Sprawozdanie lista1

Autorzy: Tomasz Hołub & Piotr Piotrowski

Wstęp:

Sprawozdanie zawiera opis i analizę badań prowadzonych nad algorytmami heurystycznymi, rozwiązującymi problem komiwojażera. Rozpatrywanymi czynnikami są:

- Wpływ wielkości instancji na czas wykonywania algorytów
- Wpływ wielkości instancji na dokładność uzyskanego wyniku.

Algorytmy zostały napisane w języku Python, pełny kod znajduję się w repozytorium :

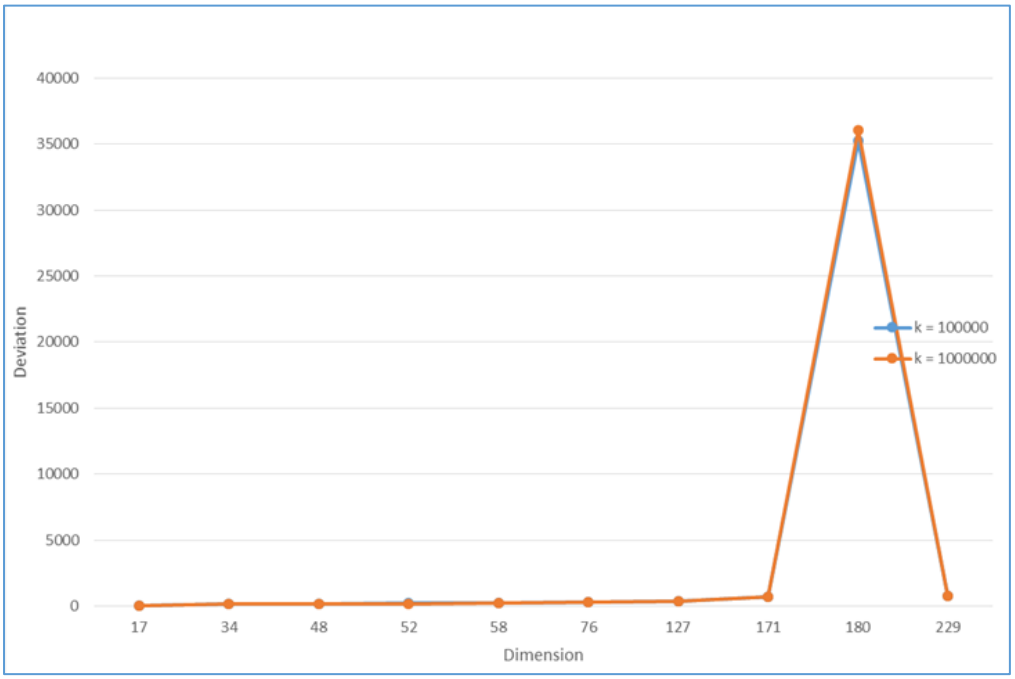
https://github.com/holubek01/Algorytmy_metaheurystyczne

Zaimplementowane algorytmy:

- K-random
- Closest neighbour
- Extended closest neighbour
- 2 OPT

1. Analiza algorytmu „K-random”:

Algorytm polega na wygenerowaniu dopuszczalnego losowego rozwiązania wiele razy i wybraniu najlepszego z nich wszystkich.



Wykres 1.1 opisujący procentową wartość odchylenia w zależności od wielkości instancji dla algorytmu K-random

Wartość odchylenia liczona jest według poniższego wzoru:

$$\frac{(x-y)}{y} * 100\ \%$$

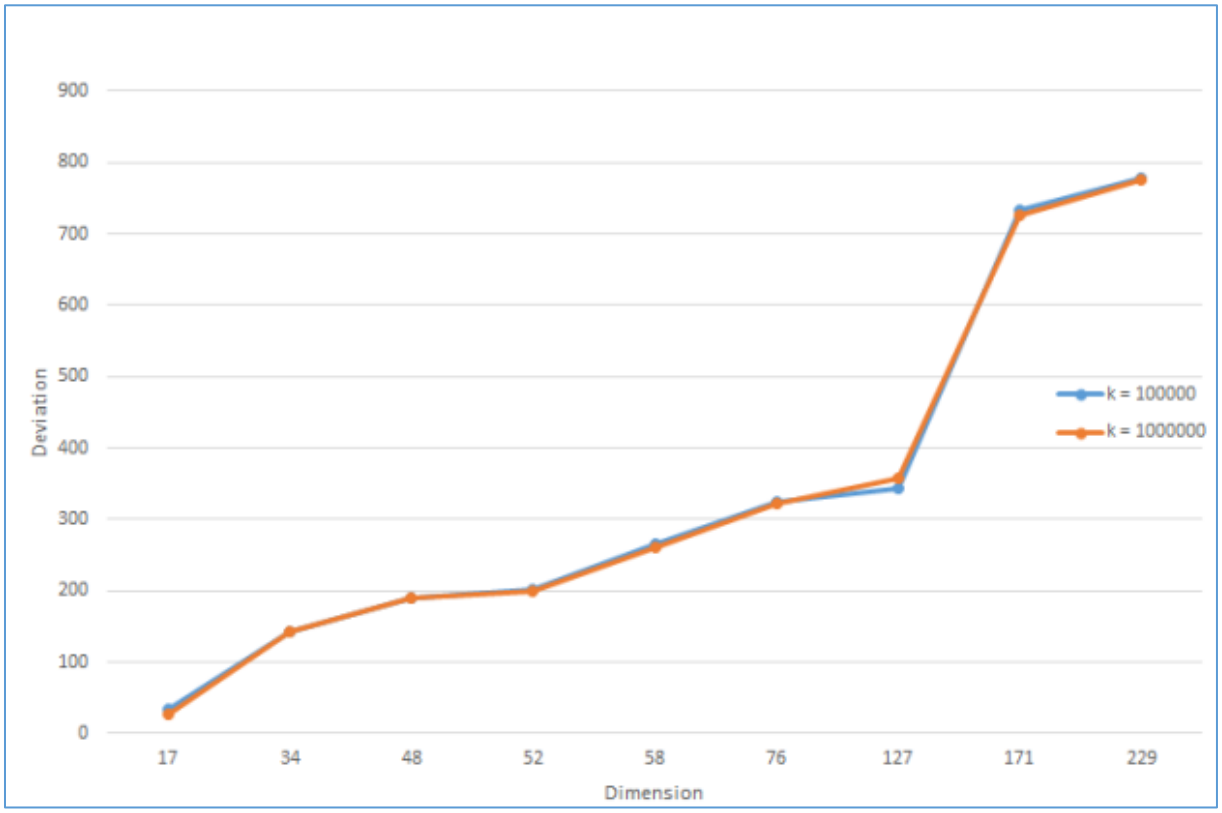
Gdzie:

$$x - wartość\ funkcji\ celu\ dla\ danej\ instancji\ (algorytm\ K\ random)$$

$$y - wartość\ funkcji\ celu\ dla\ najlepszego\ znanego\ rozwiązania$$

Analiza:

Dla instancji o wielkości 180 procentowa wartość odchylenia gwałtownie wzrasta. Po przeanalizowaniu tej instancji zauważyliśmy, że odległości między miastami nie spełniały nierówności trójkąta – miały nierealne odległości, zatem zamiana ze sobą dwóch miast mogła w tym przypadku sprawić, że długość ścieżki wzrosła nawet o 18.000 jednostek, gdzie optymalna długość ścieżki wynosi 1950 jednostek. Na niekorzyść wpływa też fakt, że dla dla 179! możliwych ścieżek losujemy jedynie 1.000.000 lub 100.000.



Wykres 1.2 opisujący procentową wartość odchylenia w zależności od wielkości instancji (bez 180) dla algorytmu K-random

