Zadanie 6. Własna metoda i testy globalnej wypukłości

Oskar Kiliańczyk 151863 & Wojciech Kot 151879

1 Opis zadania

Zadanie polega na wygenerowaniu losowo 1000 optimów lokalnych, za pomocą zachłannej wersji przeszukiwania lokalnego, oraz bardzo dobrego rozwiązania za pomocą najlepszej do tej pory metody. Tą metodą powinna być własna metoda, której opracowanie również jest częścią tego zadania. Następnie dla każdego z lokalnych optimów należy obliczyć podobieństwo do rozwiązania referencyjnego oraz średnie podobieństwo do poozstałych optimów lokalnych ze zbioru, oraz współczynnik korelacji. Stosowane miary podobieństwa to:

1. Liczba par wierzchołków przydzielonych w obu rozwiązaniach razem do jednego cyklu
2. Liczba wspólnych par krawędzi

2 Własna Metoda

2.1 Pierwsza heurystyka konstrukcyjna

Już w ramach pierwszego sprawozdania zaimplementowaliśmy własną metodę, dlatego wykorzystamy ją tutaj w porównaniach. Dla przypomnienia oto jej pseudokod:

1. Utworzenie listy pozostałych wierzchołków (wszystkie możliwe, poza startowymi)
2. Utworzenie dwóch list zawierających odpowiednio dystanse każdego wierzchołka do pierwszego i do drugiego wierzchołka startowego
3. Sortowanie utworzonych uprzednio list dystansów wierzchołków
4. Aż do przydzielenia wszystkich wierzchołków z listy pozostałych wierzchołków wykonuje:
5. Zmianę decyzji, do którego zestawu wierzchołków obecnie będzie przydzielać wierzchołek (aby robić to naprzemiennie)
6. Wyszukuje pierwszy wierzchołek na liście dystansów, który nie został jeszcze przydzielony do żadnego zestawu i przydziela go tam

W skutek zastosowania takiego przydziału uzyskujemy dwa równo-liczne zbiory, oraz zapewniamy, że gdyby ilość badanych wierzchołków była nieparzysta, to zbiory będą różnić się długością najwyżej o 1.

Następnie wykorzystujemy tradycyjny algorytm rozbudowy cyklu na podstawie dwużalu, osobno na obu listach. Wygląda on następująco:

1. Algorytm zaczyna od ścieżki zawierającej wierzchołek startowy podwójnie
2. Dopóki w ścieżce nie znajdują sie wszystkie wierzchołki z zadanego mu zestawu powtarza:
3. Dla każdego nieodwiedzonego wierzchołka oblicza koszty jego wstawienia
4. Następnie oblicza żal (dwużal) dla danego wierzchołka
5. Rozbudowuje cykl o wierzchołek z największym obliczonym żalem i zaznacza go jako odwiedzonego
6. Zwraca ścieżkę

2.2 Ulepszenie naszej heurystyki konstrukcyjnej

TODO Ulepszenie poprzez lepszy podział Trzeba wypróbować kilka, Kmeans, DBSCAN, etc.

2.3 Finalna własna metoda

TODO Finalnie zdecydowaliśmy się zastosować Large Neighbourhood Search

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
HAE +M +LS	29975	30872.7	31499	269.287	269.296	269.306	12946.9
nasz - konstukcyjna	30293.00	33044.22	36854.00	-	-	-	-
nasz - cały	x	x	x	x	x	x	x

Tabela 1: Wyniki dla `kroA200`

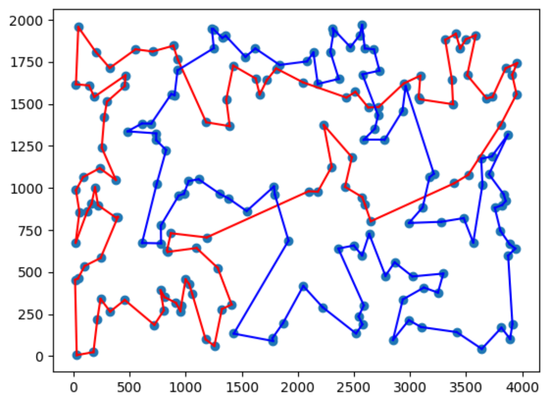
Algorytm	Best	Avg	Worst	Best Time	Avg Time	Worst Time	Avg Iterations
MSLS	34819	35501.6	36081	256.952	279.947	380.768	-
ILS	31475	32454.2	33343	281.034	281.280	281.655	6057.1
LNS	30092	30770.9	31762	282.524	282.865	283.275	5768.0
HAE +M +LS	30604	31231	31837	279.949	279.959	279.975	12954.5
nasz - konstukcyjna	31218.00	33317.82	36913.00	-	-	-	-
nasz - cały	x	x	x	x	x	x	x

