6,35	225	202	10	0,06
c5315	343332	333220	2,95	2401	2307	10	23,78
c6288	354874	340392	4,08	2500	2416	10	28,84
c7552	475893	466838	1,9	3600	3513	10	77,17
c880	16173	15660	3,17	400	383	10	0,31

Алгоритм моделирования отжига (1)

Основные понятия :

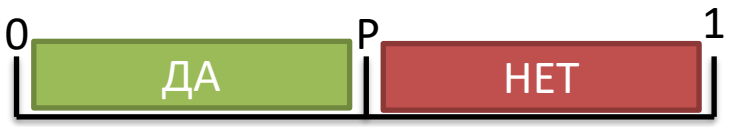
- S_i - состояние системы на i-той итерации;
- t_i - температура на i-той итерации;
- E_i - энергия системы на i-той итерации;
- E – функция подсчёта энергии системы;
- F – функция, порождающая новое состояние, $S_{i+1} = F(S_i)$;
- T- функция изменения температуры, $t_{i+1} = T(t_i)$.

Алгоритм моделирования отжига (2)

Входные данные для алгоритма:

- S_0 - начальное состояние системы;
- $t_0 = t_{max}$ - начальная температура системы;

Работа алгоритма:



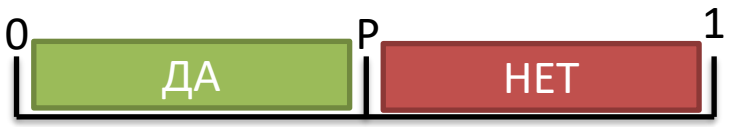
while $t_i > t_{min}$:
$S_{new} = F(S_i)$
$\Delta E = E(S_{new}) - E_i$
if $\Delta E \leq 0$:
$S_{i+1} = S_{new}$
else:
$P = e^{\frac{-\Delta E}{t_i}}$
if rand() < P:
$S_{i+1} = S_{new}$
$t_{i+1} = T(t_i)$

Алгоритм моделирования отжига (3)

Входные данные для алгоритма:

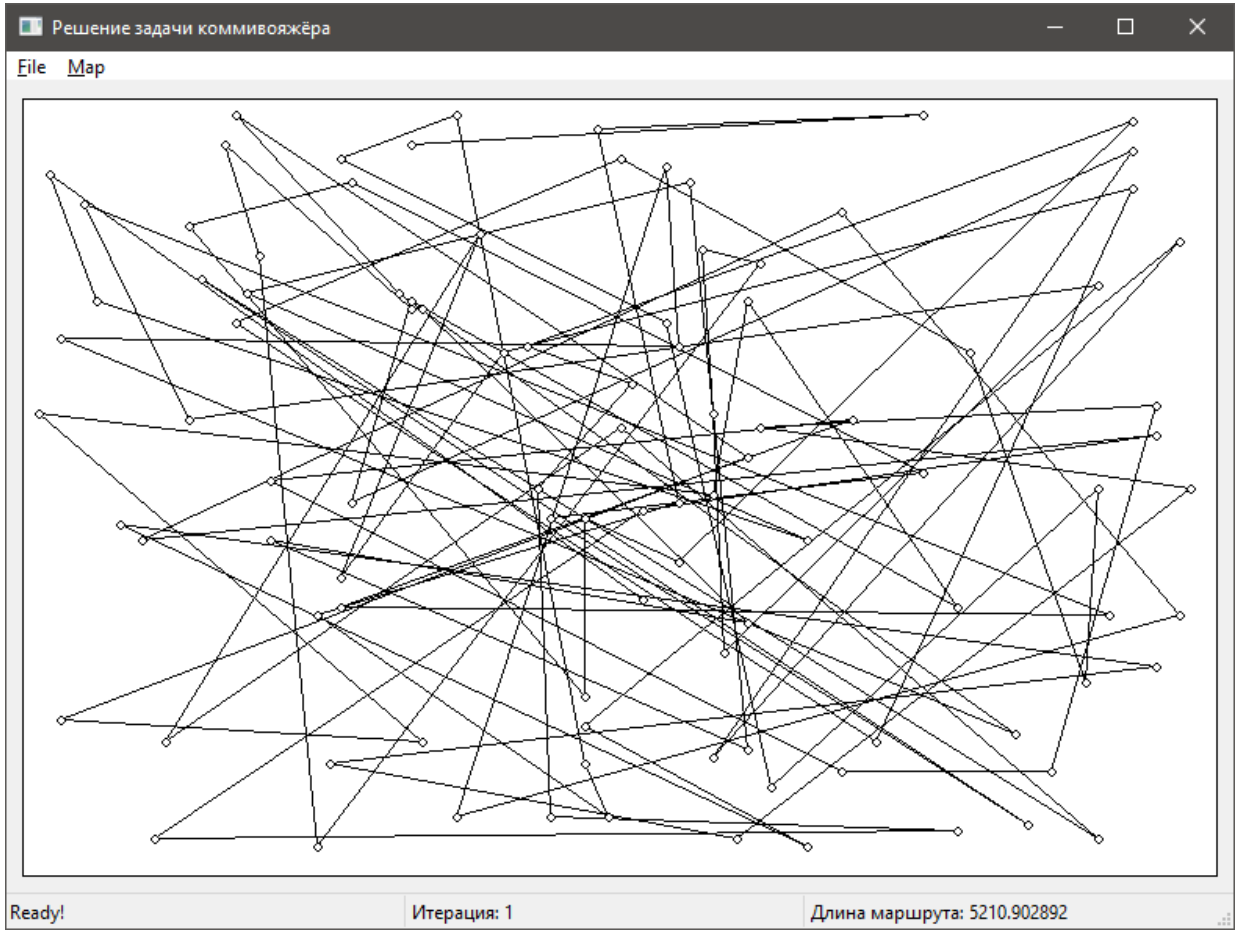
- S_0 - начальное состояние системы;
- $t_0 = t_{max}$ - начальная температура системы;

