Metody Obliczeniowe w Nauce i Technice

Laboratorium 4 - Symulowane wyżarzanie
Sprawozdanie

Jakub Pinowski

2020-03-23

1 TSP - problem komiwojażera

Tematem pierwszego zadania było znalezienie przybliżonego rozwiązania problemu komiwojażera przy pomocy algorytmu symulowanego wyżarzania.

1.1 Wyniki działania algorytmu dla przykładowych danych

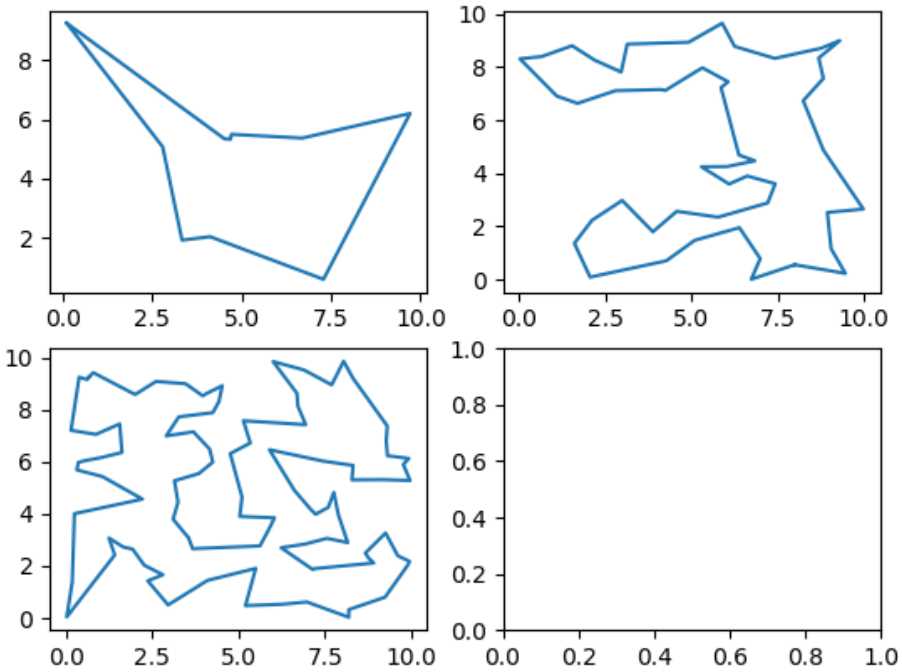

Figure 1: Punkty z realizacji rozkładu jednostajnego; kolejno 10, 50, 100 punktów

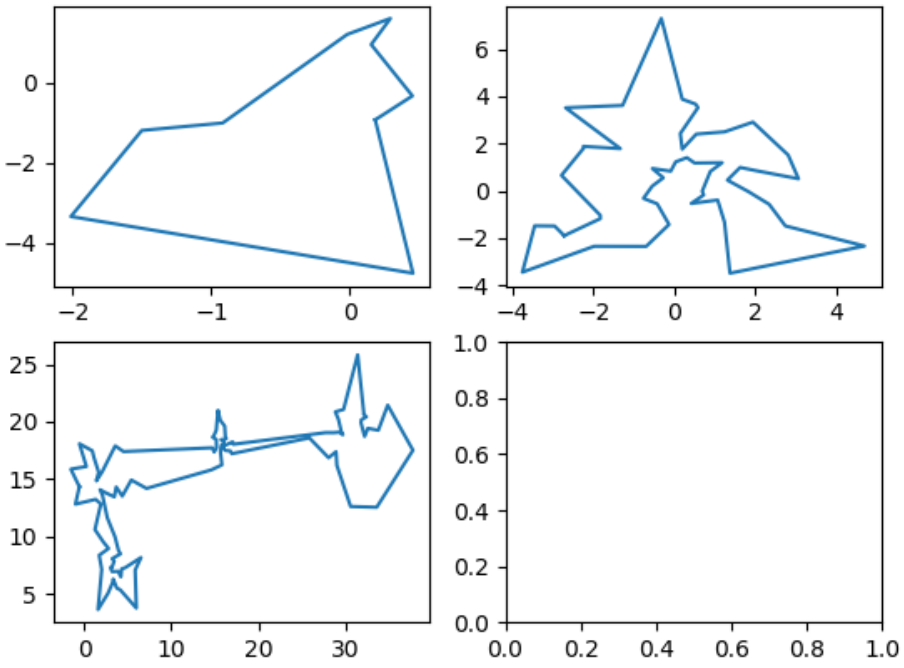

Figure 2: Rozkład normalny z 4 losowymi grupami parametrów; kolejno 20, 60, 100 punktów

