Zadanie 5. Hybrydowy Algorytm Ewolucyjny

Oskar Kiliańczyk 151863 & Wojciech Kot 151879

1 Opis zadania

Celem zadania jest porównanie hybrydowego algorytmu z i porównanie go z metodami MSLS, ILS i LNS z poprzedniego zadania. Opisy tych metod znajdują się w poprzednim sprawozdaniu. Dla porównania, w zestawieniach znajduje się też metoda heurystyczna z pierwszego sprawozdania, a mianowicie zachłanna rozbudowa cyklu, której to heurystyka jest wykorzystywana w rekonstrukcji zarówno w LNS jak i HAE.

2 Opisy algorytmów

2.1 Hybrydowy Algorytm Ewolucyjny

- steady state
- z populacją elitarną 20 osobników
- Usuwanie duplikatów rozwiązań (Dwa rozwiązania są uznawane za duplikat, jeśli mają identyczną wartość funkcji celu)
- Wykorzystywane parametry to flagi use_mutation, use_local_search oraz mutation_rate = 0.1

2.1.1 Główny algorytm

1. Wygeneruj populację początkową.
2. Dopóki czas nie został przekroczony, powtarzaj:
   (a) Wybierz parę rodziców (rodzic1, rodzic2) losowo z populacji
   (b) Utwórz dwa rozwiązania potomne* (Procedura rekombinacji)
   (c) Jeśli jest ustawiona flaga use_mutation - wykonaj mutację** (Procedura mutacji)
   (d) Jeśli jest ustawiona flaga use_local_search - wykonaj przeszukiwanie lokalne
   (e) Oblicz koszt rozwiązań potomnych
   (f) Dla każdego z rozwiązań potomnych:
      i. Jeśli koszt rozwiązania potomnego jest lepszy niż obecnie najgorszy koszt w populacji, oraz jeśli nie ma identycznego w populacji:
      ii. Zamień najgorsze rozwiązanie z populacji z obecnie sprawdzanym rozwiązaniem potomnym.
3. Zapisz i zwróć najlepsze rozwiązanie z obecnej populacji

2.1.2 Procedura rekombinacji

1. Parametry: dwa rozwiązania wejściowe
2. Wybierz pierwsze rozwiązanie jako bazowe, drugie jako referencyjne
3. Stwórz zbiór krawędzi z rozwiązania referencyjnego (r_edges)
4. Dla każdego cyklu w rozwiązaniu bazowym:
   (a) Usuń wierzchołki, które tworzą krawędzie nieistniejące w zbiorze r_edges
5. Utwórz zbiór wierzchołków nieużytych w obecnym rozwiązaniu
6. Dopóki istnieją wierzchołki w tym zbiorze:
   (a) Wybierz losowy wierzchołek z tego zbioru
   (b) Wstaw go do cyklu w miejsce minimalizujące koszt
7. Zwróć nowe rozwiązanie jako rozwiązanie potomne

2.1.3 Procedura mutacji

1. Dla każdego cyklu:
   (a) Z prawdopodobieństwem mutacji równym mutation_rate:
      i. Wybierz losowo typ mutacji (zamiana krawędzi lub zamiana wierzchołków)
      ii. Jeśli wybrano zamianę krawędzi: zamień dwie losowo wybrane krawędzie w cyklu
      iii. W przeciwnym wypadku zamień dwa losowo wybrane wierzchołki w cyklu

3 Wyniki

3.1 Tabela wynikowa

Algorytm	Best	Avg	Worst	Best Time	Avg Time	Worst Time	Avg Iterations
MSLS	34630	35375.7	35862	235.639	269.287	327.604	-
ILS	31016	31919.8	33193	270.753	270.851	270.946	4977.8
LNS	29690	30559.4	32006	272.054	272.242	272.595	5714.6
LNS bez LS	30980	31801.6	33356	272.005	272.142	272.295	35672.0
Cycle Extension Greedy	35396	38319.41	39990	-	-	-	-
HAE	31418	33897.3	36636	269.287	269.287	269.288	692612
HAE +M	30765	32818.3	34391	269.287	269.287	269.288	773693
HAE +LS	30310	31058.1	31661	269.287	269.295	269.302	14200.9
HAE +M +LS	29975	30872.7	31499	269.287	269.296	269.306	12946.9