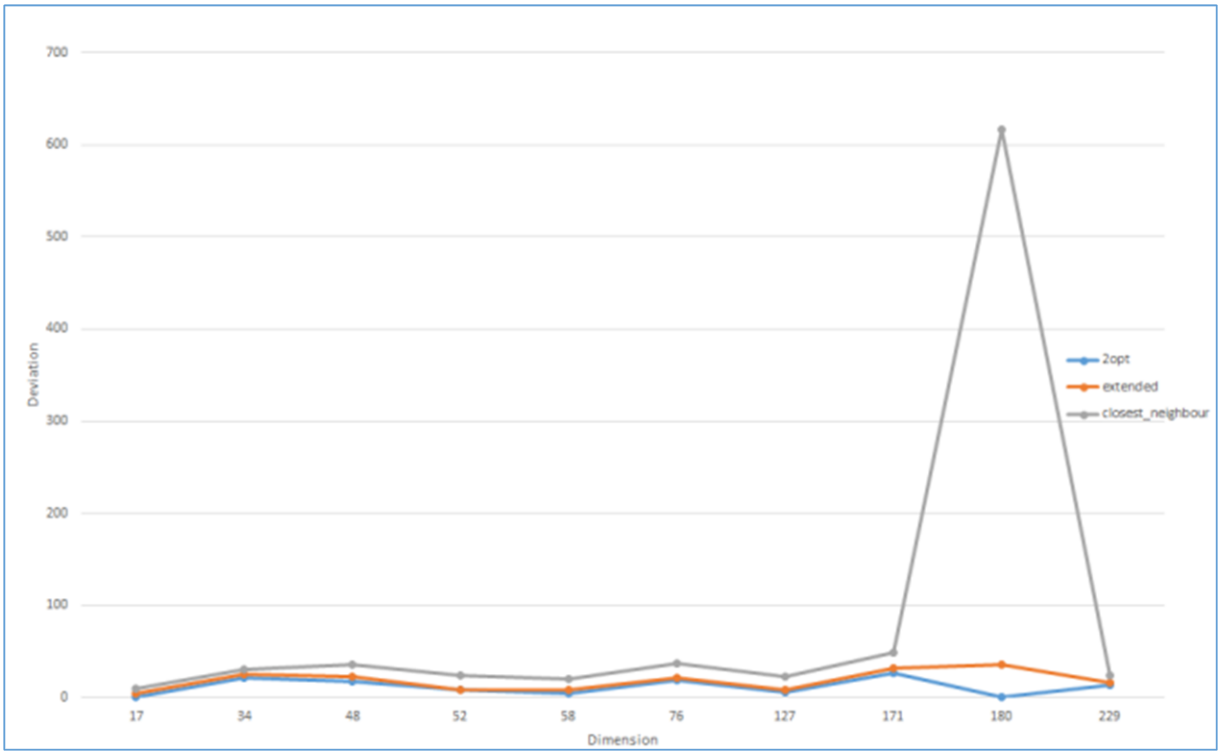
Analiza:

Wykres bez rozpatrywania instancji o wielkości 180 nie różni się już tak drastycznie. Dodatkowo można zauważyć, że liczba losowań rozwiązania dopuszczalnego (tutaj dla 100.000 i 1.000.000) nie

ma większego wpływu na wartość odchylenia. Czasami dla k = 100.000 otrzymaliśmy długość ścieżki mniejszą niż dla k = 1.000.000. Jest to spowodowane faktem, że jest to algorytm całkowicie losowy.