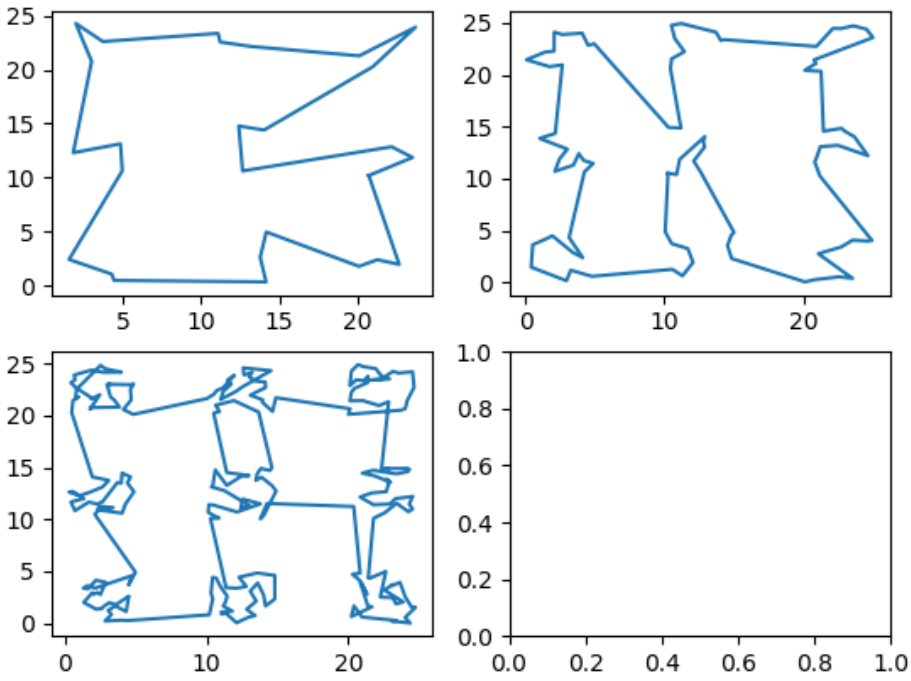

Figure 3: Punkty w 9 odseparowanych grupach; kolejno 27, 90, 270 punktów

1.2 Wpływ funkcji zmiany temperatury na zbieżność energii w układzie

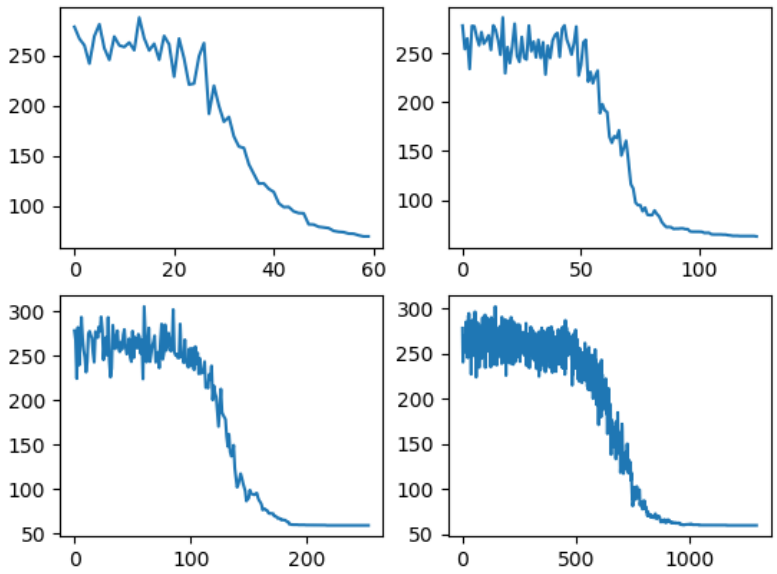

Figure 4: Zależność sumarycznej długości cyklu w zależności od numeru iteracji; kolejne funkcje zmiany temperatury: $T_n = 0.8 * T_{n-1}, T_n = 0.9 * T_{n-1}, T_n = 0.95 * T_{n-1}, T_n = 0.99 * T_{n-1}$