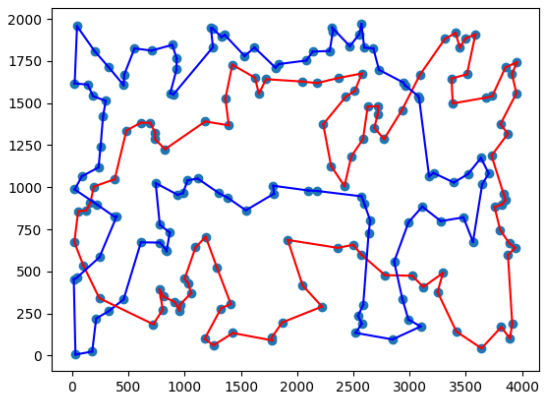
Tabela 1: Wyniki dla kroA200

Algorytm	Best	Avg	Worst	Best Time	Avg Time	Worst Time	Avg Iterations
MSLS	34819	35501.6	36081	256.952	279.947	380.768	-
ILS	31475	32454.2	33343	281.034	281.280	281.655	6057.1
LNS	30092	30770.9	31762	282.524	282.865	283.275	5768.0
LNS bez LS	30980	31801.6	33356	272.005	272.142	272.295	35672.0
Cycle Extension Greedy	35295	38455.17	40028	-	-	-	-
HAE	32389	34070.3	34951	279.947	279.948	279.948	699911
HAE +M	31383	33151.2	36564	279.947	279.947	279.948	620006
HAE +LS	31169	31873.7	32595	279.947	279.957	279.966	14551.9
HAE +M +LS	30604	31231	31837	279.949	279.959	279.975	12954.5