2. Analiza pozostałych algorytmów

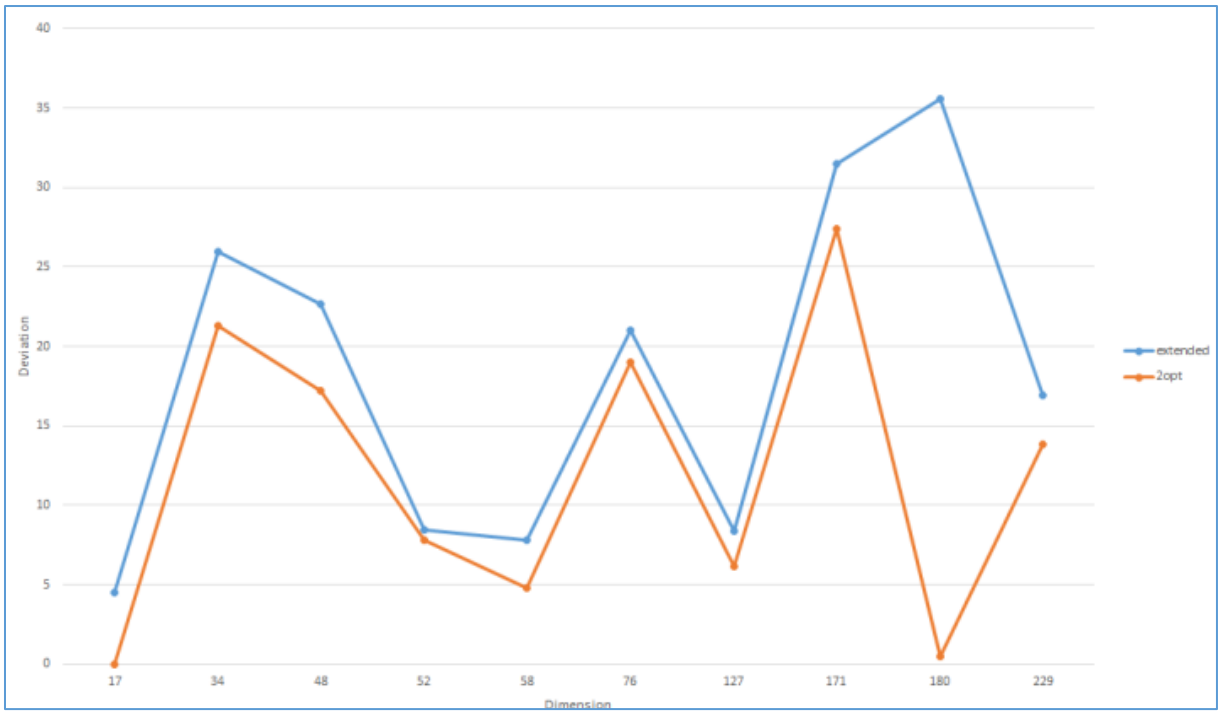
W tym punkcie analizować będziemy dokładność wyniku w zależności od wielkości instancji dla algorytmów „closest neighbour”, „extended closest neighbour” oraz „2 OPT”.



Wykres 2.1 opisujący procentową wartość odchylenia w zalezności od wielkości instancji dla 3 algorytmów

Analiza:

Na wykresie można zauważyć ponowny wpływ instancji o wielkości 180 na wartość odchylenia dla algorytmu „closest neighbour”, natomiast instancja ta nie ma znaczącego wpływu na wartość odchylenia dla algorytmów „2 OPT” oraz „Extended closest neighbour”.



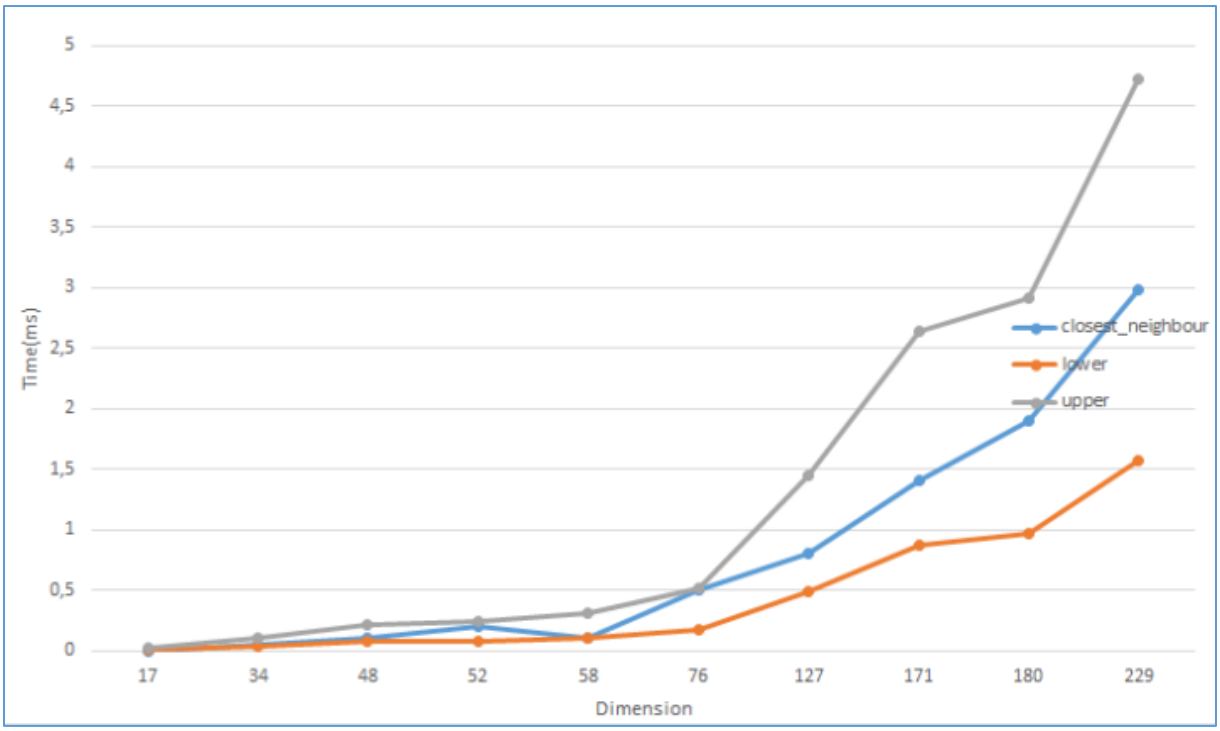
Wykres 2.2 (bez algorytmu clocest neighbour)

