Olga Słota

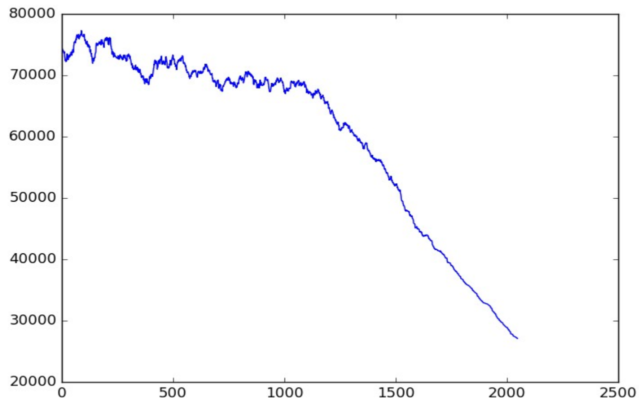
MowNiT2
Sprawozdanie

Lab 4 : Symulowane wyżarzanie

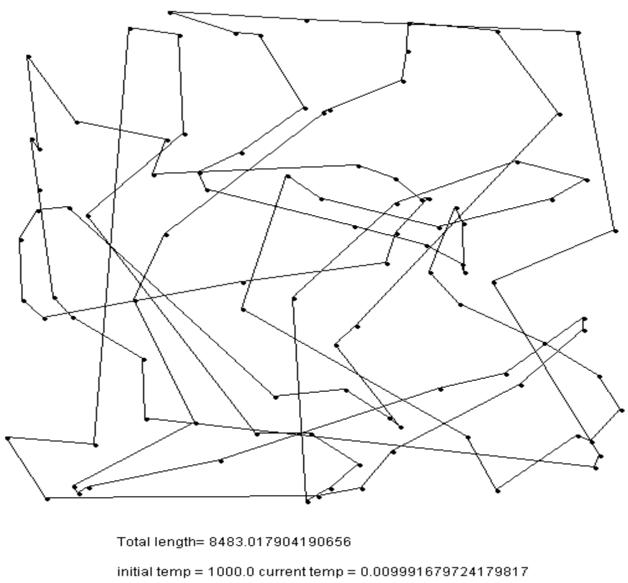
Zadanie 1: Problem Komiwojażera

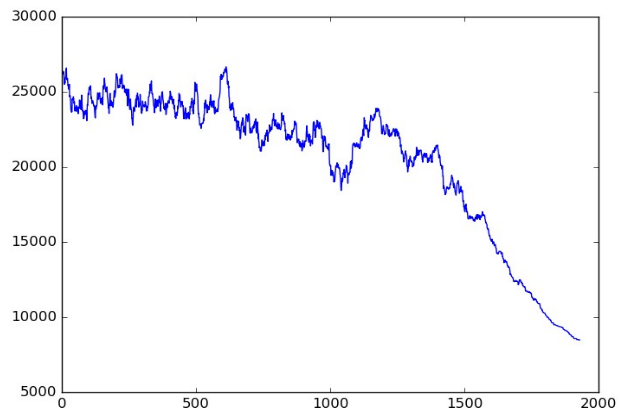
1. Rozkład jednostajny

1.1. $n = 300$, $T_0 = 1000$, $T_{i+1} = 0.999 * T_i$

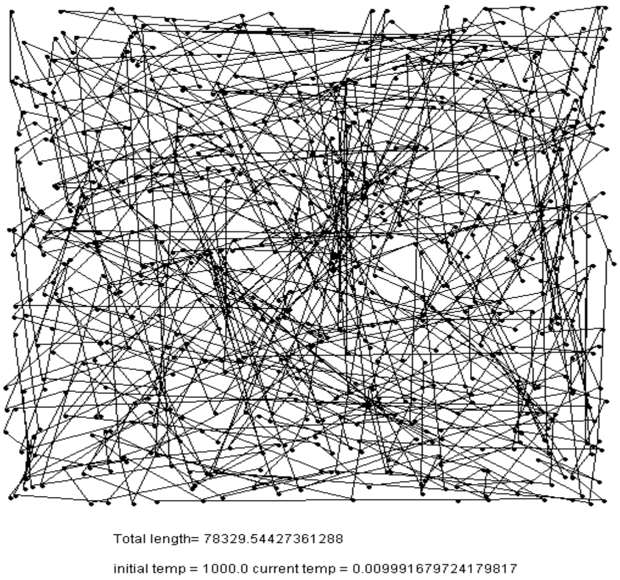


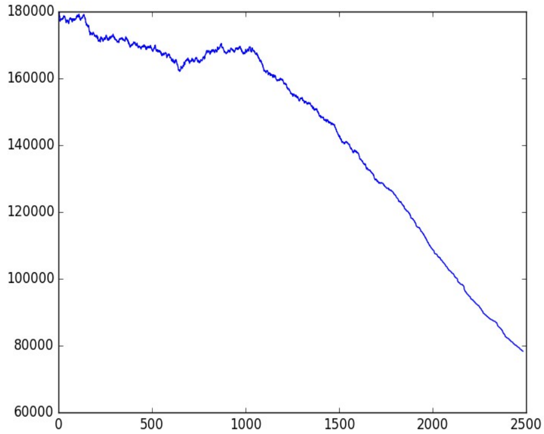
1.2. $n = 100$, $T_0 = 1000$, $T_{i+1} = 0.999 * T_i$





