Badania operacyjne i wspomaganie decyzji

IMPLEMENTACJA ALGORYTMU TABU SEARCH DLA PROBLEMU MINIMALIZACJI KOSZTÓW UTWORZENIA SIECI WODOCIAGÓW

Aleksander Pasiut
Michał Trojnarski

Ogólne założenia zadania

Celem zadania jest utworzenie sieci wodociągów między miastami, a stacjami dystrybucji wody. Każde miasto ma określone zapotrzebowanie na wodę, które musi być zaspokojone, a każda stacja ma określoną wydajność, której nie może przekroczyć. Dane mamy również rodzaje wodociągów jakimi można połączyć stacje z miastami. Każdy rodzaj definiowany jest poprzez przepustowość i koszt jednostkowy. Dodatkowo utworzenie każdego połączenia pociąga za sobą koszt utworzenia systemu pomp, który będzie je obsługiwał. Celem jest wybór takich połączeń i takich ich rodzajów, aby koszt był jak najmniejszy. Dopuszczalne jest używanie miast, jako punktów tranzytowych w przekazywaniu wody do innego miasta.

Opis formalny

Dany mamy zbiór obiektów:

$$\Omega = \{\Theta_1 \dots \Theta_n\}$$

Każdy obiekt określony jest przez trzy parametry:

- położenie na płaszczyźnie (wyrażone poprzez dwuwymiarowy wektor; współrzędne wyrażone przy pomocy jednostek długości):
$$l(\Theta)$$
- zapotrzebowanie własne (wyrażone w jednostkach objetości na jednostkę czasu; równe 0, w przypadku, gdy obiekt jest stacją dystrybucji wody):
$$z(\Theta)$$
- wydajność własna (wyrażona w jednostkach objętości na jednostkę czasu; równa 0, w przypadku, gdy obiekt jest miastem):
$$w(\Theta)$$

Zakładamy, że sumaryczna wydajność wszystkich obiektów jest niemniejsza od sumarycznego zapotrzebowania wszystkich obiektów:

$$\sum_{i=1}^{n} w(\Theta_i) \geq \sum_{i=1}^{n} z(\Theta_i)$$

Dany mamy zbiór rodzajów rur:

$$\Omega_R = \{R_1 \dots R_z\}$$

Każdy rodzaj określony jest przez dwa parametry:

- przepustowość (wyrażona w jednostkach objętości na jednostkę czasu):
$$p(R)$$
- cena (wyrażona w jednostkach pieniężnych na jednostkę długości):
$$c(R)$$

Dany mamy zbiór połączeń:

$$\Omega_P = \{(i,j) \in \{1 \dots n\} \times \{1 \dots n\}, i \neq j; k \in \{1 \dots z\}: P_{ijk}\}$$

Każde połączenie jest uporządkowaną trójką zawierającą obiekt źródłowy, obiekt docelowy i rodzaj rury:

$$P_{ijk} = (\Theta_i, \Theta_j, R_k)$$

Dla każdego połączenia można wyznaczyć jego przepustowość, która jest równa przepustowości rodzaju rury, z jakiego wykonane jest połaczenie:

$$p(P_{ijk}) = p(R_k)$$

Można również wyznaczyć koszt, który jest równy długości połączenia pomnożonej przez cenę jednostkową rury, z jakiej połączenie jest wykonane powiększony o koszt utworzenia w obiekcie źródłowym dodatkowego systemu pomp, który jest zależny od przepustowości połączenia:

$$c(P_{ijk}) = |l(\Theta_i) - l(\Theta_j)| \cdot c(R_k) + g\left(|l(\Theta_i) - l(\Theta_j)|, p(R_k)\right)$$

Funkcja g jest zależnością między przepustowością połączenia i jego długoscią, a kosztem systemu pomp. Może to być zależność liniowa:

$$g(x, y) = g_1 x + g_2 y$$

Załóżmy, że $In(i)$ jest zbiorem połączeń wchodzących do obiektu Θ_i, a $Out(i)$ jest zbiorem połączeń wychodzących z obiektu Θ_i. Formalnie:

$$\Omega_P \supset In(i) = \{P_{z_1 i k_1} \dots P_{z_s i k_s}\}$$
$$\Omega_P \supset Out(i) = \{P_{i z_1 k_1} \dots P_{i z_t k_t}\}$$