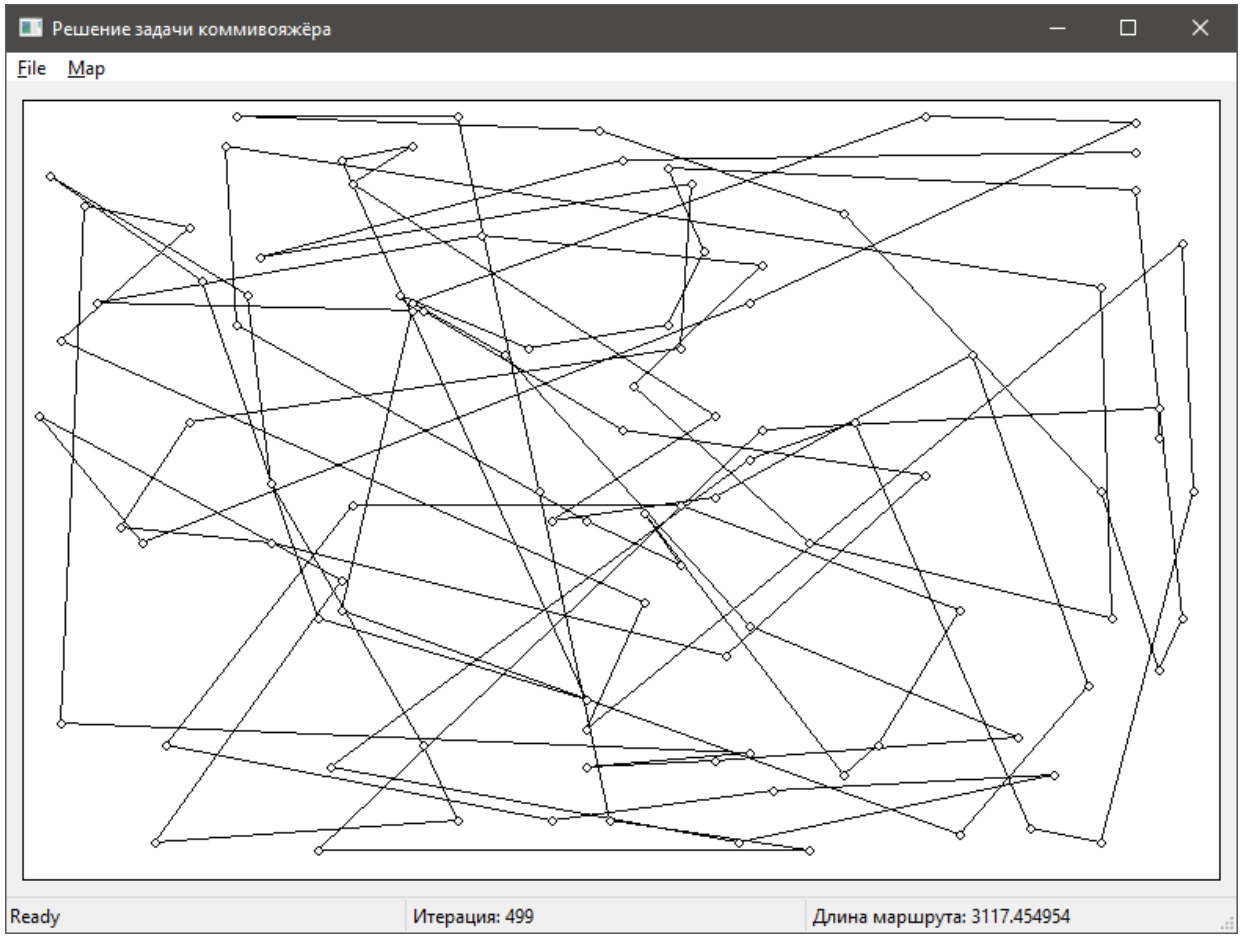
Работа алгоритма:



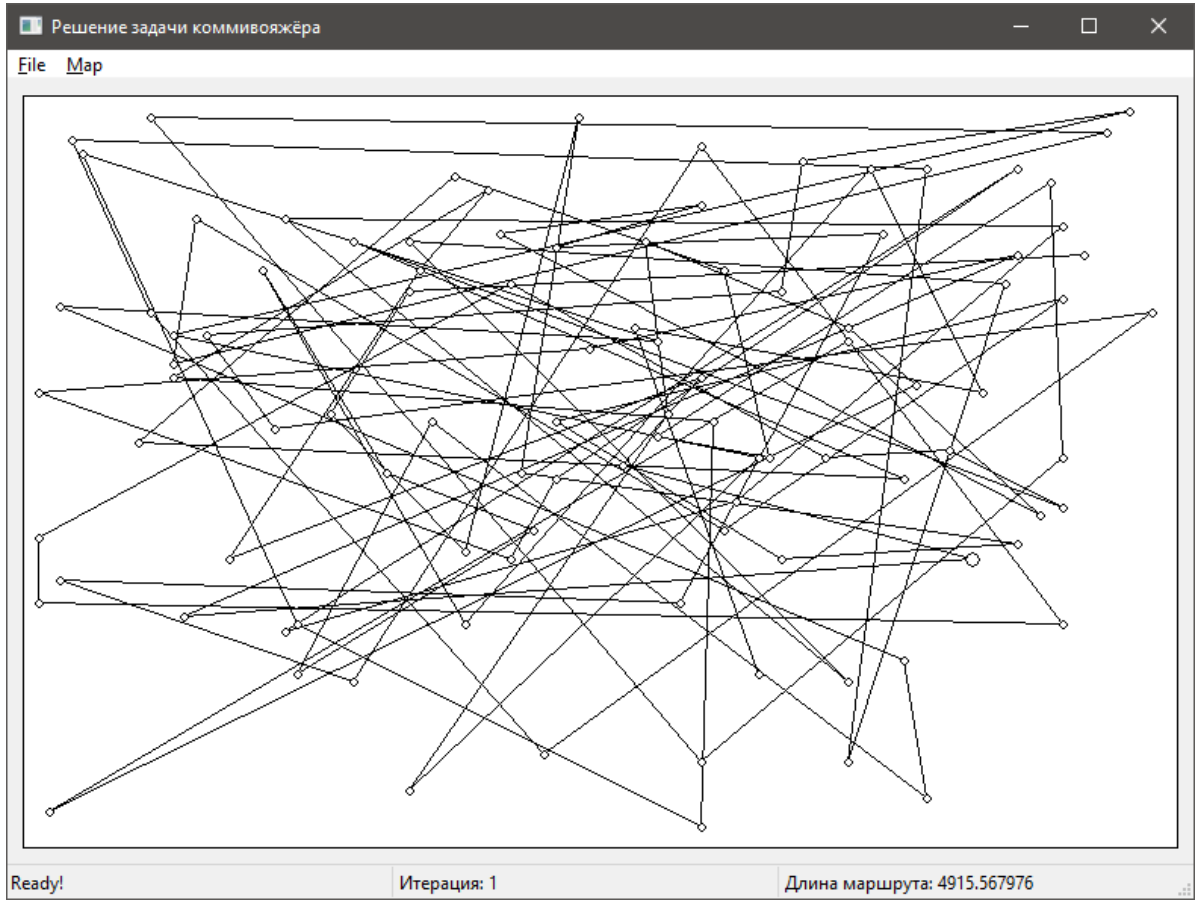
while $i < i\_max$:
$S_{new} = F(S_i)$
$\Delta E = E(S_{new}) - E_i$
if $\Delta E \leq 0$:
$S_{i+1} = S_{new}$
else:
$P = e^{\frac{-\Delta E}{t_i}}$
if rand() < P:
$S_{i+1} = S_{new}$
$t_{i+1} = T(t_i)$