1.3. $n = 700$, $T_0 = 1000$, $T_{i+1} = 0.999 * T_i$

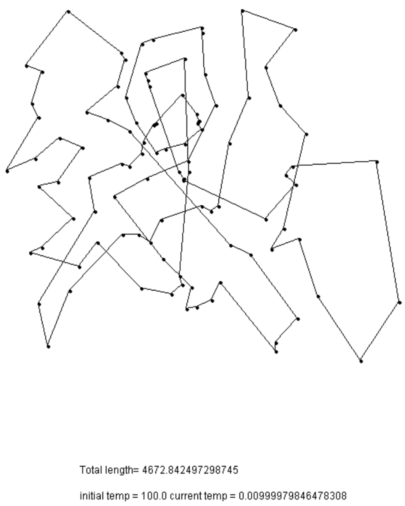


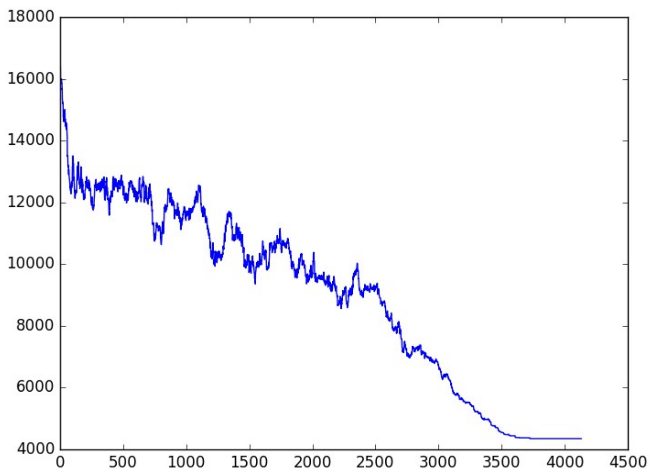


2. Rozkład normalny

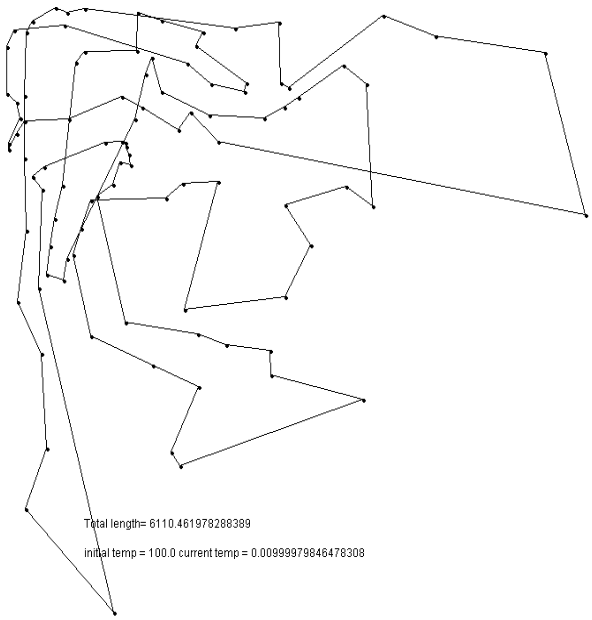
2.1. $n = 100$, $T_0 = 100$, $T_{i+1} = 0.9999 * T_i$

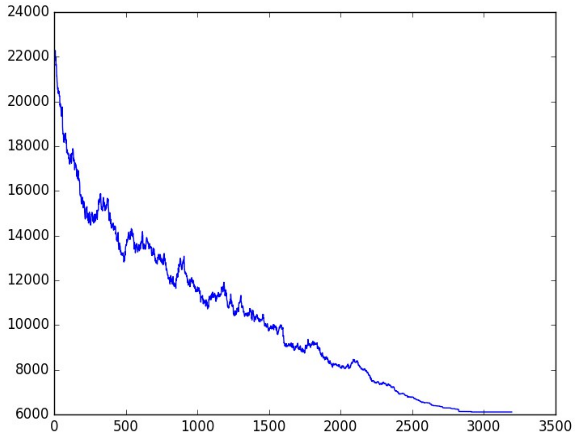
2.1.1. parametry: $\mu = 200$, $\sigma = 100$