Tabela 2: Wyniki dla `kroB200`

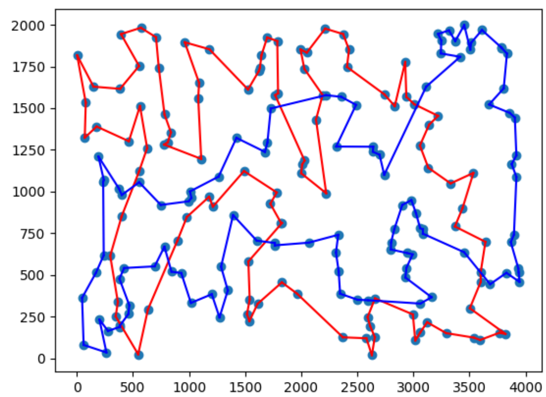
2.4 Wyniki obliczeniowe dla algorytmów

2.5 Wizualizacje najlepszych wyników

2.5.1 MSLS

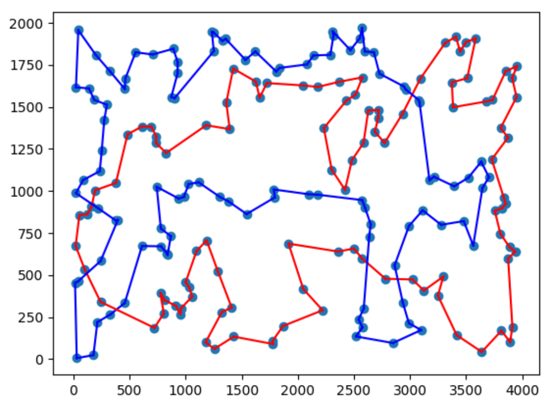


Rysunek 1: kroA200, losowy start

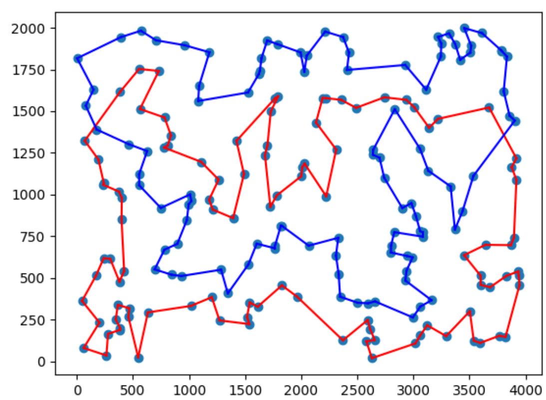


Rysunek 2: kroB200, losowy start

2.5.2 ILS

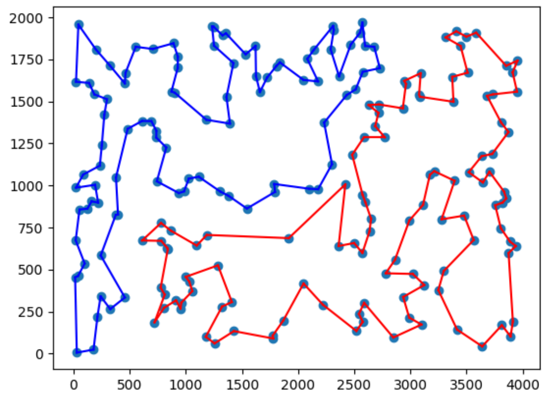


Rysunek 3: kroA200, losowy start

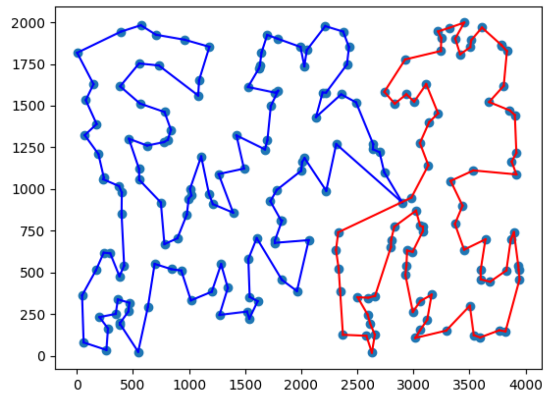


Rysunek 4: kroB200, losowy start

2.5.3 LNS

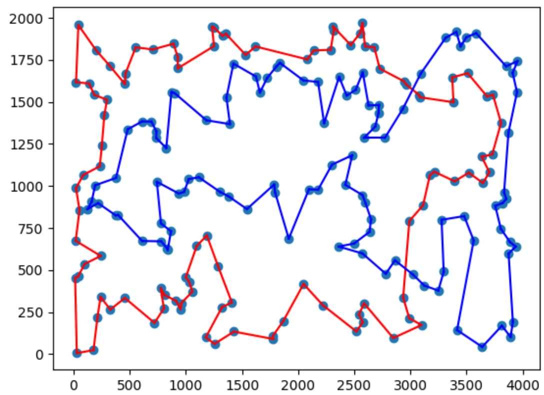