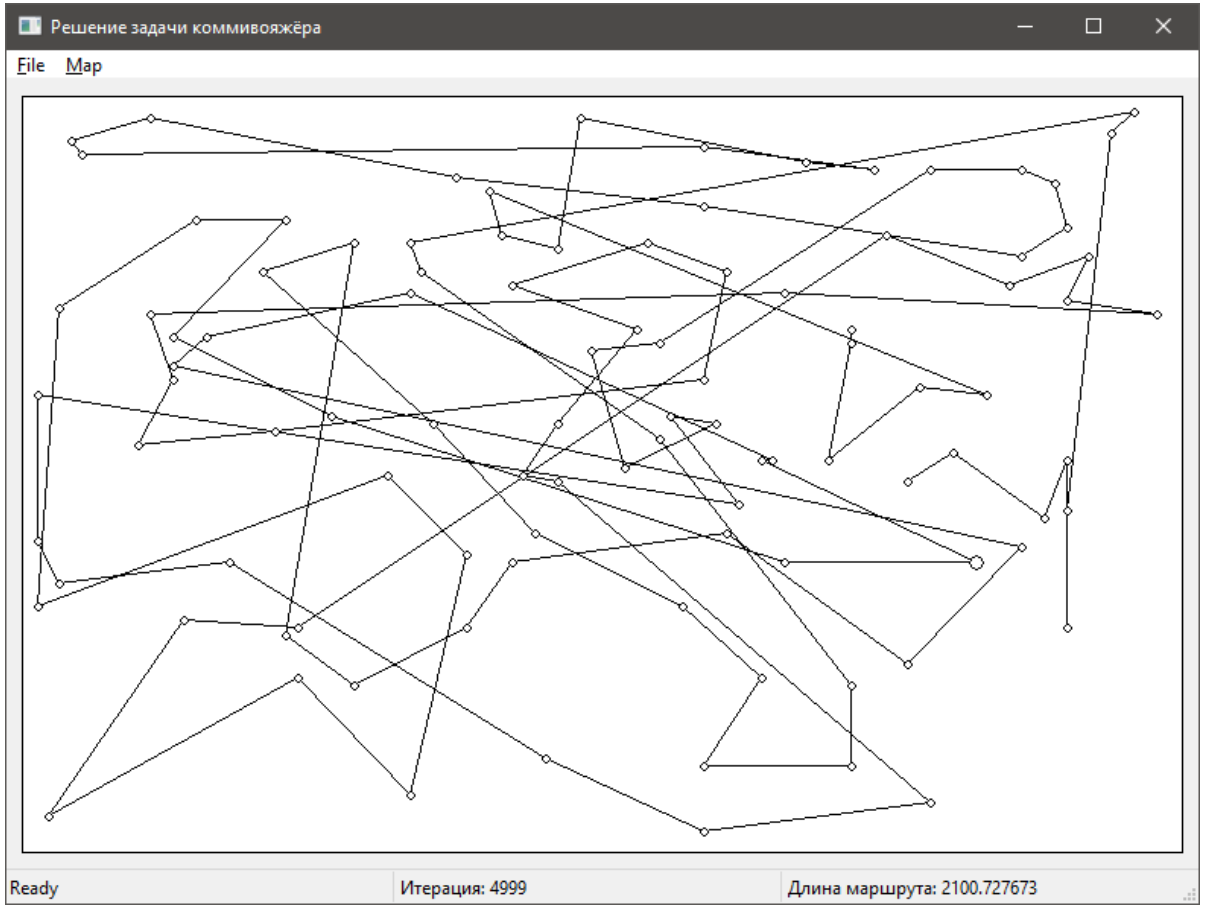
Сравнение исходного и конечного вариантов (1)





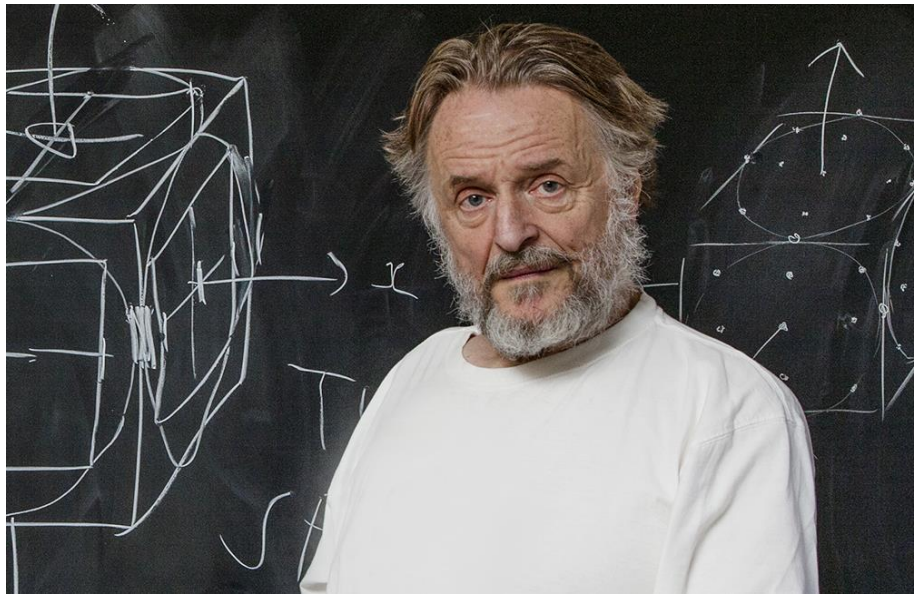
Сравнение исходного и конечного вариантов (2)