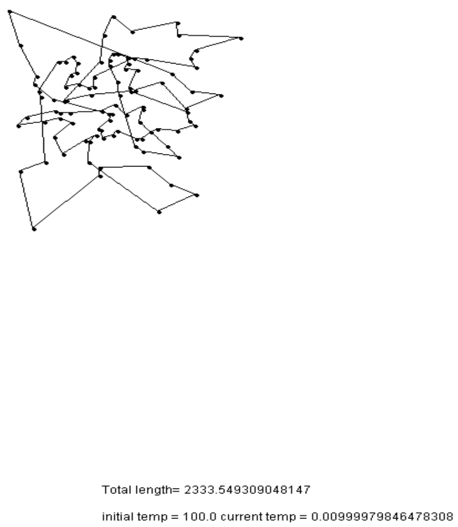


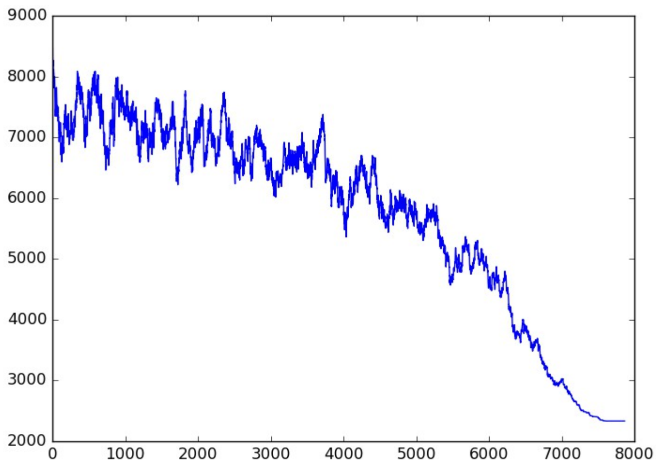
2.1.2. parametry: μ = 0, σ = 200



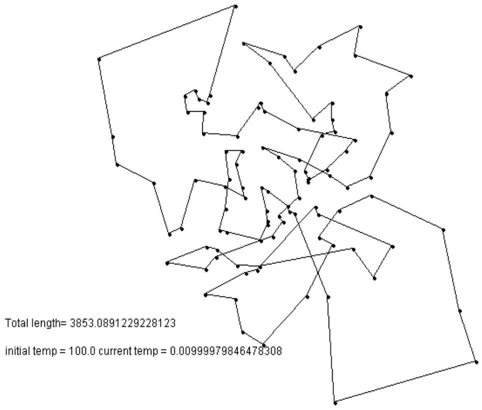


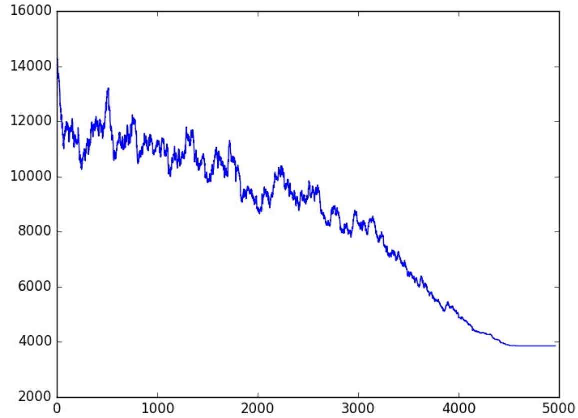
2.1.3. parametry: μ = 100, σ = 50





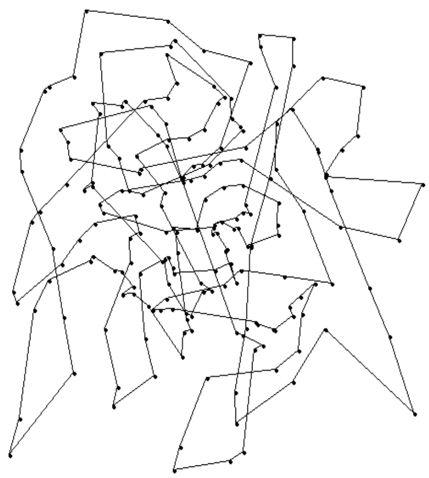
2.1.4. parametry: μ = 400, σ = 80

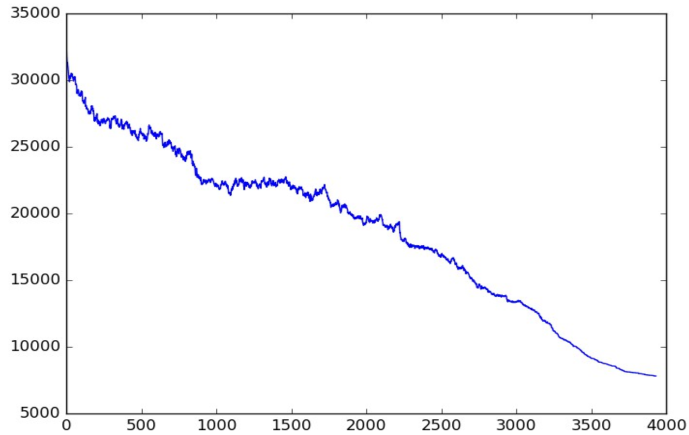




2.2. n = 200 , $T_0 = 100$, $T_{i+1} = 0.9999 * T_i$

2.2.1. parametry: μ = 200, σ = 100

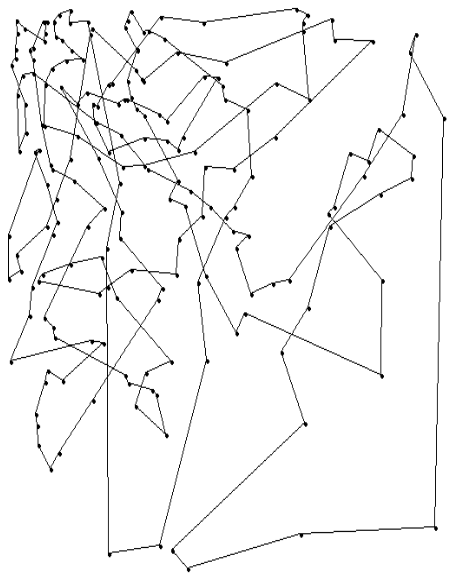


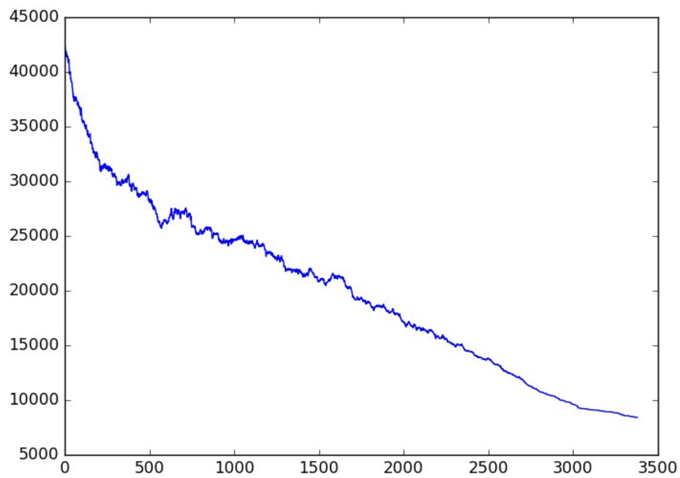


Total length= 7812.715934379186

initial temp = 100.0 current temp = 0.00999979846478308

2.2.2. parametry: μ = 0, σ = 200

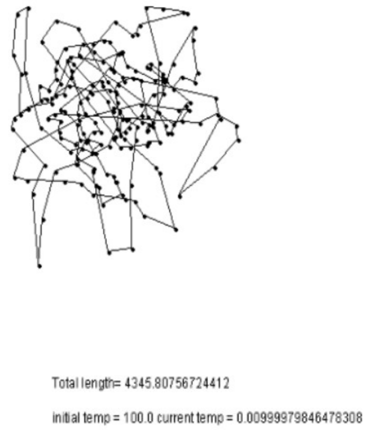


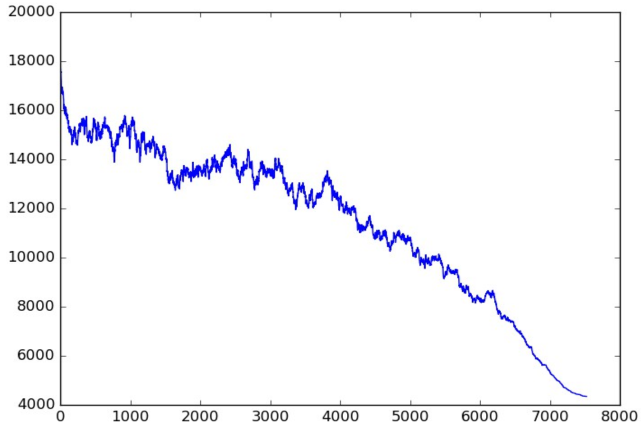


Total length= 8426.467215003508

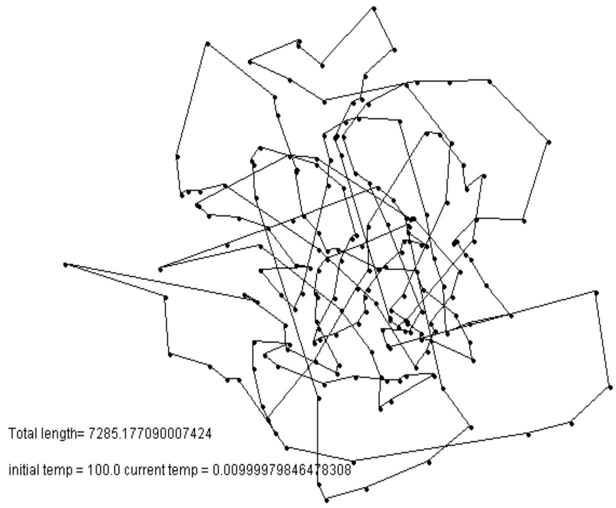
initial temp = 100.0 current temp = 0.00999979846478308

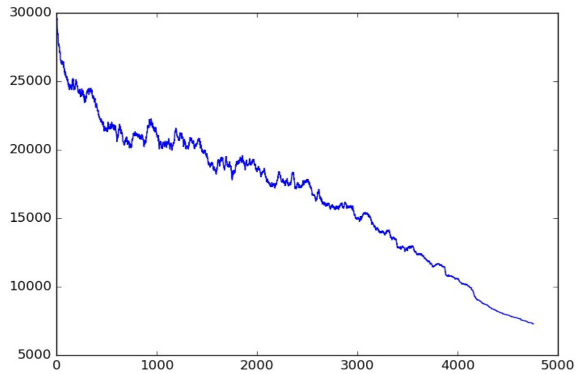
2.2.3. parametry: $\mu = 100$, $\sigma = 50$





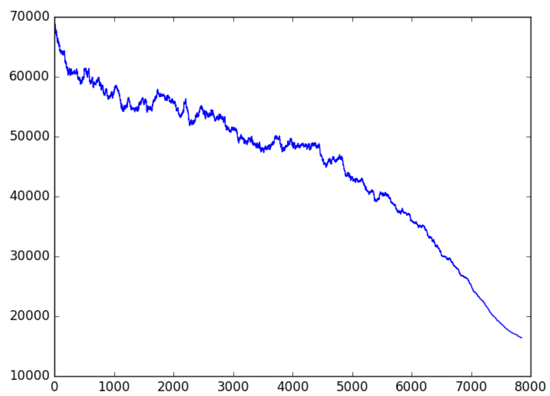
2.2.4. parametry: $\mu = 400$, $\sigma = 80$