Rysunek 5: kroA200, losowy start

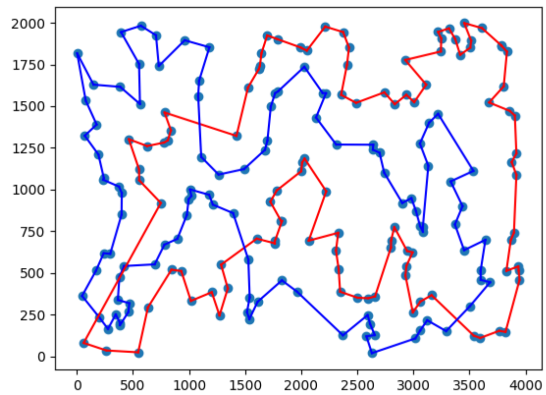


Rysunek 6: kroB200, losowy start

2.5.4 HAE z mutacją i lokalnym przeszukiwaniem



Rysunek 7: kroA200, losowy start



Rysunek 8: kroB200, losowy start

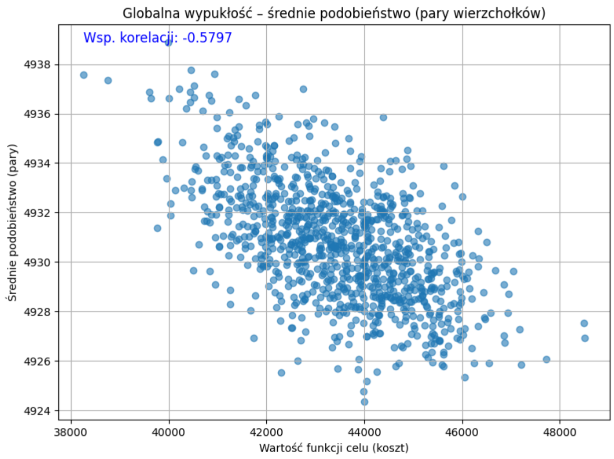
2.6 Wnioski

Zaprojektowana przez nas heurystyka konstrukcyjna wykazuje się niebywałą szybkością i jest świetną opcją jeśli zależy nam na czasie. W przypadku, gdy dążymy do jak najdokładniejszych rozwiązań nie zważając tak na czas, to połączenie jej z TODO algorytmem TODO robi TODO