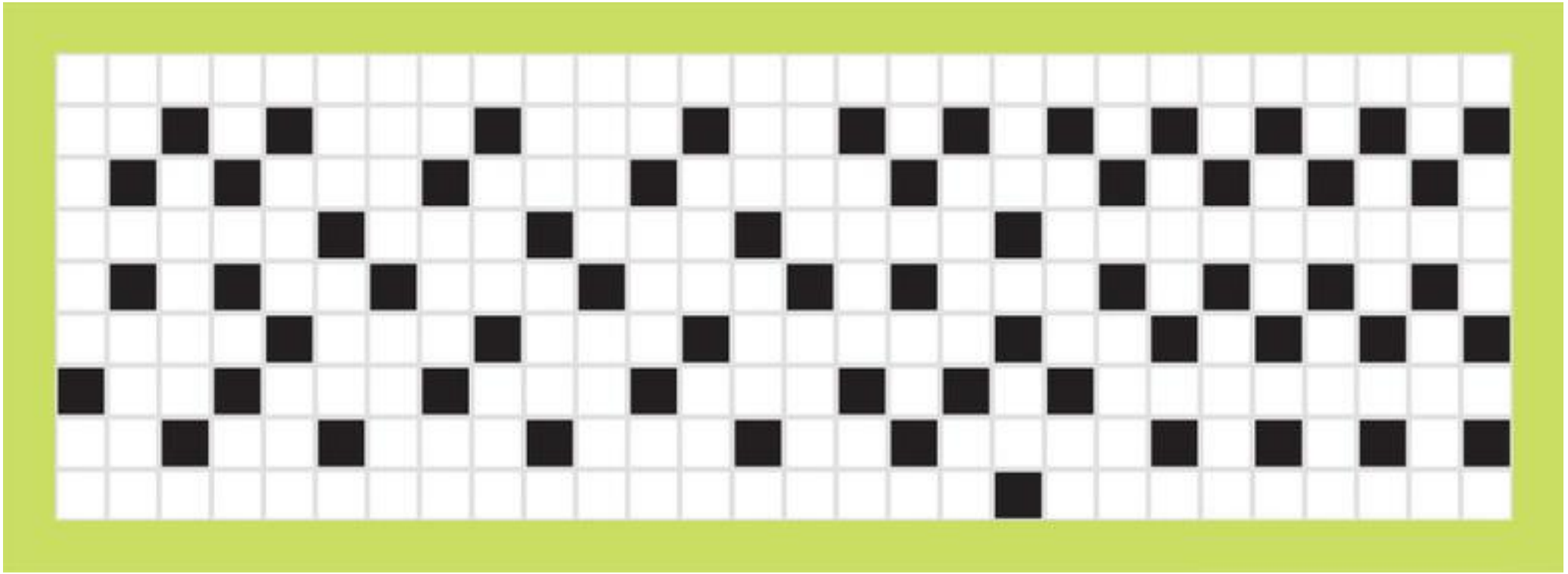
Клеточные автоматы: «Жизнь» Конвея

1. Клетки могут находиться в одном из двух состояний: «живая и «мёртвая».
2. Если вокруг живой летки менее двух соседей, она умирает от одиночества.
3. Если вокруг живой клетки более трёх соседей, она умирает от перенаселения.
4. Если вокруг мёртвой клетки ровно три соседа, она оживает.