Widzimy, że dla "gęstszych" funkcji temperatury znajdowane długości są na początku mocno rozproszone, jednak z kolejnymi iteracjami dużo łagodniej się stabilizują. Szczególnie widać to w porównaniu pierwszego wykresu z czwartym. Ostatni wykres dużo wyraźniej swoim kształtem przypomina gładką krzywą, a pierwszy aż do samego końca ma wahania wartości.

1.3 Wpływ sposobu znajdowania stanów sąsiednich na zbieżność energii w układzie

Na wykresie widzimy porównanie dwóch taktyk wyboru sąsiada. Widać, że obracanie dłuższych fragmentów daje nam wyraźniejszą zbieżność w trakcie optymalizacji, a także niższe wartości energii układu. Zamiana tylko sąsiadujących elementów daje nam mniejsze zmiany energii, co powoduje że wykres sumarycznej długości bardziej faluje, gdyż łatwiej jest wejść w stan o wyższej energii. W końcowych iteracjach łatwo jest wejść w stan, który jest daleki od

optymalnego, ale zamiana dwóch sąsiednich elementów może tylko zwiększyć energię, a prawdopodobieństwo wyjścia z tego stanu cały czas się zmniejsza.

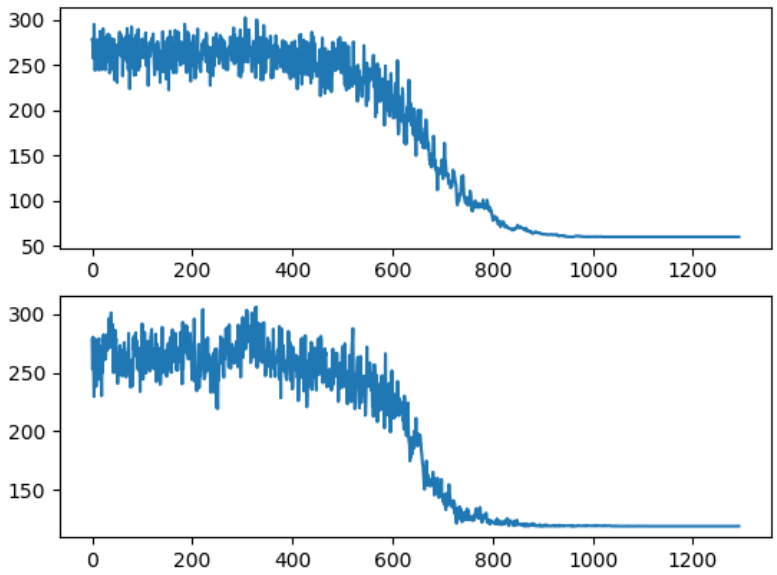

Figure 5: Zależność sumarycznej długości cyklu w zależności od numeru iteracji; u góry obrót fragmentu cyklu, na dole zamiana tylko dwóch elementów będących koło siebie

2 Obraz binarny

Dla losowych obrazów binarnych zastosowany został algorytm symulowanego wyżarzania z różnymi funkcjami energii i spadku temperatury oraz z różnym sposobem doboru stanów sąsiednich. Podstawową funkcją energii E jest funkcja która dla każdego punktu *P1* sprawdza każdy punkt *P2* z jego sąsiedztwa.

$$E(P1) = \sum F(P1, P2)$$

gdzie

$$F(P1, P2) = \begin{cases} 1 & \text{if } color(P1) = color(P2) \\ -1 & \text{if } color(P1) \neq color(P2) \end{cases}$$

Funkcję tę minimalizujemy. Jest ona więc równoważna przyciągającym i odpychającym się magnesom, gdzie kolor to biegun magnesu.

2.1 Uzyskane wyniki dla przykładowych obrazów o różnych gęstościach

Po lewej obraz oryginalny, po prawej wynik działania programu .