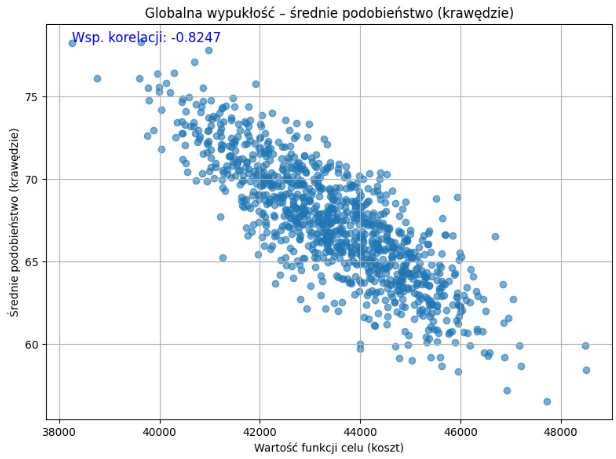
3 Testy globalnej wypukłości

Wygenerowaliśmy 1000 lokalnych optimów dla każdej z analizowanych instancji problemu, za pomocą lokalnego przeszukiwania w wersji zachłannej z losowych rozwiązań początkowych. Obliczyliśmy podobieństwa każdego lokalnego optimum do rozwiązania referencyjnego (bardzo dobrego) oraz średnie podobieństwo do pozostałych lokalnych optimów. Korzystamy z dwóch miar podobieństwa: Liczby par wierzchołków przydzielonych do tego samego cyklu w obu rozwiązaniach oraz z liczby wspólnych krawędzi w rozwiązaniach.