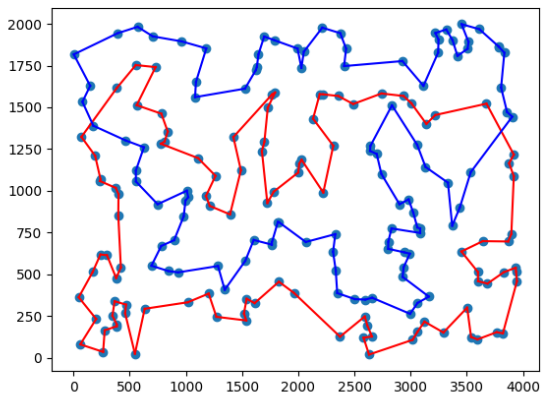
Tabela 2: Wyniki dla kroB200

3.2 Wizualizacja wyników

3.2.1 ILS

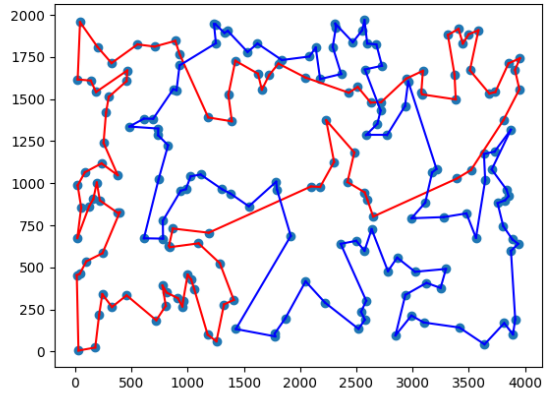


Rysunek 1: kroA200, losowy start

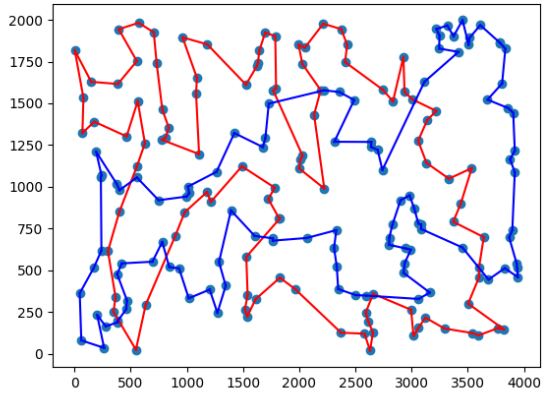


Rysunek 2: kroB200, losowy start

3.2.2 MSLS

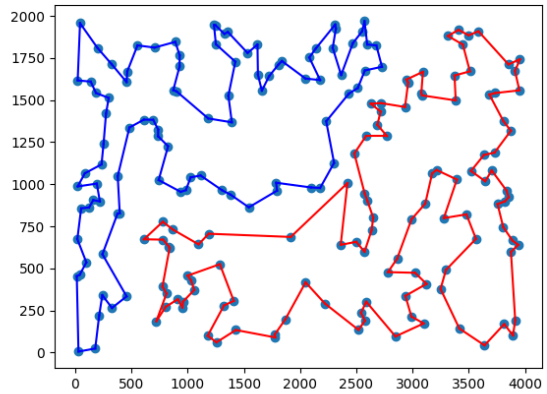


Rysunek 3: kroA200, losowy start

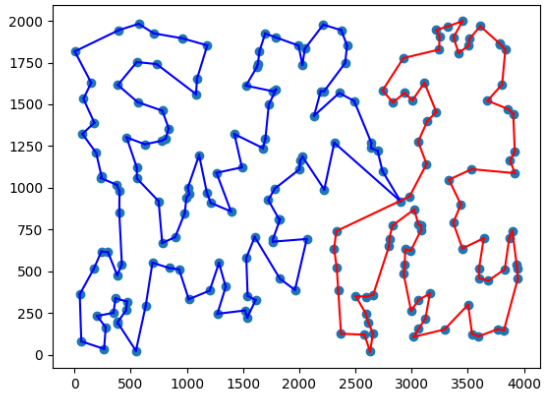


Rysunek 4: kroB200, losowy start

3.2.3 LNS

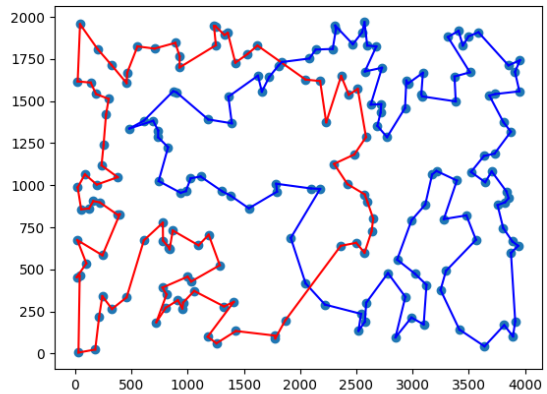


Rysunek 5: kroA200, losowy start

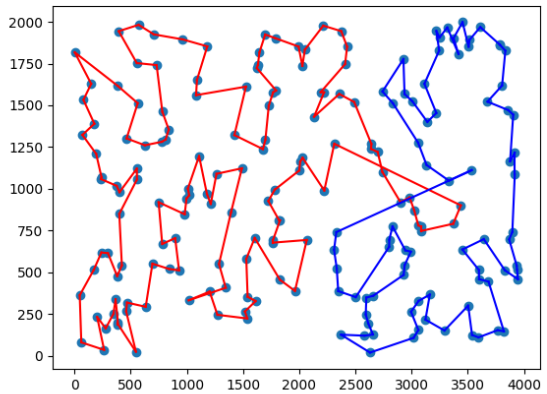


Rysunek 6: kroB200, losowy start

3.2.4 LNS bez LS

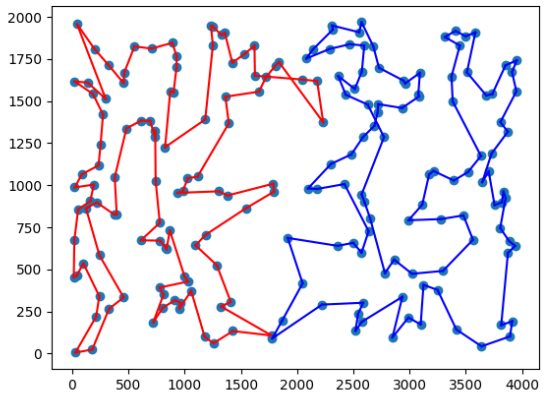


Rysunek 7: kroA200, losowy start

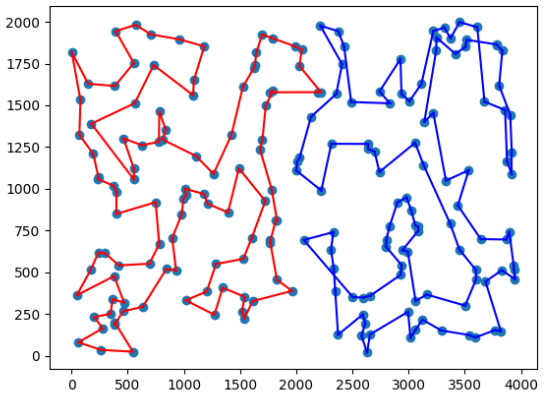