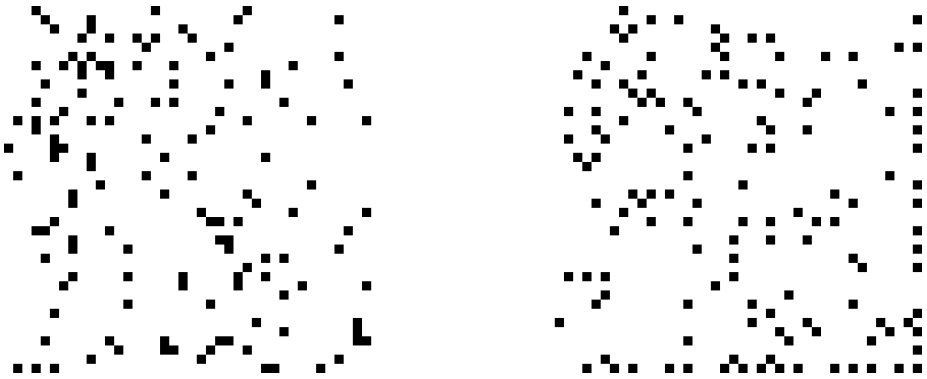

Figure 6: Gęstość czarnych punktów 0.1

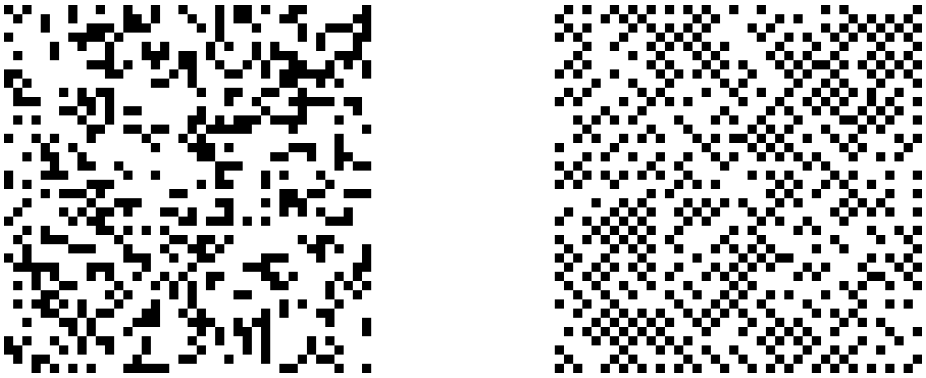

Figure 7: Gęstość czarnych punktów 0.3

Figure 8: Gęstość czarnych punktów 0.4

2.2 Wyniki dla różnych rodzajów sąsiedztwa i funkcji energii

W tym podpunkcie przyjrzymy się wpływowi rodzaju sąsiedztwa na wyniki dla dwóch różnych funkcji energi, takiej jaką podaliśmy na początku (lewa strona) oraz dla funkcji przeciwnej (takie same kolory się przyciągają, różne odpychają; prawa strona).