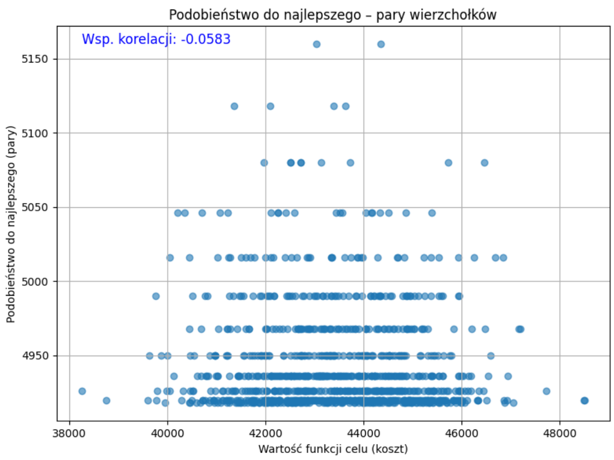
3.1 Wyniki dla instancji kroA200



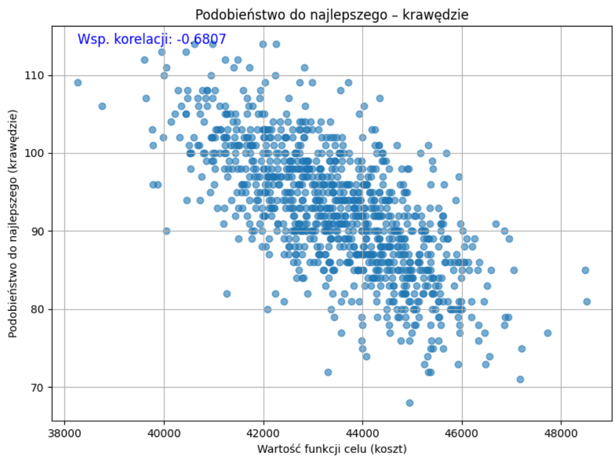
Rysunek 9: podobieństwo parami - kroA



Rysunek 10: podobieństwo krawedziami - kroA

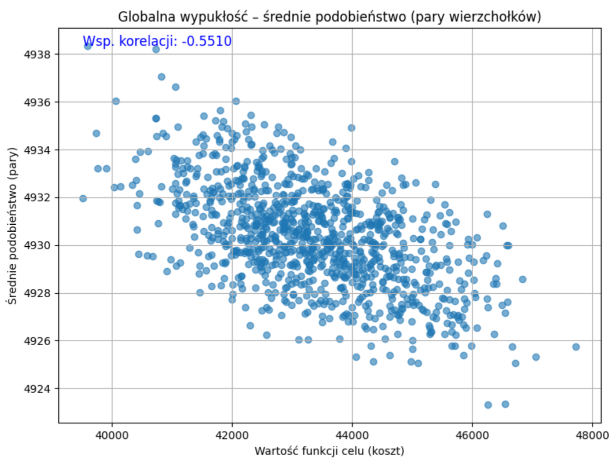


Rysunek 11: podobieństwo parami do najlepszego- kroA

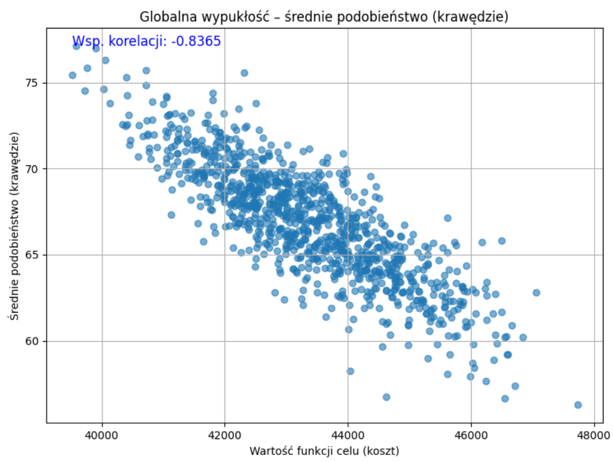


Rysunek 12: podobieństwo krawedziami do najlepszego- kroA

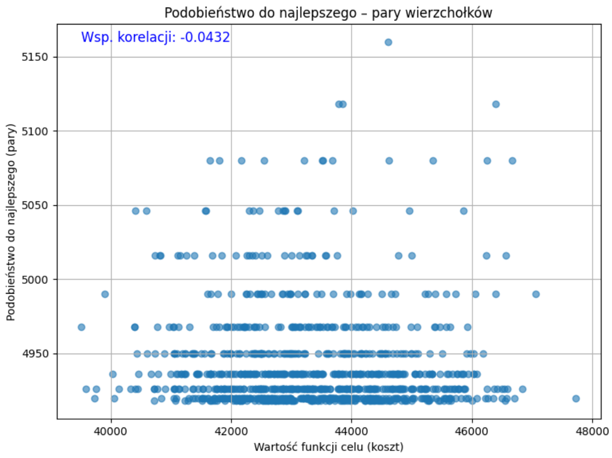
3.2 Wyniki dla instancji kroB200



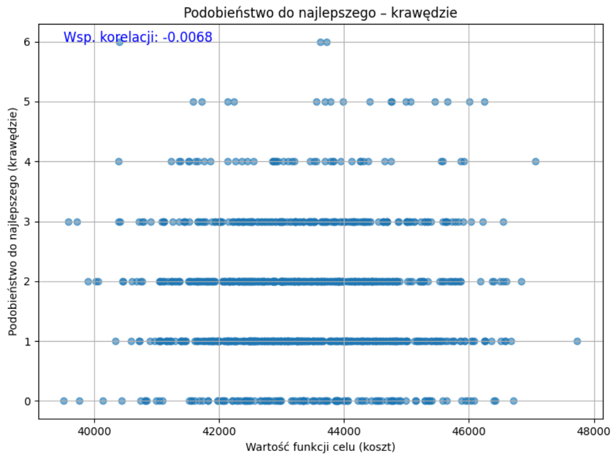
Rysunek 13: podobieństwo parami - kroB



Rysunek 14: podobieństwo krawedziami - kroB



Rysunek 15: podobieństwo parami do najlepszego- kroB



Rysunek 16: podobieństwo krawedziami do najlepszego- kroB

3.3 Wnioski

Na podstawie powyższych eksperymentów można stwierdzić że funkcja wykazuje cechy globalnej wypukłości. Przede wszystkim, przemawia za tym silna ujemna korelacja między wartością funkcji celu, a średnim podobieństwem do innych lokalnych minimów. Miara podobieństwa oparta na krawędziach zdaje się lepiej oddawać strukturę przestrzeni rozwiązań, niż ta oparta na parach wierzchołków (patrząc stricte na siłę korelacji). W ogólności, eksperyment pokazał że dobre rozwiązania są do siebie podobne, co jest zgodne z definicją globalnej wypukłości - choć nie implikuje że wszystkie podobne rozwiązania będą dobre.

4 Link do repozytorium

Kod źródłowy w repozytorium GitHub dostępny pod linkiem:
Repozytorium.