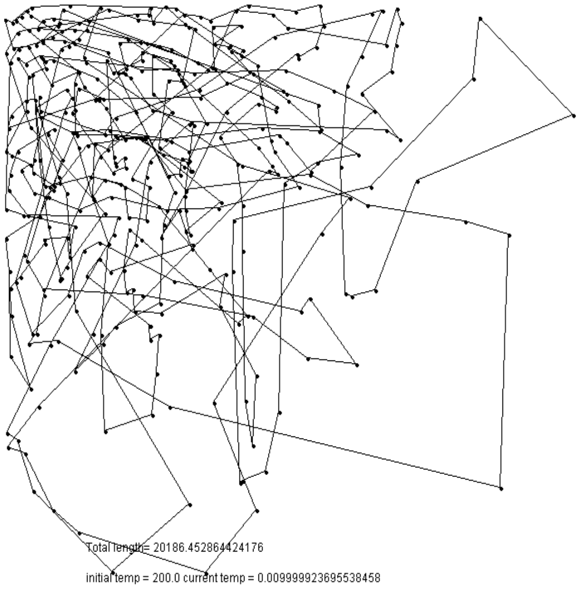


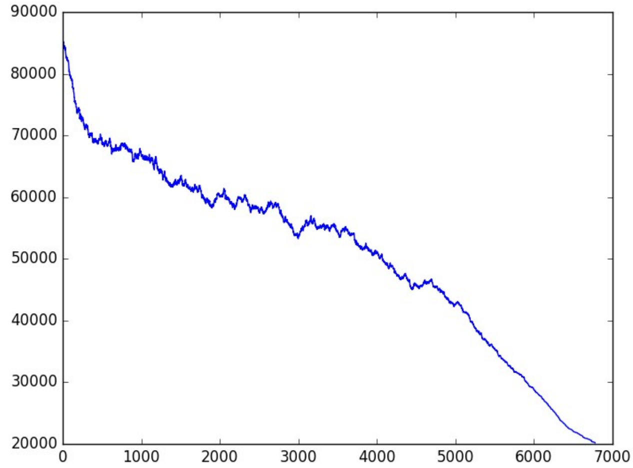
2.3. $n = 400$, $T_0 = 200$, $T_{i+1} = 0.9999 * T_i$

2.3.1. parametry: $\mu = 200$, $\sigma = 100$

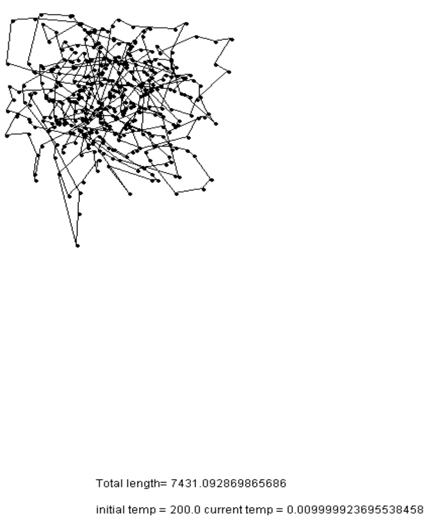


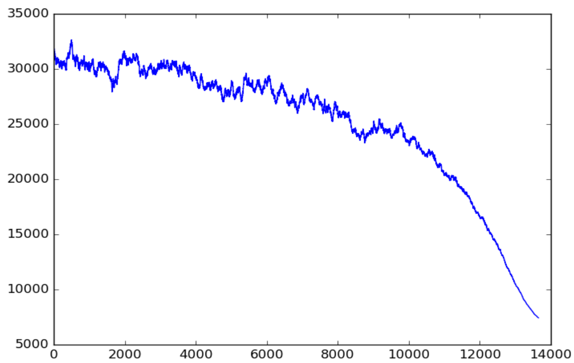
2.3.2. parametry: μ = 0, σ = 200



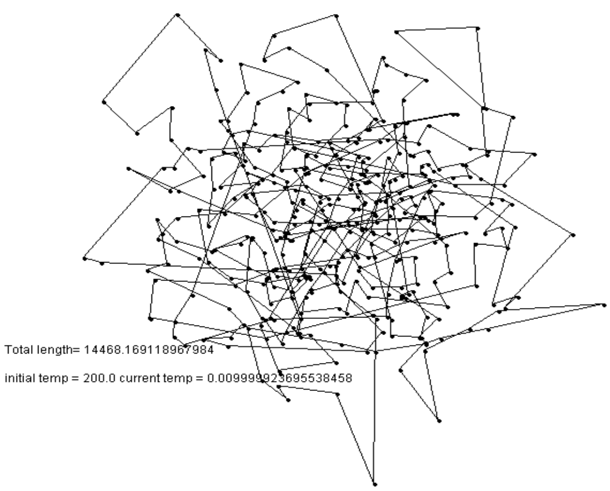


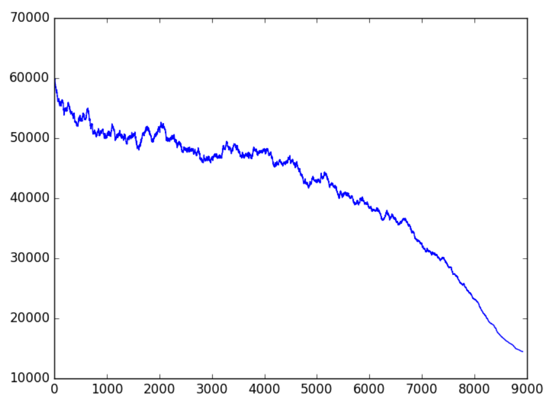
2.3.3. parametry: μ = 100, σ = 50





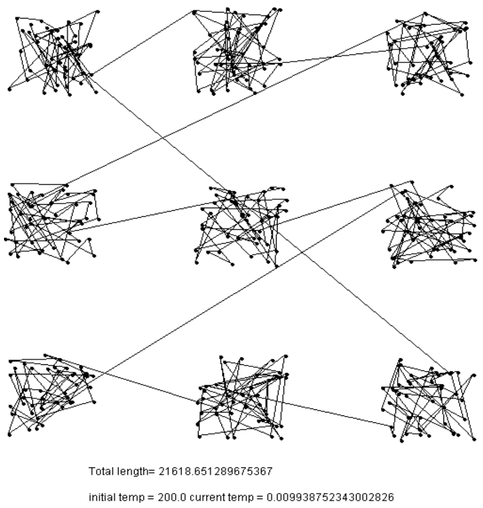
2.3.4. parametry: μ = 400, σ = 80

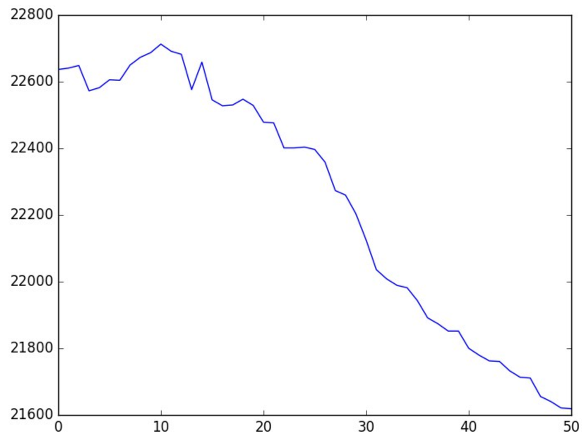




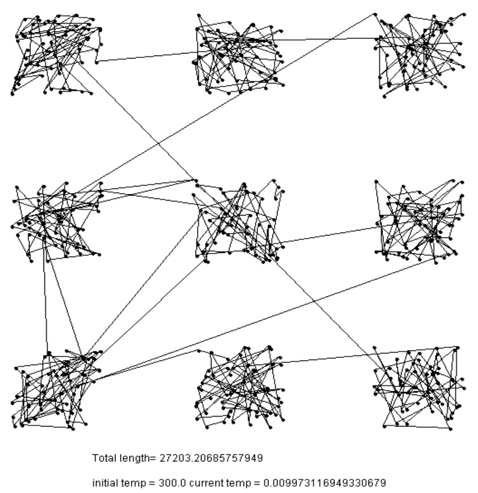
3. Odseparowane grupy punktów

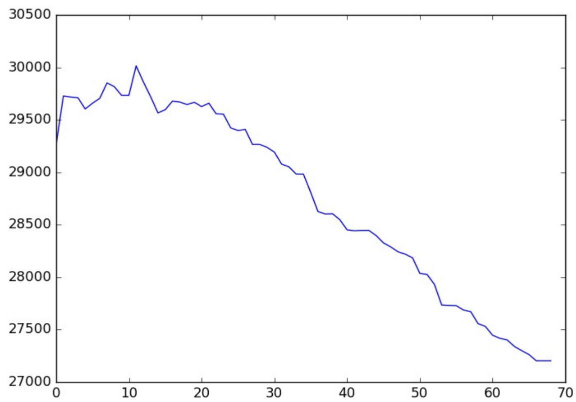
3.1. $n = 400$, $T_0 = 200$, $T_{i+1} = 0.99 * T_i$