Dla każdego obiektu musi być spełniona zależność:

$$\forall i: \Theta_i \in \Omega: w(\Theta_i) + \sum_{P \in In(i)} p(P) \geq z(\Theta_i) + \sum_{P \in Out(i)} p(P)$$

Zadanie to zminimalizować wartość funkcji celu będącej sumarycznym kosztem utworzenia sieci połączeń poprzez wybór najlepszego podzbioru połączeń Π:

$$f(\Pi) \sum_{P \in \Pi} c(P) = min, \qquad \Pi \subset \Omega_P$$

Pojęcia niezbędne do zastosowania algorytmu Tabu Search

W przypadku zastosowania algorytmu w standardowej wersji Tabu Search konieczne jest uprzednie opracowanie algorytmu konstrukcyjnego. Ma on za zadanie zwrócić początkowe dowolne rozwiązanie dopuszczalne.

W przypadku rozwiązywanego problemu istnieją jednak sytuacje, kiedy samo znalezienie rozwiązania dopuszczalnego jest problemem samym w sobie. Konieczność stosowania algorytmu konstrukcyjnego można jednak ominąć stosując funkcję kary.

Funkcja celu przybiera wtedy postać:

$$f(\Pi) = \sum_{P \in \Pi} c(P) + k(\Pi) = min, \Pi \subset \Omega_P$$

gdzie k jest funkcją kary.

Opis wyznaczenia funkcji kary znajduje się w rozdziale "Funkcja kary".

Dla zastosowania algorytmu Tabu Search konieczne jest jeszcze zdefiniowanie kilku dodatkowych pojęć: operacji poruszania się po przestrzeni rozwiązań i sąsiedztwa danego rozwiązania.

Biorąc pod uwagę, że rozwiązanie jest podzbiorem zbioru wszystkich połączeń $\Pi \subset \Omega_P$, można zdefiniować następujące trzy operacje:

- dodanie do rozwiązania Π połączenia P tworząc rozwiązanie Π^*:
$$\Pi^* = \Pi \cup \{P\}$$
- usunięcie z rozwiązania Π połączenia P tworząc rozwiązanie Π^*:
$$\Pi^* = \Pi \setminus \{P\}$$
- zastąpienie połączenia P_1 połączeniem P_2 w rozwiązaniu Π tworząc rozwiązanie Π^*:
$$\Pi^* = \Pi \cup \{P_2\} \setminus \{P_1\}$$

Sąsiedztwo rozwiązania Π będzie zdefiniowane jako zbiór wszystkich rozwiązań, które można uzyskać poprzez wykonanie jednej z powyższych operacji na rozwiązaniu Π. W liście tabu umieszczane są dwuelementowe ciągi zbiorów jednoelementowych:

- $(\{\emptyset\},\{P\})$ - reprezentacja dodania połączenia P
- $(\{P\},\{\emptyset\})$ - reprezentacja usunięcia połączenia P
- $(\{P_1\},\{P_2\})$ - reprezentacja zastąpienia połączenia P_1 połączeniem P_2

Funkcja kary

Funkcja kary musi dostatecznie pogarszać wartość funkcji celu w sytuacji, gdy otrzymane rozwiązanie jest niedopuszczalne, aby zapewnić ostateczne osiągnięcie rozwiązania dopuszczalnego.
Aby jednak wiedzieć jakiego rzędu wielkości powinniśmy oczekiwać od funkcji kary, konieczna jest znajomość rzędu wielkości wartości funkcji celu przy rozwiązaniu optymalnym.
Wielkość tą można oszacować na kilka sposobów Skoro koszt utworzenia pojedynczego połączenia wyraża się funkcją:

$$c(P_{ijk}) = |l(\Theta_i) - l(\Theta_j)| \cdot c(R_k) + g\left(|l(\Theta_i) - l(\Theta_j)|, p(R_k)\right)$$

to wtedy oszacować można w następujący sposób koszt rozwiązania optymalnego:

$$c_T(\Omega, \Omega_R) = \sum_{\Theta_c \in \Omega_C} \left(c(z(\Theta_c))l(\Theta_c) + g(l(\Theta_c), z(\Theta_c))\right)$$