Figure 9: Początkowy obraz

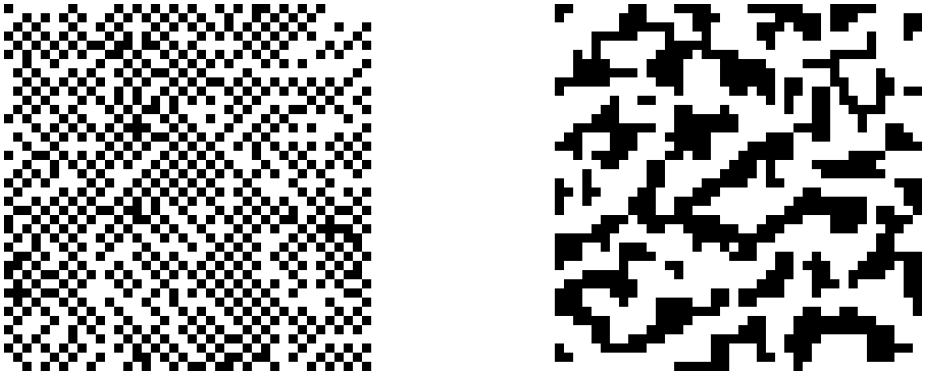

Figure 10: 4-sąsiedztwo

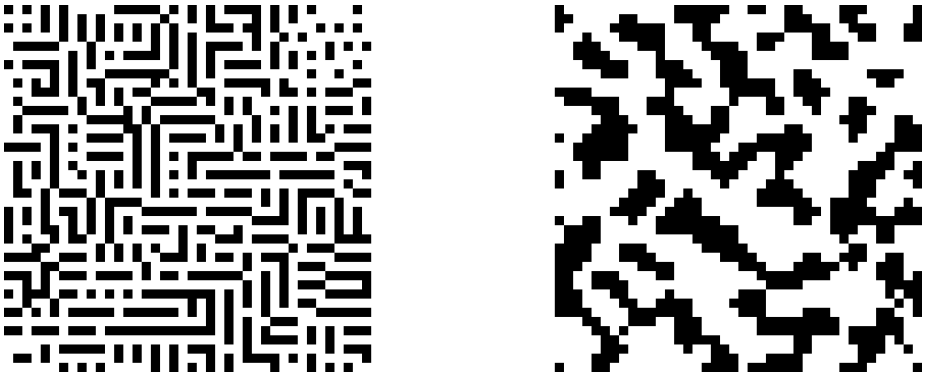

Figure 11: 8-sąsiedztwo

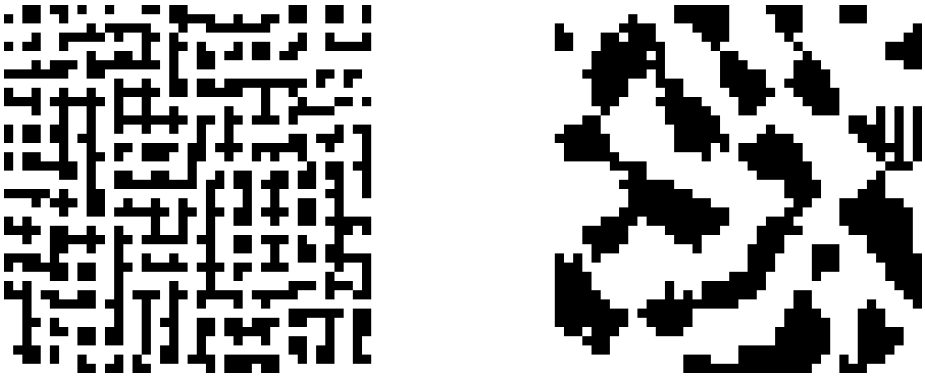

Figure 12: 8-16-sąsiedztwo

Widzimy, że czym większa grupa sąsiedztwa, tym większe grupy punktów otrzymujemy. Dla naszej podstawowej funkcji energii dostajemy grubsze ścieżki, układające się pionowo i poziomo. Ścieżki te nie tworzą się dla 4-sąsiedztwa, ponieważ optymalnym rozwiązaniem tego problemu jest obraz tworzący szachownicę. Dla 8-sąsiedztwa takie rozwiązanie daje dla każdego punktu energię 0, ułożenie obok siebie pionowych ścieżek o różnych kolorach daje zaś dla każdego punktu energię równą -4, czyli jest dużo bardziej optymalnym rozwiązaniem.

Dla przeciwnej funkcji energii obserwujemy tworzenie się coraz większych grup punktów o takim samym kolorze. Było to do przewidzenia, ponieważ czym więcej sąsiadów, tym więcej każdy punkt chce do siebie przyciągnąć.