Wykres dla tych samych algorytmów ale bez „closest neighbour” dostarcza więcej informacji na temat dominacji algorytmu „2 OPT” nad „Extended closest neigbour”. Mianowicie wartości odchylenia dla „2 OPT” są zawsze mniejsze niż dla „Extended closest neighbour”. Wynika to z tego, że w naszym przypadku wartością na wejściu algorytmu „2 OPT” jest wynik zwracany przez „Extended closest neighour”, a następnie jest on jeszcze poprawiany.

3. Złożoność czasowa algorytmów

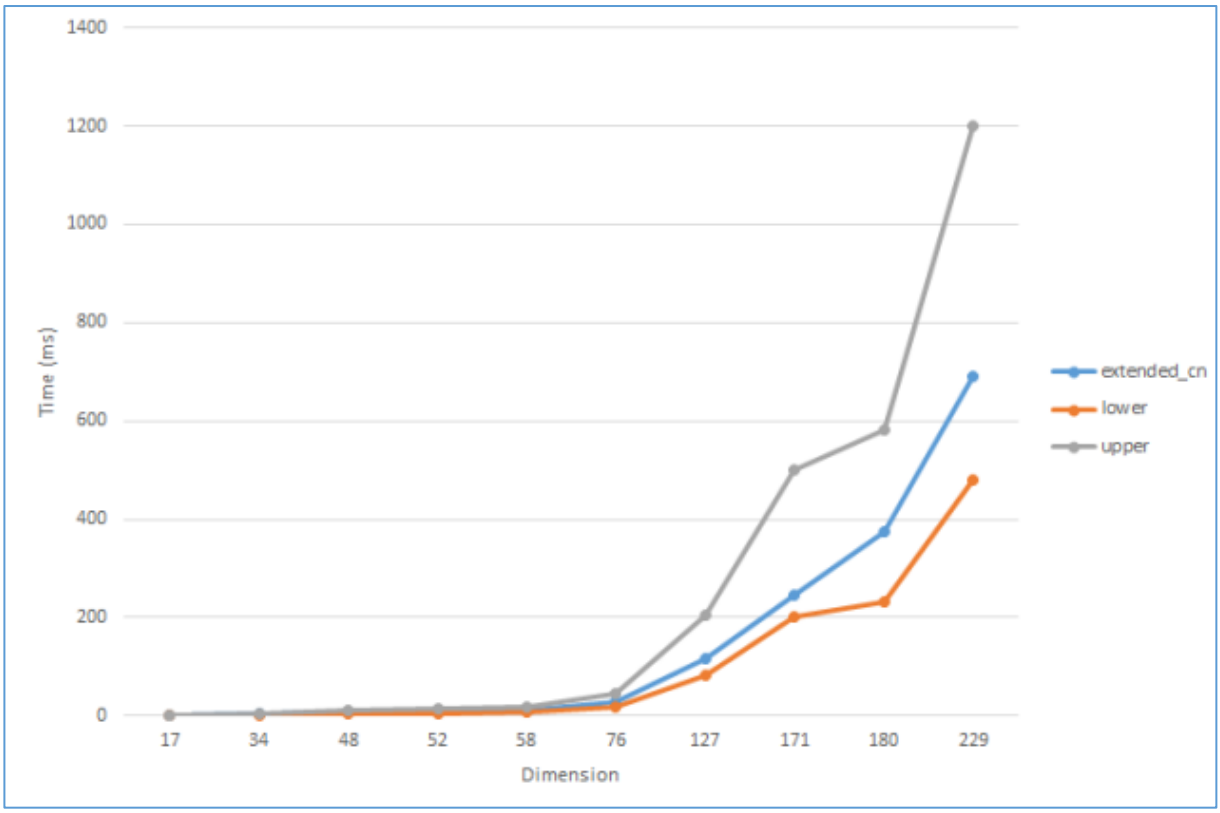
W tym punkcie analizować będziemy złożoność czasową wcześniej wymienionych algorytmów.