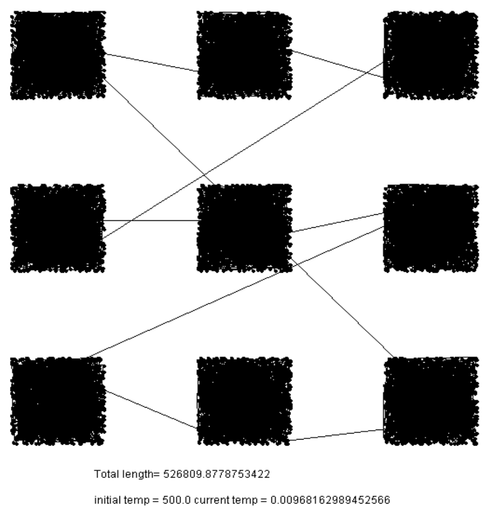


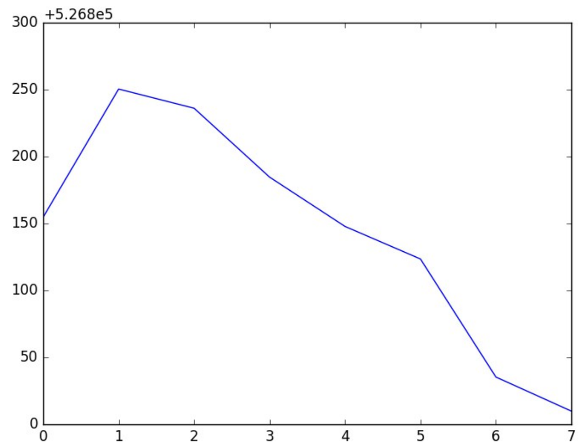
3.2. $n = 500$, $T_0 = 300$, $T_{i+1} = 0.99 * T_i$





3.3. $n = 10000$, $T_0 = 500$, $T_{i+1} = 0.9 * T_i$

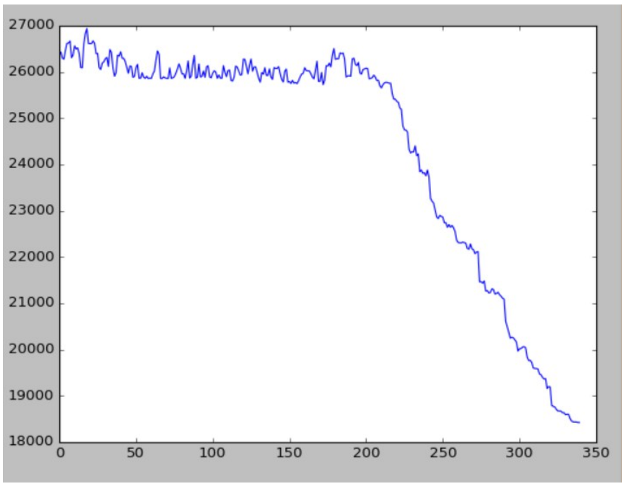




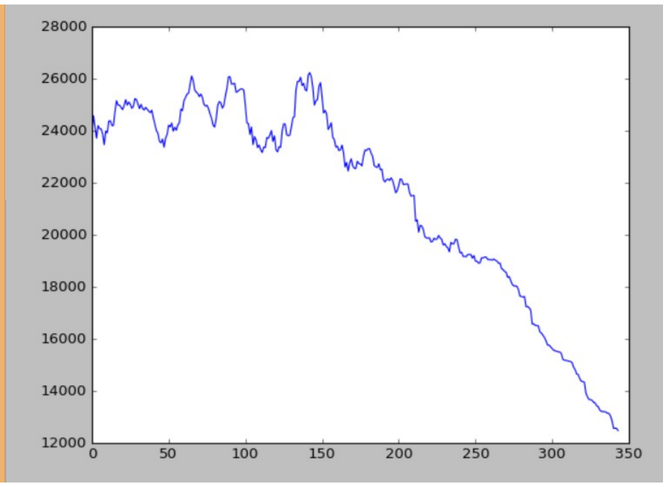
b)
Wpływ sposobu generacji stanu następnego (consecutive swap vs. arbitrary swap) na wynik:

Dla porównania przyjrzyjmy się wizualizacji przebiegu minimalizacji długości ścieżki dla tych samych parametrów wejściowych i dwóch sposobów generowania stanu następnego:

Consecutive swap :

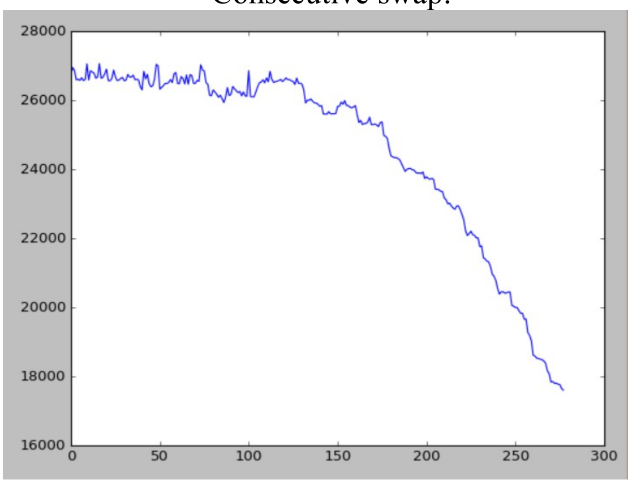


Arbitrary swap:

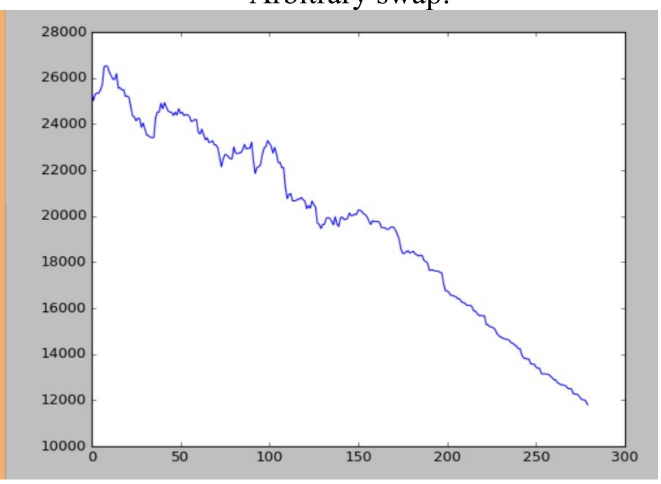


Jak widać wybór losowych punktów do zamiany powoduje bardziej gwałtowne skoki i w ostateczności szybciej prowadzi do lepszego rozwiązania niż w przypadku wyboru kolejnych punktów.
Jeszcze jeden przykład z innymi parametrami na potwierdzenie:

Consecutive swap:



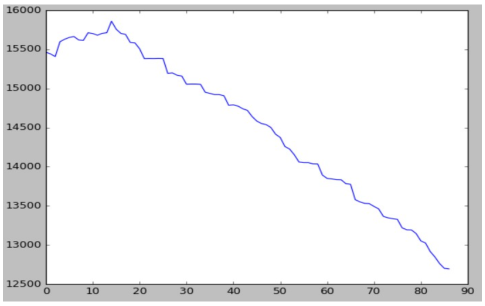
Arbitrary swap:



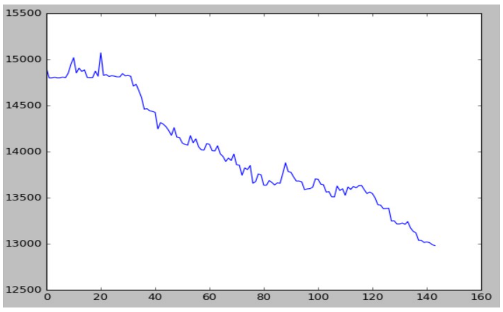
Wpływ funkcji schładzania na zbieżność procesu optymalizacji :

Porównanie funkcji wykładniczego schładzania (po lewej) z funkcją liniowego schładzania.
Są to dwa najprostsze warianty funkcji chłodzących.

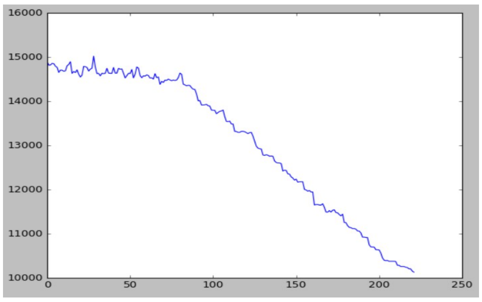
$T_{i+1} = 0.95 * T_i$



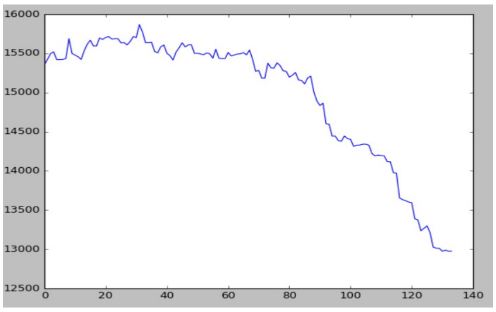
$T_{i+1} = T0 - 0.95 * i$



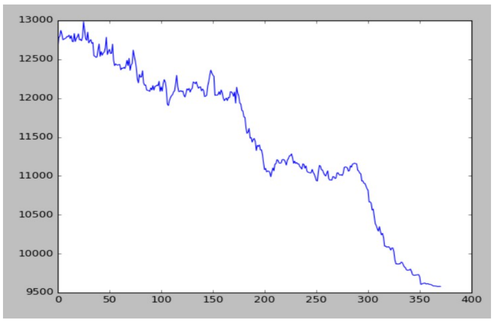
$T_{i+1} = 0.99 * T_i$



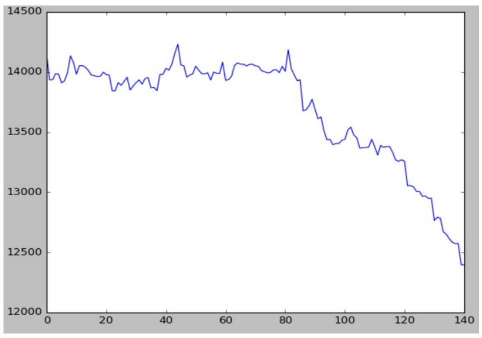
$T_{i+1} = T0 - 0.99 * i$



$T_{i+1} = 0.995 * T_i$



$T_{i+1} = T0 - 0.995 * i$



Z wykresów łatwo można odczytać szybszą zbieżność w przypadku schematu wykładniczego.

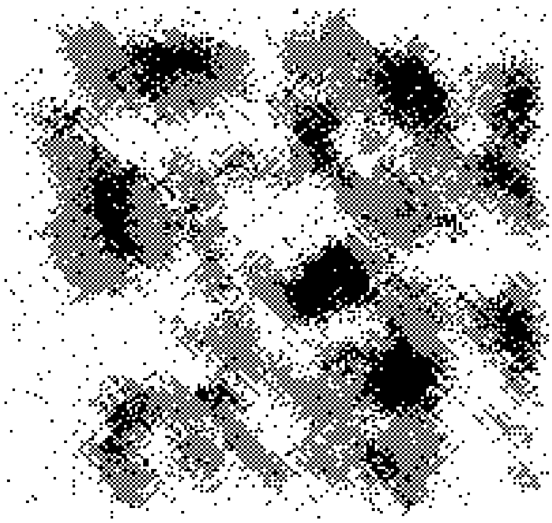
Zadanie 2 : Obraz binarny

Energia – punkty czarne przyciągają się energia między dwoma punktami odwrotnie proporcjonalna do odległości

1. gęstość punktów czarnych = 0.3

1.1. Sąsiedztwo diagonalne

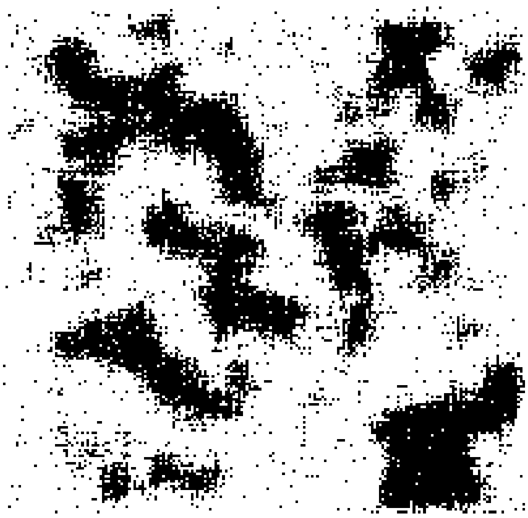
1.1.1. n = 300 , T_0 = 50000 , T_{i+1} = 0.9999 * T_i