2.3 Porównanie dla różnych szybkości spadku energii

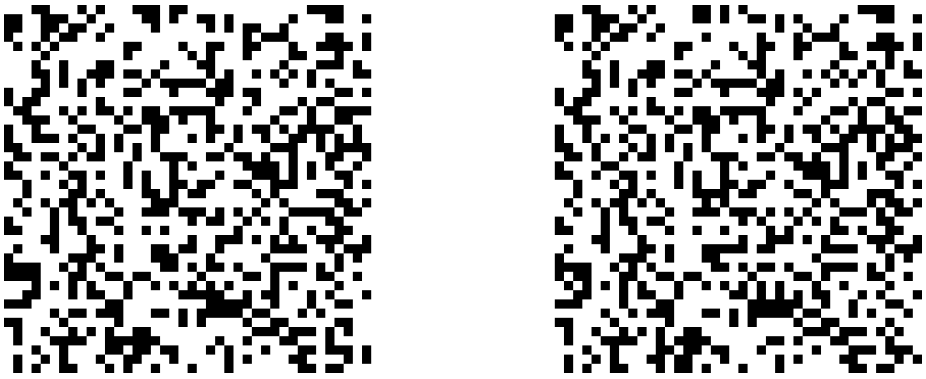

Figure 13: $T_n = 0.9 * T_{n-1}$

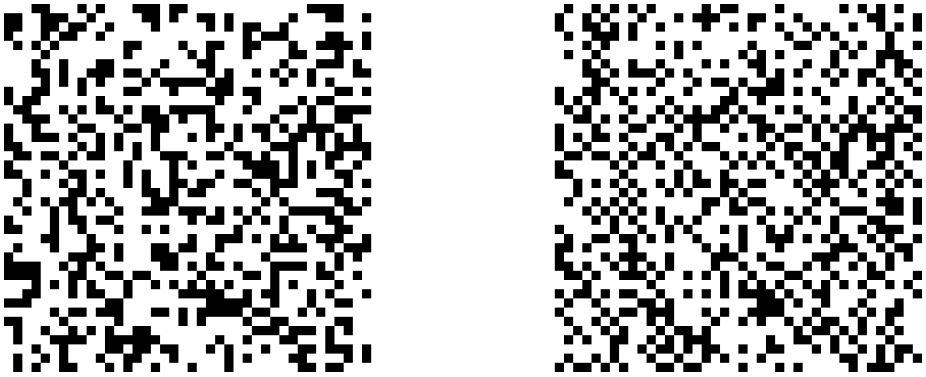

Figure 14: $T_n = 0.99 * T_{n-1}$

Możemy tu zaobserwować efekty zbyt szybkiego ochładzania układu, na pierwszym obrazie ciężko zauważyć jakiekolwiek skutki optymalizacji.

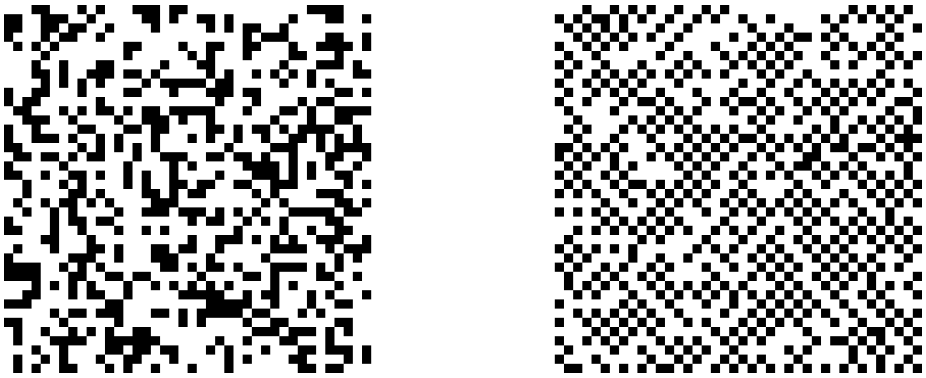

Figure 15: $T_n = 0.999 * T_{n-1}$

2.4 Porównanie dla różnych sposobów generacji stanów sąsiednich

Do wygenerowania obrazu po lewej użyto zamiany dwóch losowych, bezpośrednich sąsiadów (z 4-sąsiedztwa), po prawej użyto zamiany dwóch losowych punktów z całego obrazu.