Dla każdego algorytmu przed stworzeniem wykresu analizowaliśmy analizie nasz kod, aby oszacować jaka powinna być złożoność czasowa danego algorytmu.



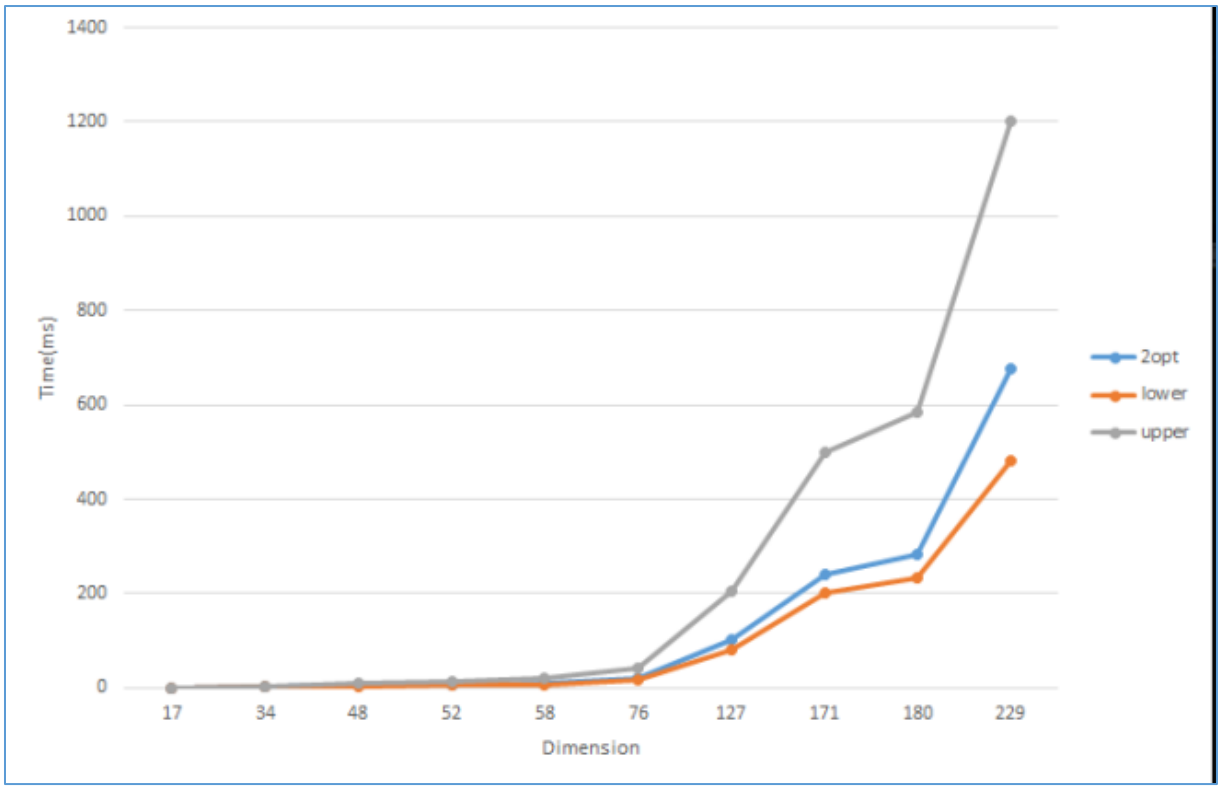
Wykres 3.1 opisujący złożoność algorytmu closest neighbour