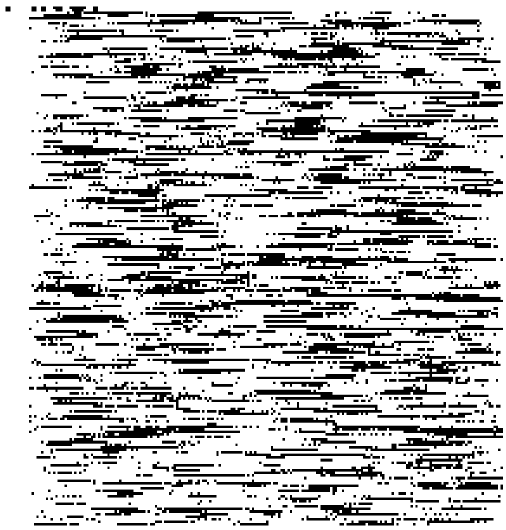
1.2. Sąsiedztwo krzyżykowe

1.2.1. n = 300 , T_0 = 500 , T_{i+1} = 0.99999 * T_i

Energia : punkty czarne przyciągają się w bliskiej odległości

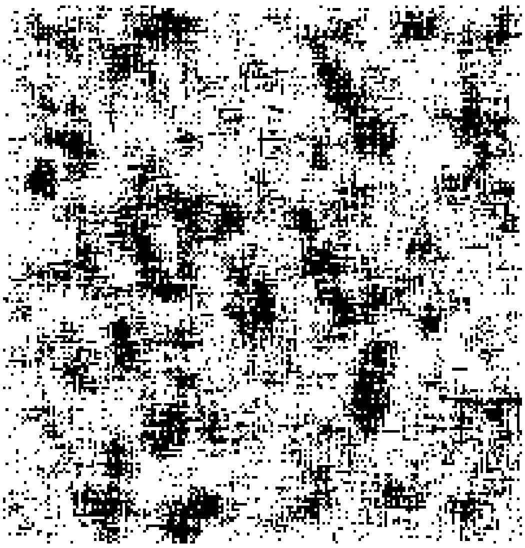


Energia: przyciąganie tylko po współrzędnej poziomej



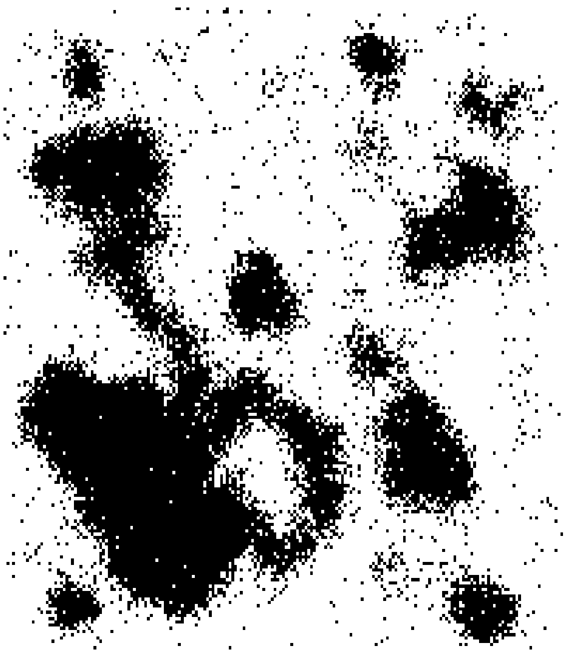
1.2.2. $n = 300$, $T_0 = 500$, $T_{i+1} = 0.9999 * T_i$

Energia : punkty czarne przyciągają się w bliskiej odległości



1.3. Sąsiedztwo ósemkowe

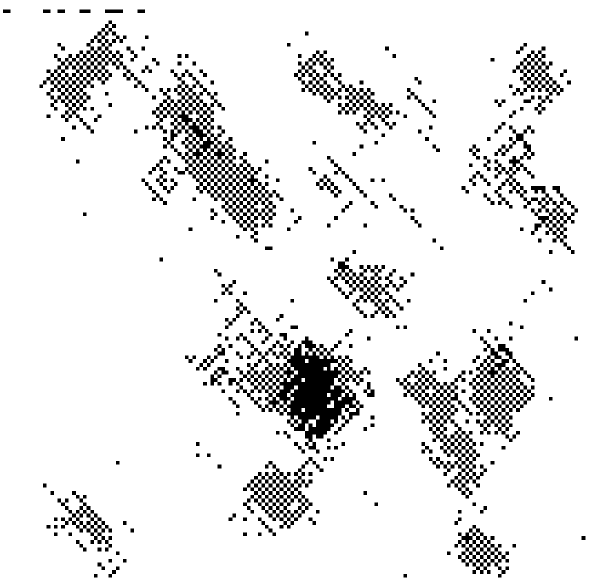
Energia – punkty czarne przyciągają się w bliskiej odległości



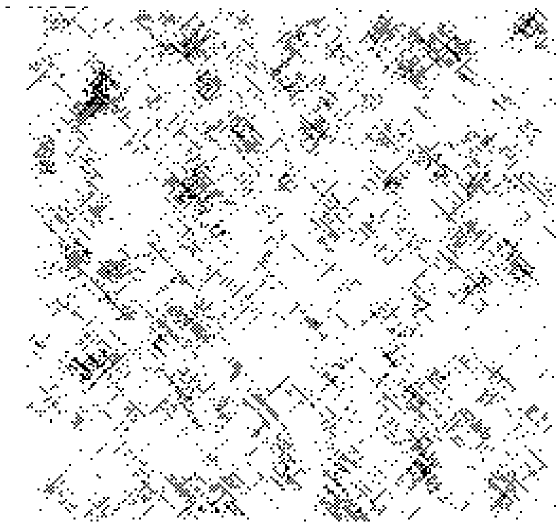
2. Gęstość punktów czarnych = 0.1
Energia – punkty czarne przyciągają się w bliskiej odległości