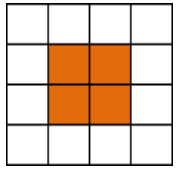
Джон Хортон Конвей,
Принстонский университет

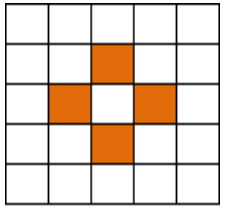
«Жизнь» Конвея: Эдемский сад

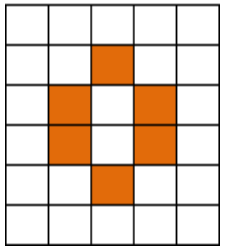


«Жизнь» Конвея: стабильные состояния и осцилляторы

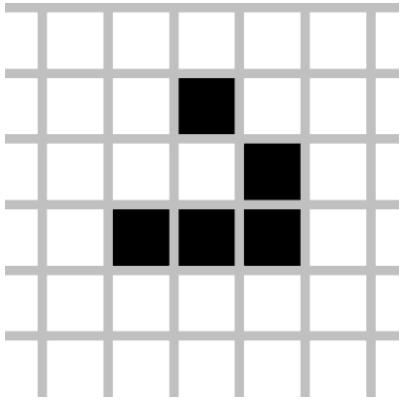
Устойчивые конфигурации:

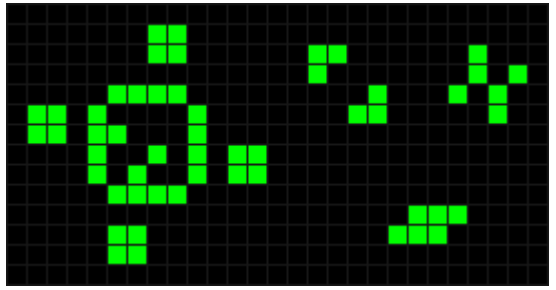






Осциляторы:





Клеточные автоматы: Wire World

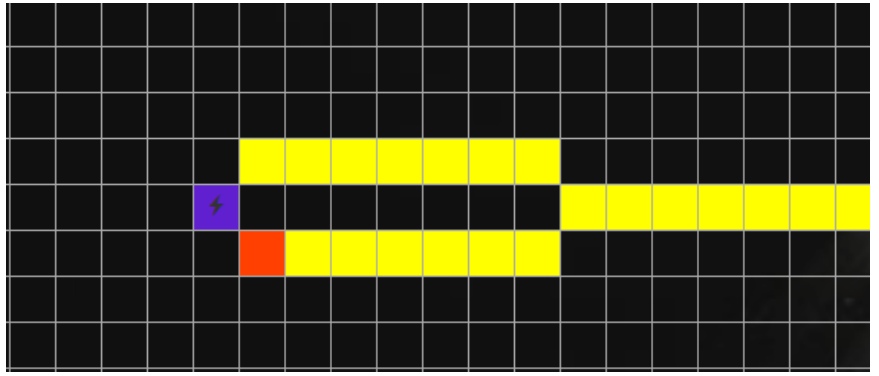
Клетка в Wireworld может находиться в одном из четырех состояний:

1. пустая;
2. голова электрона;
3. хвост электрона;
4. проводник.

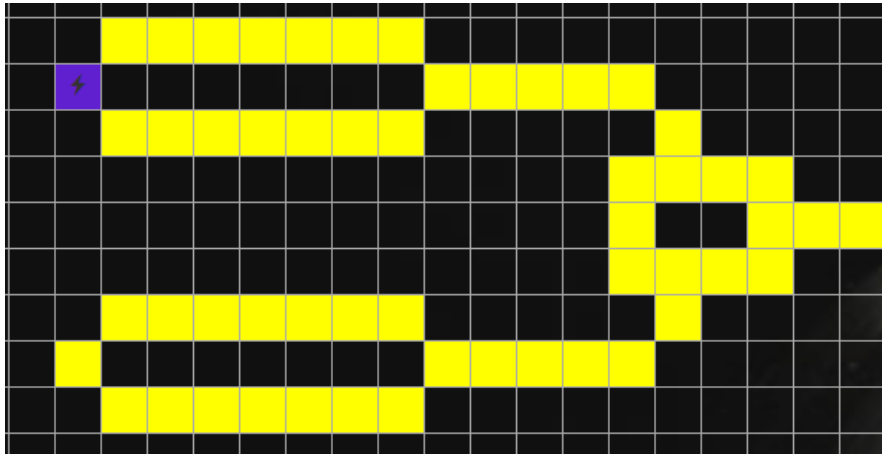
Клетки ведут себя следующим образом:

- пустая → пустая;
- голова электрона → хвост электрона;
- хвост электрона → проводник;
- проводник:
  → голова электрона при условии, что на соседних клетках есть ровно 1 или 2 головы электронов,
  → остаются проводниками в противном случае.