Figure 16: Oryginalny obraz

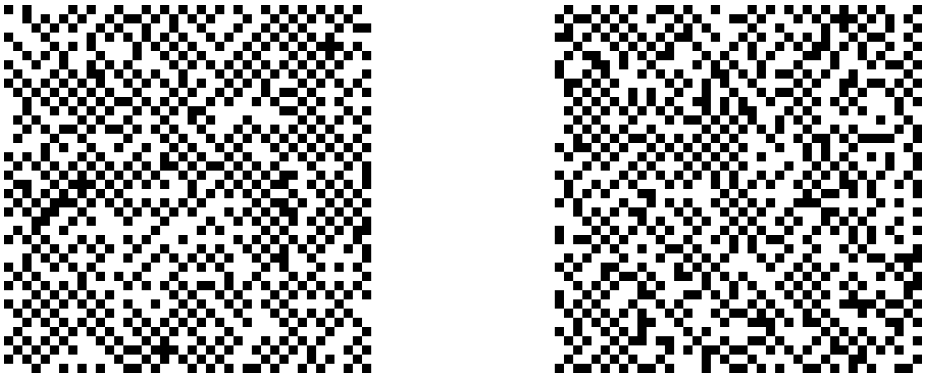

Figure 17: 4-sąsiedztwo

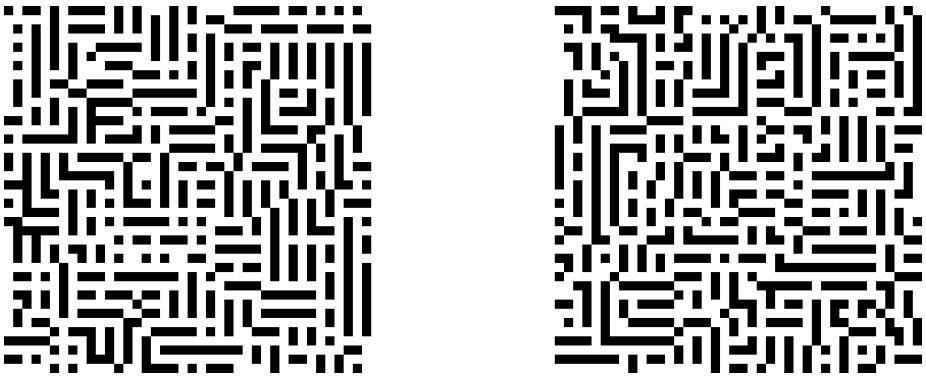

Figure 18: 4-sąsiedztwo

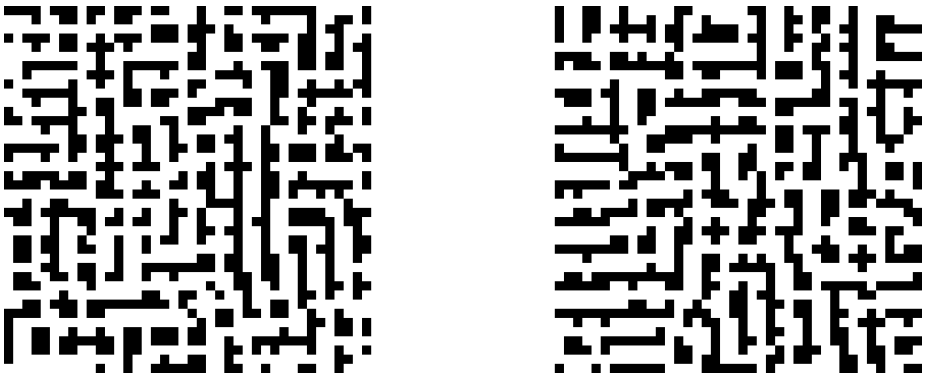

Figure 19: 8-16-sąsiedztwo

Wygenerowane obrazy nie różnią się znacznie od siebie, można zauważyć że drugi obraz w pierwszym przykładzie ma większą tendencję do gromadzenia grup punktów. Może to wynikać z faktu, iż dużo rzadziej próbuje on poprawić wartość funkcji energii lokalnie, gdzie przy generowaniu pierwszego obrazu z każdą iteracją poprawiamy lokalne rezultaty.

3 Sudoku

Specyfikacja algorytmu symulowanego wyżarzania zaimplementowanego w celu rozwiązywania sudoku:

- Początkowe rozwiązanie → każdy kwadrat losowo wypełniony tak, aby w kwadratach wystąpiły wszystkie cyfry od 1 do 9
- Eneriga układu → ilość powtarzających się elementów w każdej kolumnie, wierszu
- Stan sąsiedni → plansza po n zamianach pojedynczych elementów, gdzie n jest liczbą losową i $n \in \{1, 2, 3\}$

3.1 Zależność ilości iteracji potrzebnych do rozwiązania od ilości pustych pól na planszy

Do przedstawienia tej zależności użyta została pełna plansza z której usunięto losowe elementy, dla każdej ilości pustych miejsc algorytm został uruchomiony 3 razy, a z ilości iteracji[1] wyciągnięta została średnia arytmetyczna.