Wykres czasu w zależności od wielkości instancji dla algorytmu „Closest neighbour" ograniczyliśmy z góry funkcją daną wzorem $f(x) = \frac{9}{100000}x^2$ natomiast z dołu funkcją daną wzorem $f(x) = \frac{3}{100000}x^2$. Zatem można wywnioskować, że złożoność czasowa algorytmu „Closest neighbour" wynosi n^2.



Wykres 3.2 opisujący złożoność algorytmu extended closest neighbour

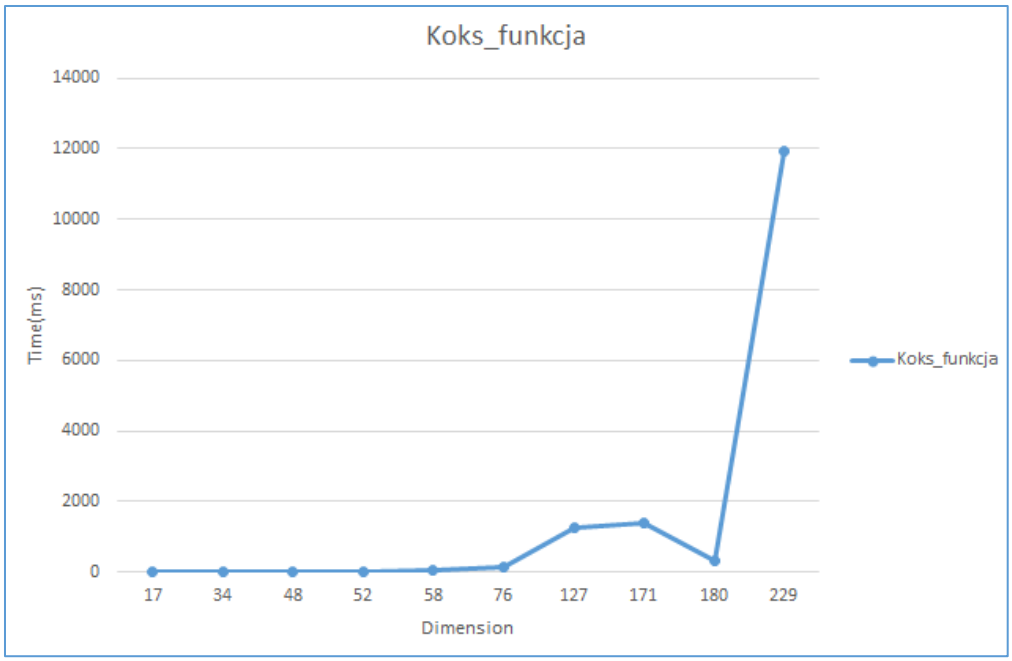
Wykres czasu w zależności od wielkości instancji dla algorytmu „Extended closest neighbour" ograniczyliśmy z góry funkcją daną wzorem $f(x) = \frac{4}{10000}\ x^3$ natomiast z dołu funkcją daną wzorem $f(x) = \frac{1}{10000}\ x^3$. Zatem można wywnioskować, że złożoność czasowa algorytmu „Closest neighbour" wynosi n^3.



Wykres 3.3 opisujący złożoność algorytmu 2 opt

Wykres czasu w zależności od wielkości instancji dla algorytmu „2 OPT" ograniczyliśmy z góry funkcją daną wzorem $f(x) = \frac{4}{10000}\ x^3$ natomiast z dołu funkcją daną wzorem $f(x) = \frac{1}{10000}\ x^3$. Zatem można wywnioskować, że złożoność czasowa algorytmu „Closest neighbour" wynosi n^3.

4. EKSTRA WYKRES


Koks_funkcja
14000
12000
10000
8000
6000
4000
2000
0
Time(ms)
17
34
48
52
58
76
127
171
180
229
Dimension
Koks_funkcja