Генератор сигнала:

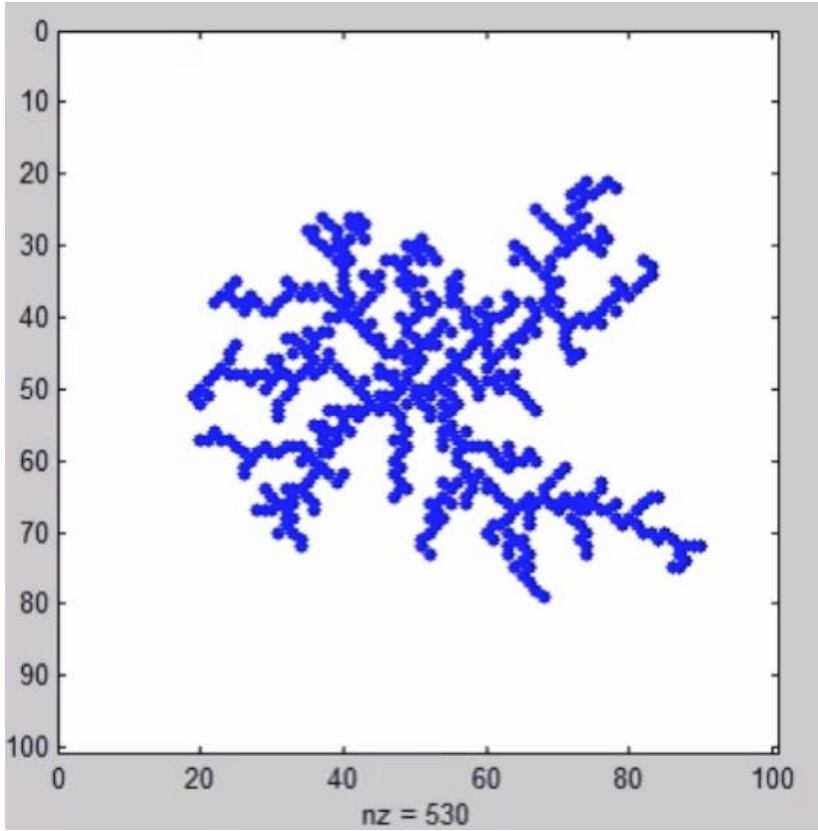


Логический вентиль XOR:



Клеточные автоматы: модель DLA

1. Частица появляется в произвольном месте ДРП.
2. Частица совершает броуновские движения до тех пор, пока не наткнётся на уже существующую частицу.
3. Частица «замораживается» в текущей позиции.
4. Повторяется пункт 1.



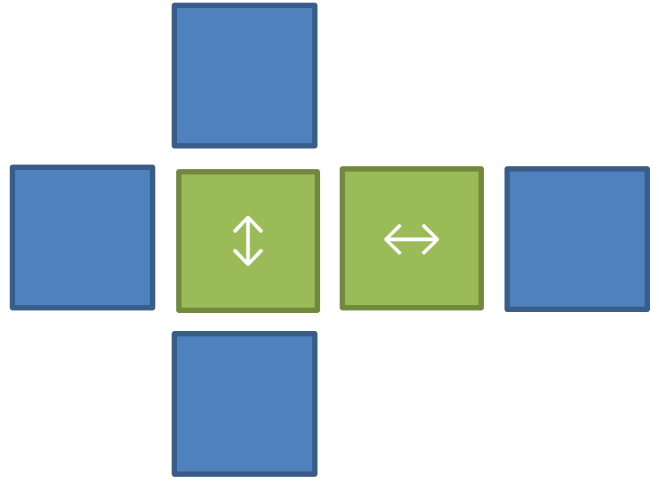
Клеточные автоматы: геометрический КА

Есть два типа клеток:

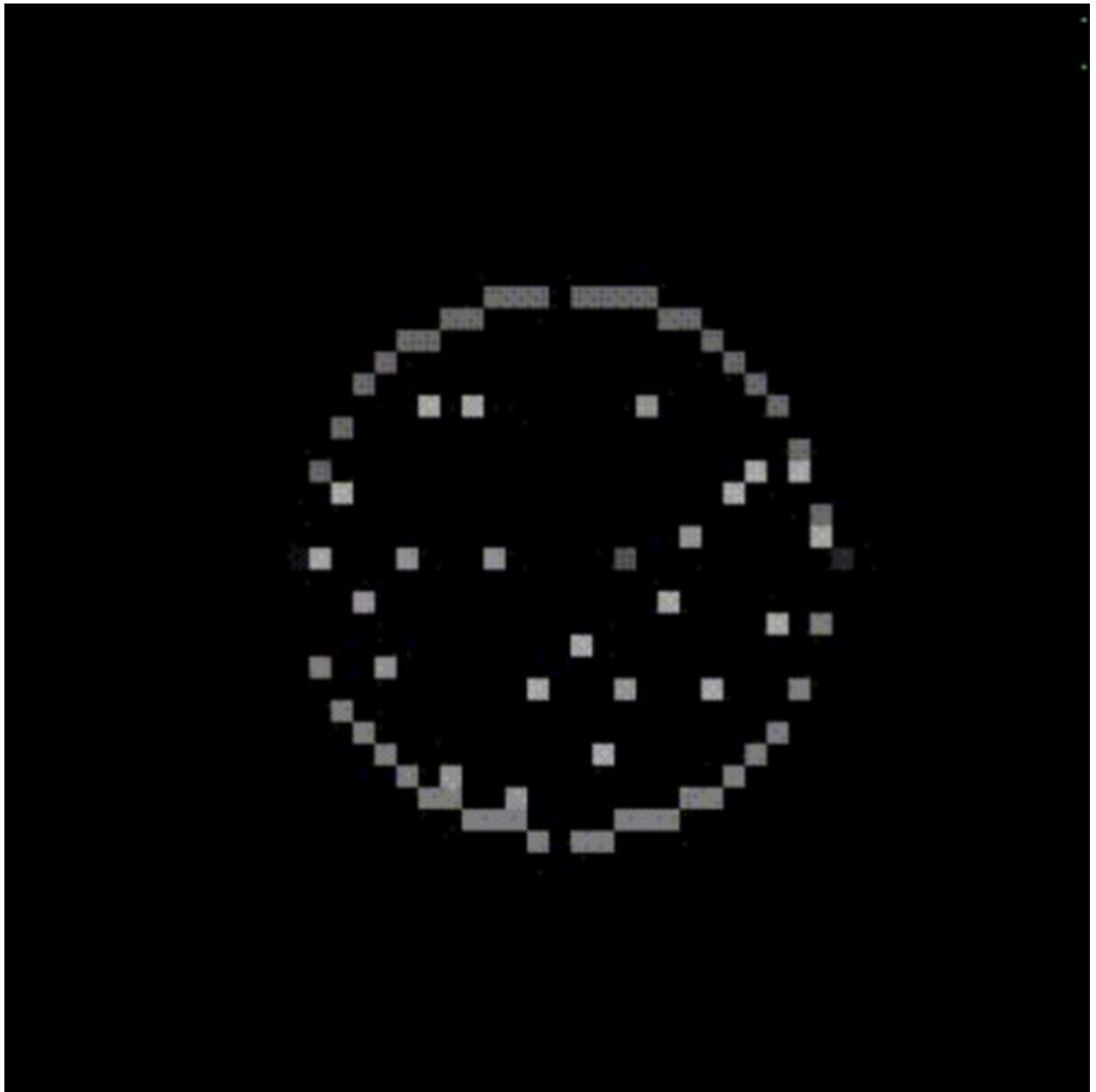
1. «фиксированные» клетки, нет никакого состояния, просто занимают клетку ДРП;
2. «бегающие» клетки, имеют одно из двух состояний – «бежать в одну сторону» и «бежать в другую сторону»;
3. «бегающие» клетки не взаимодействуют между собой.

Правила:

1. есть две «фиксированные» клетки;
2. между ними размещается «бегающая» клетка, выбирается произвольное направление движения;
3. когда «бегающая» клетка натыкается на «фиксированную», она:
   - а. толкает «фиксированную» клетку на 1 ячейку;
   - б. порождает сверху и снизу от себя по одной «фиксированной» клетке;
   - в. порождает на своём месте «бегающую» клетку, которая перемещается в перпендикулярном направлении;



Клеточные автоматы: геометрический КА (3)



Муравьиный алгоритм: обновление феромона

Формула обновления феромона:

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \sum_K \frac{Q}{L_K}$$

$\tau_{ij}(t)$ - количество феромона на ребре между i-тым и j-тым узлами в момент времени t;
ρ – интенсивность испарения, эвристический параметр;
Q – величина, имеющая значение порядка длины оптимального пути;
L_K – уже пройденное расстояние k-тым муравьём;

Муравьиный алгоритм: принцип работы

Задать значения эвристических параметров α, β, ρ, Q
Нанести на рёбра графа начальное значение феромона

Разместить муравьёв в требуемой вершине

Цикл по t - числу итераций алгоритма:
  Пока не все вершины посещены:
    Цикл по K - числу муравьёв:
      Рассчитать все ${P_{ij}}^K$
      Сгенерировать случайное p
      Выбрать путь для перемещения
    Обновить состояние феромона $\tau_{ij}(t+1)$

$$P_{ij}(t) = \frac{\tau_{ij}(t)^{\alpha} \left(\frac{1}{d_{ij}}\right)^{\beta}}{\sum_{j \in N\text{разр.узл.}} \tau_{ij}(t)^{\alpha} \left(\frac{1}{d_{ij}}\right)^{\beta}}$$

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \sum_K \frac{Q}{L_K}$$