Rysunek 8: kroB200, losowy start

3.2.5 Zachłanna rozbudowa cyklu

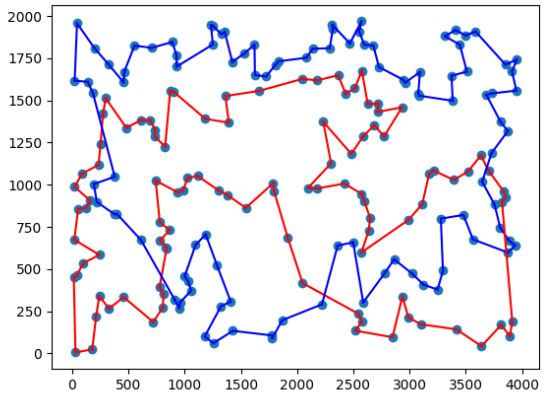


Rysunek 9: kroA200, losowy start

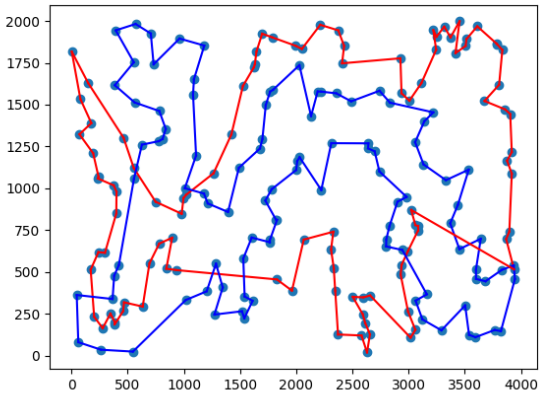


Rysunek 10: kroB200, losowy start

3.2.6 HAE - bazowy

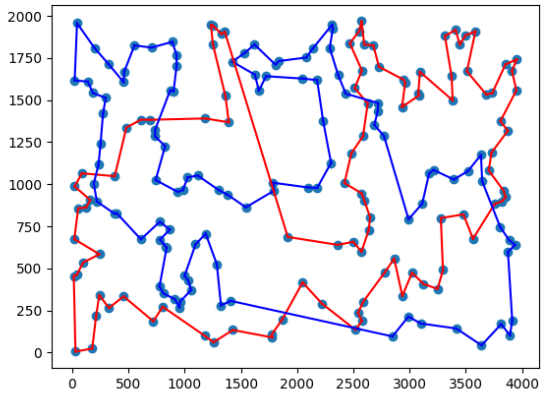


Rysunek 11: kroA200, losowy start

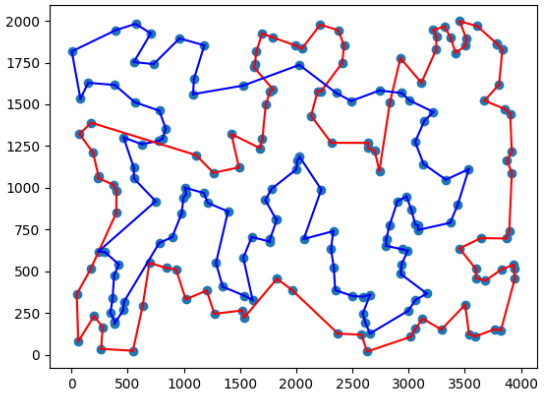


Rysunek 12: kroB200, losowy start

3.2.7 HAE z mutacją

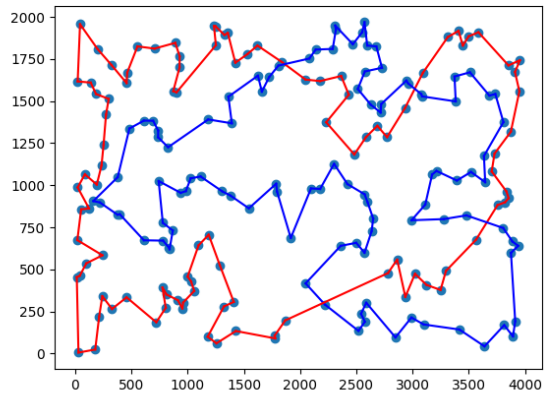


Rysunek 13: kroA200, losowy start

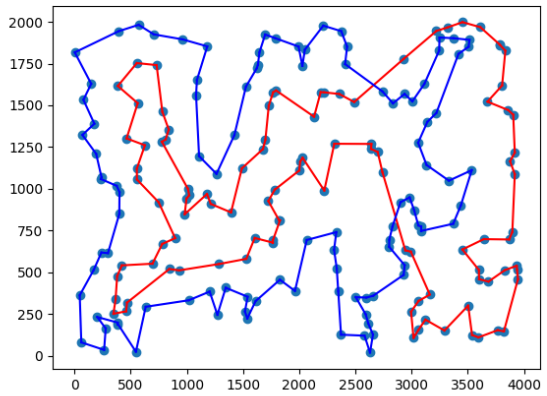


Rysunek 14: kroB200, losowy start

3.2.8 HAE z lokalnym przeszukiwaniem

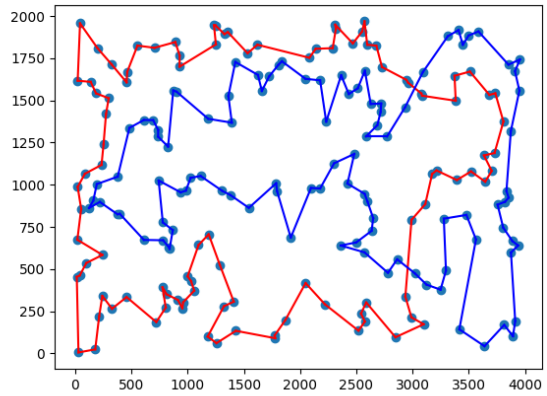


Rysunek 15: kroA200, losowy start