2.1.Sąsiedztwo diagonalne

$n = 300$, $T_0 = 30000$, $T_{i+1} = 0.99995 * T_i$

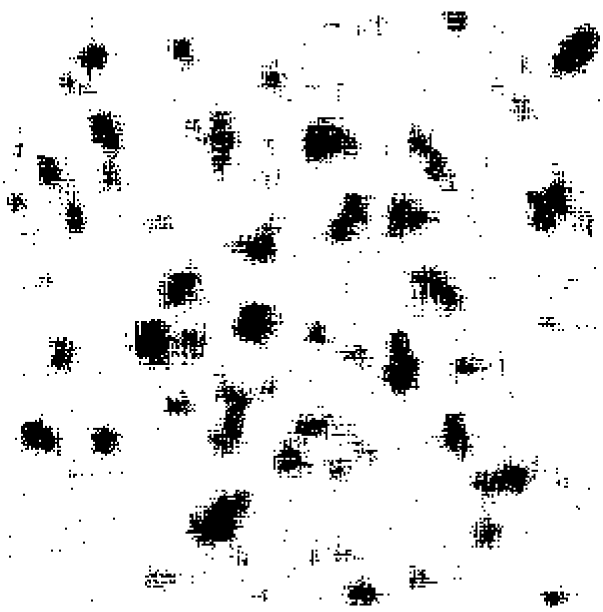


$n = 300$, $T_0 = 800$, $T_{i+1} = 0.9999 * T_i$

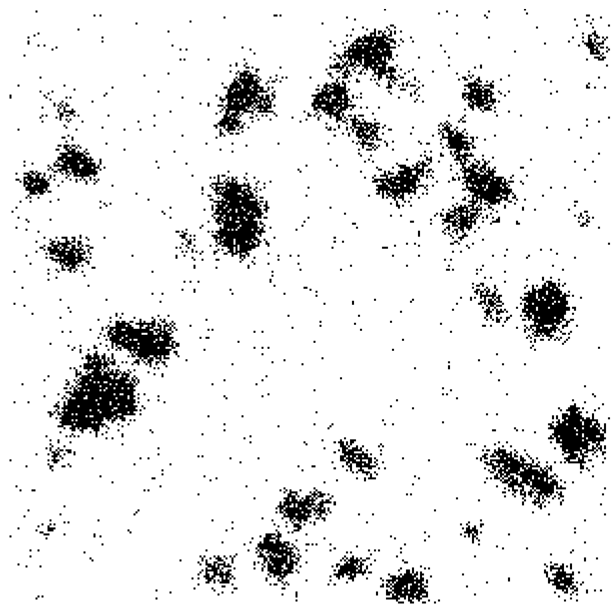


2.2. Sąsiedztwo krzyżykowe

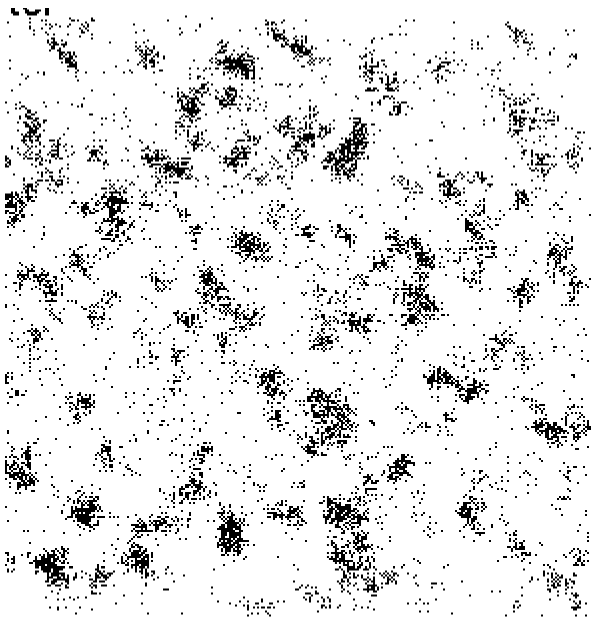
$n = 350$, $T_0 = 700$, $T_{i+1} = 0.9999 * T_i$



$n = 350$, $T_0 = 400$, $T_{i+1} = 0.9999 * T_i$

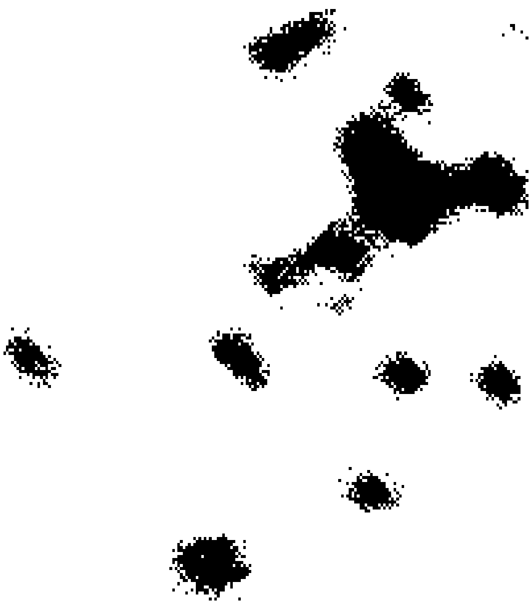


$n = 450$, $T_0 = 800$, $T_{i+1} = 0.9999 * T_i$

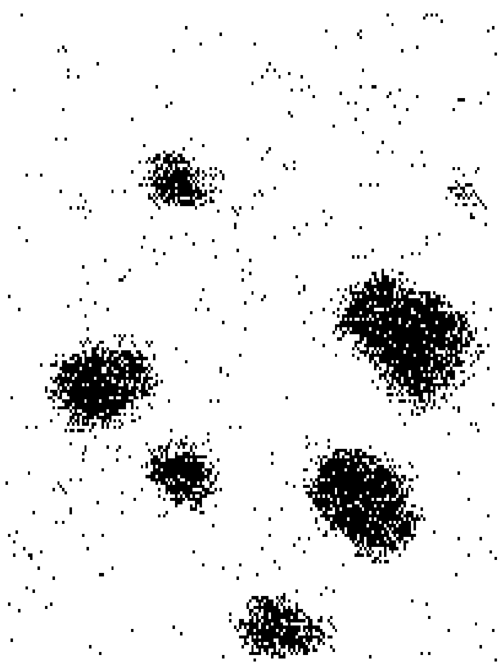


2.3. Sąsiedztwo ósemkowe

$n = 350$, $T_0 = 9000$, $T_{i+1} = 0.9999 * T_i$



$n = 350$, $T_0 = 5000$, $T_{i+1} = 0.9999 * T_i$



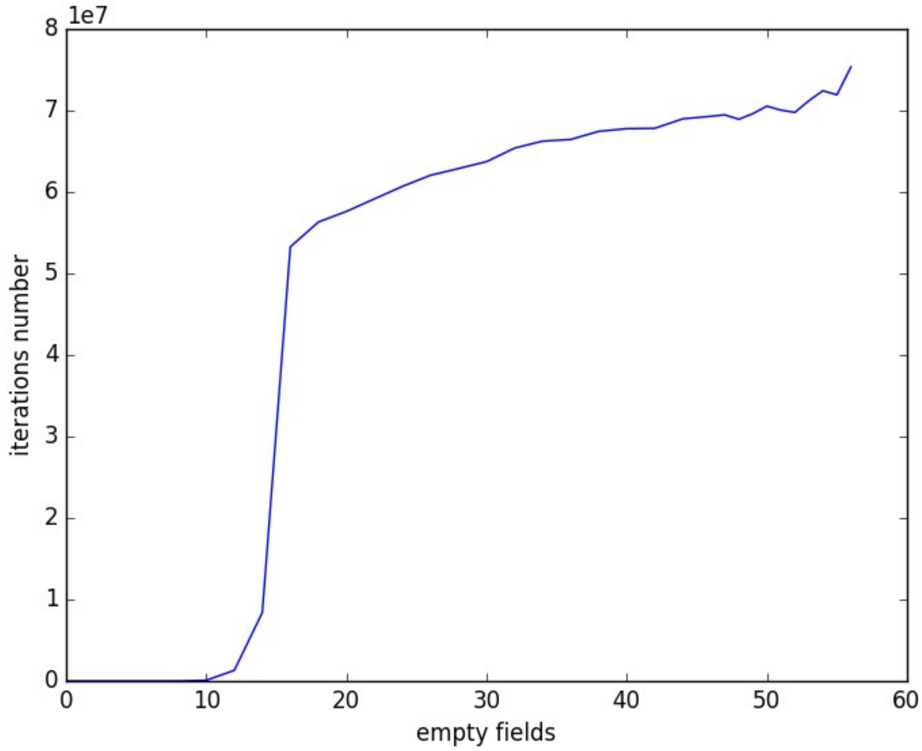
Zadanie 3 : Sudoku

Zastosowanie symulowanego wyżarzania do rozwiązania łamigłówki sudoku.

Po wczytaniu z pliku puste pole są wypełniane cyframi losowo, jeśli chodzi o pozycje, ale w odpowiedniej ilości każdej cyfry. Tzn np. w wejściowej planszy mamy w sumie 4 jedynki , to oznacza, że brakuje jeszcze pięciu, więc wpisujemy je w losowych (ale pustych) miejscach.

Generowanie kolejnego statu odbywa się poprzez zamianę losowych dwóch cyfr miejscami.

Wizualizacja zależności liczby potrzebnych iteracji od liczby pustych miejsc na planszy:



Algorytm nie rozwiąże na pewno planszy, która już przy wprowadzaniu z pliku jest nierozwiązywalna (także ręcznie). Jeśli natomiast wejściowa plansza jest poprawna i możliwa do rozwiązania ręcznie to algorytm po mniejszej lub większej liczbie iteracji radzi sobie z rozwiązaniem łamigłówki.