gdzie:
Ω_S - zbiór stacji dystrybucji wody
Ω_C - zbiór miast
$l(\Theta_C)$ - odległość od miasta Θ_c do najbliższej stacji dystrybucji wody
$c(z(\Theta_c))$ - średnia cena jednostkowa rury uzyskana poprzez interpolację / ekstrapolację zależności przepustowość - cena; żądana przepustowość jest równa zapotrzebowaniu miasta podzielonemu przez liczbę stacji dystrybucji wody
Funkcja kary wyraża się zatem wzorem:

$$k(\Pi) = k_f c_T(\Omega, \Omega_R) \sum_{i \in \{1 \ldots n\}} max\left(z(\Theta_i) + \sum_{P \in Out(i)} p(P) - w(\Theta_i) - \sum_{P \in In(i)} p(P), 0\right)$$

k_f - współczynnik podawany przez użytkownika

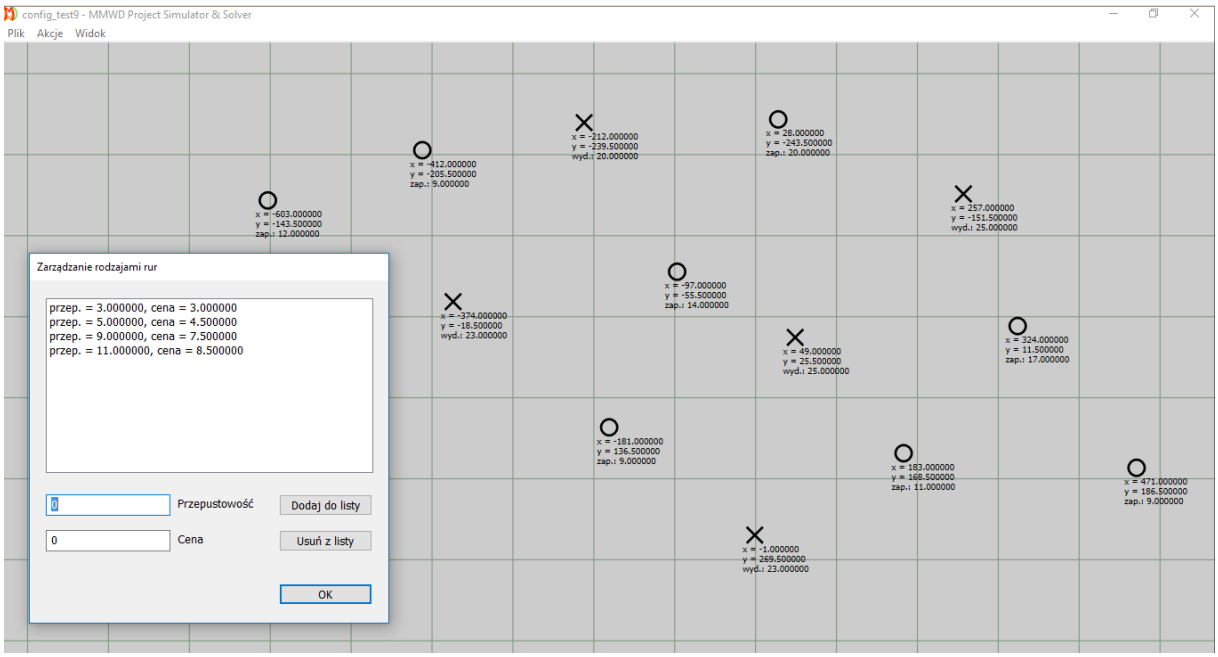
Implementacja algorytmu

Program został zaimplementowany w aplikacji okienkowej przeznaczonej dla systemów operacyjnych Windows 7 i nowszych. Cały program został napisany w języku C++ w środowisku Visual Studio 2015. Aplikacja udostępnia interfejs graficzny, który umożliwia wygodne ustawienie obiektów na płaszczyźnie, przypisanie im wartości, zdefiniowanie rodzajów połączeń, ustawienie parametrów algorytmu Tabu Search, wyświetlenie rozwiązania, a także eksport przebiegu funkcji celu podczas wykonywania algorytmu i możliwość zapisu konfiguracji.