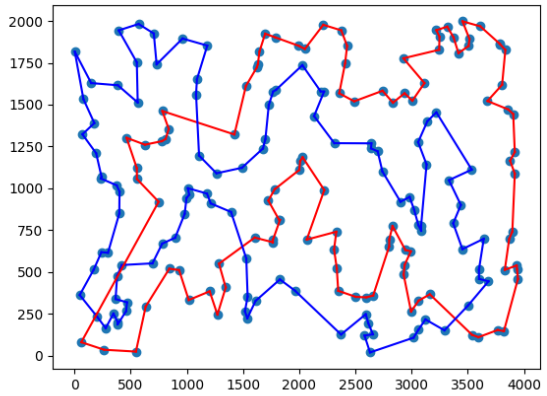


Rysunek 16: kroB200, losowy start

3.2.9 HAE z mutacją i lokalnym przeszukiwaniem



Rysunek 17: kroA200, losowy start



Rysunek 18: kroB200, losowy start

4 Wnioski i analiza wyników

Na podstawie wyników eksperymentu można zauważyć że HAE (Hybrydowy Algorytm Ewolucyjny) osiągnął wyniki porównywalne z najlepszą z metod badanych w poprzednim sprawozdaniu, czyli z LNS (Large Neighbourhood Search). Kluczowe w uzyskaniu lepszego wyniku jest jednak zastosowanie zarówno mutacji jak i lokalnego przeszukiwania, bez tych dodatków HAE w porównaniu wypada gorzej od LNS oraz ILS (Iterated Local Search).

5 Link do repozytorium

Kod źródłowy w repozytorium GitHub dostępny pod linkiem:
Repozytorium.