[1] Do średniej wliczane były tylko ilości iteracji wykonań, które dały poprawne rozwiązanie

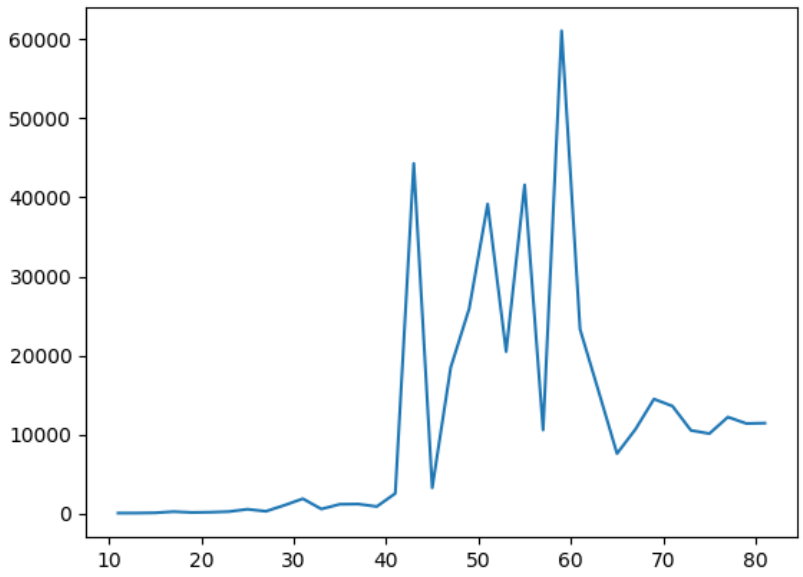

Figure 20: Wykres ilości iteracji potrzebnych do otrzymania prawidłowego rozwiązania od ilości pustych pól

Na wykresie widać, że największe problemy z rozwiązaniem program miał w okolicach 53 pustych pól. Dwa razy (dla 43 i 59 pustych pól) zdarzyło się, że algorytm nie znalazł rozwiązania. Na wykresie widzimy, że miał oba przypadki były dla niego najtrudniejsze i wymagały największej ilości iteracji.

Dla małej ilości pustych pól oczywiste jest, że algorytm szybko znajdzie rozwiązanie, jednak dla wartości bliskich 81 (prawie pusta plansza) rozwiązania też znajdowane są relatywnie szybko. Wynika to z faktu, iż algorytm ma dużo większą dowolność zamiana pozycji komórek, przez co jest mniejsza szansa na utknięcie w stanie który jest daleki od rozwiązania.

3.2 Problemy algorytmu wyżarzania w problemie sudoku

Dla niektórych plansz rozwiązanie sudoku było dla programu prawie niemożliwe, bardzo rzadko zdarzało się, że zostały rozwiązane. Problemem tutaj jest charakterystyka algorytmu wyżarzania. Przy rozwiązywaniu sudoku jedyne co nas interesuje to otrzymanie planszy, której energia jest równa 0. Przy rozwiązywaniu TSP zgadzaliśmy się na to, że trasa nie będzie najprawdopodobniej najkrótszą możliwą. Po zwizualizowaniu otrzymanych wyników gołym okiem widać, że trasa została zadowalająco dobrze zoptymalizowana.

W zaprojektowanym na potrzeby sudoku algorytmie, maksymalna energia w układzie wynosi 162 (każdy punkt może dodawać 1 energii w pionie i 1 energii w poziomie, a z powodu doboru stanu początkowego w kwadratach energia jest równa 0 i się nie zmienia). Optymalizacja dająca nam planszę o energii 4 jest więc bardzo dobrym wynikiem w porównaniu do jej maksymalnej wartości, jednak daje nam to nieprawidłowe sudoku, które może być bardzo dalekie od poprawnego rozwiązania.