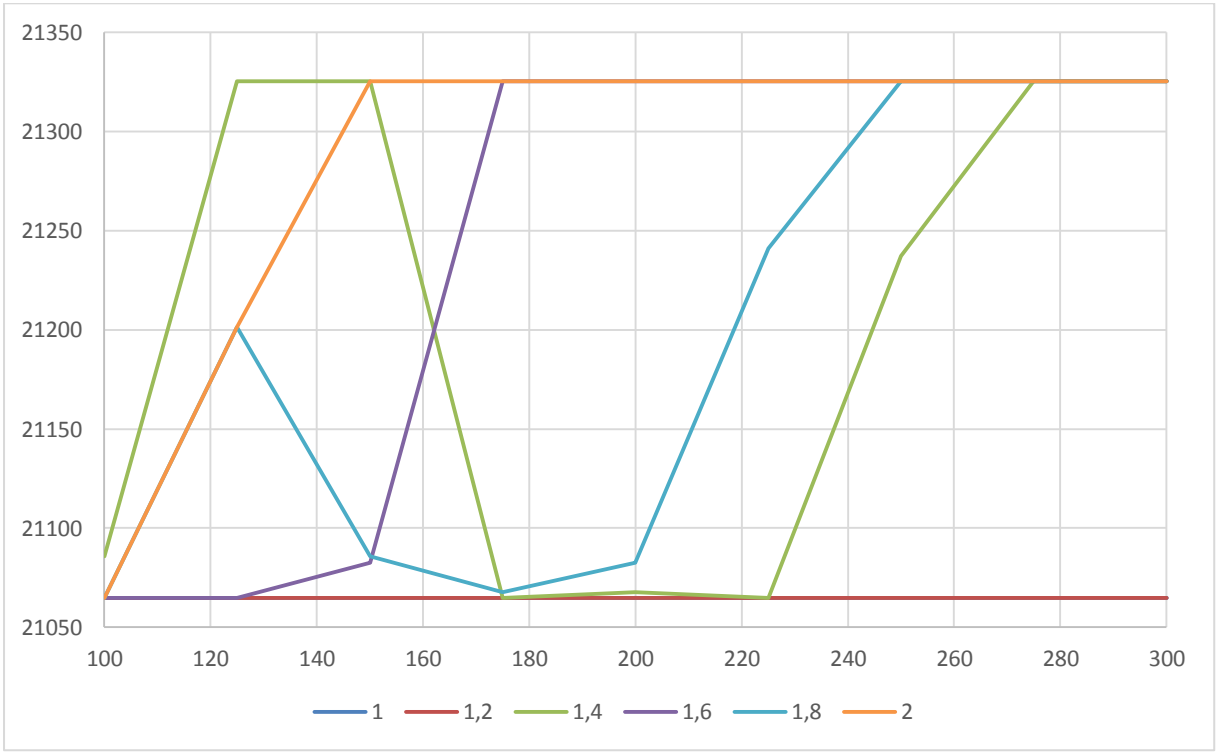
Testy algorytmu

Test 1

Analizowana konfiguracja:

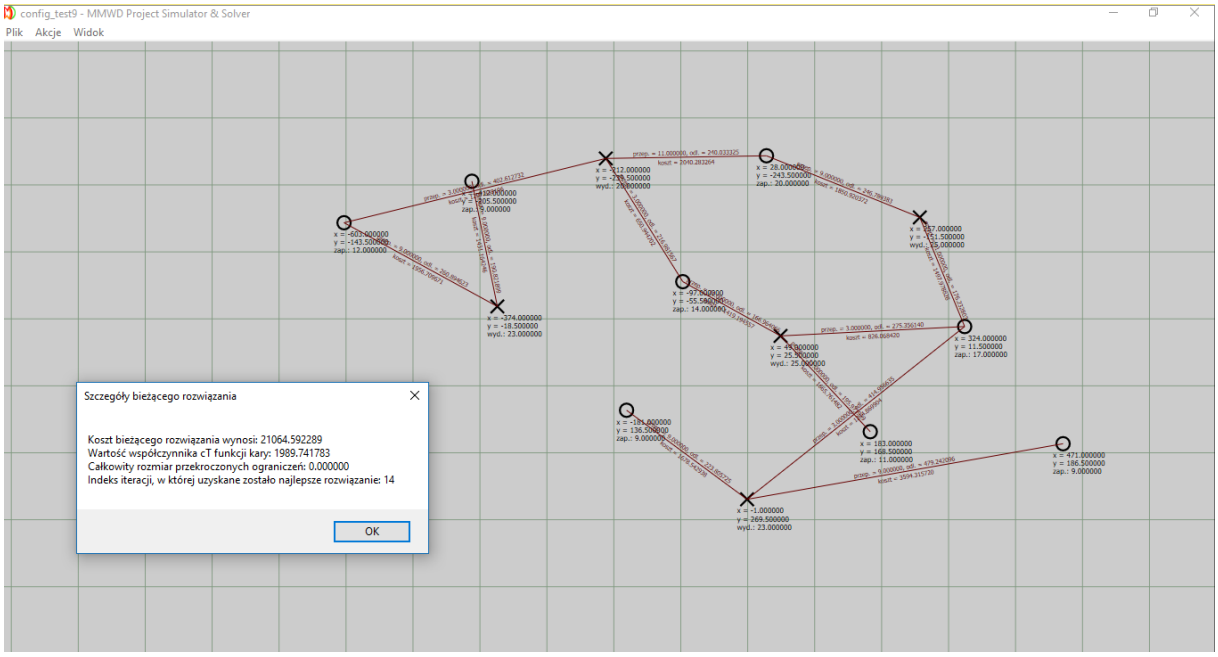


Zależność uzyskanego najlepszego rozwiązania od współczynnika k_f i rozmiaru listy tabu (1500 iteracji):



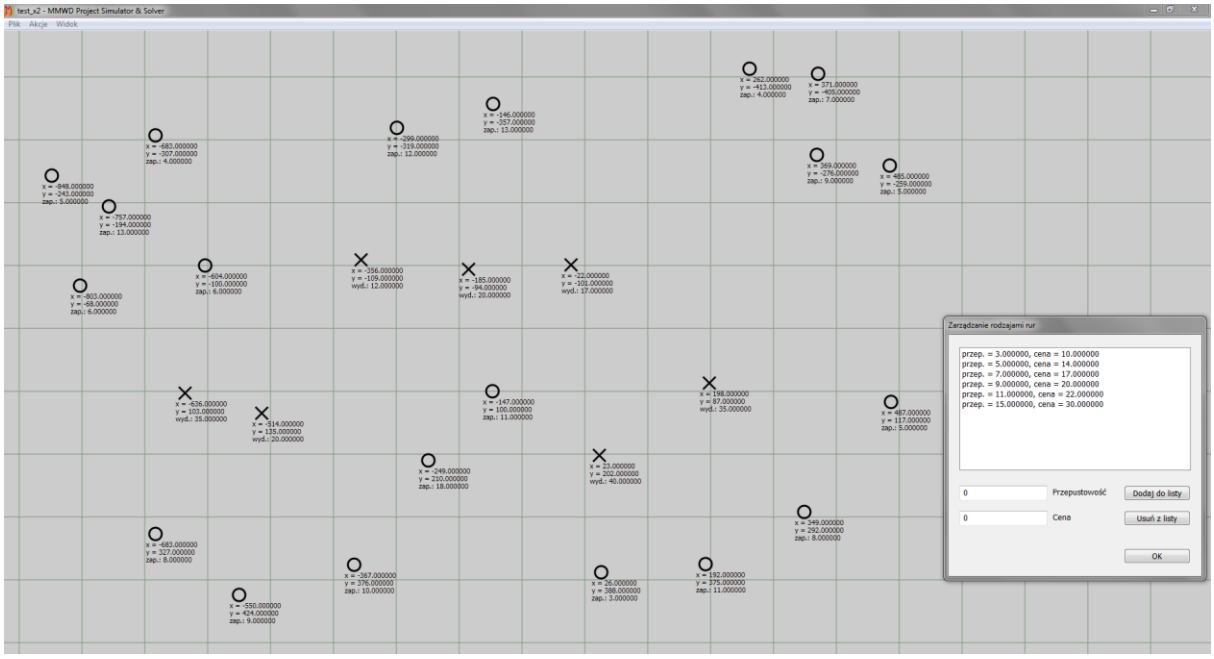
Można zaobserwować, że dla małych współczynników funkcji kary (1, 1.2) funkcja dla każdego z przetestowanych rozmiarów list tabu osiąga najlepsze rozwiązanie. Dla większych współczynników występuje zależność: im większy rozmiar listy tabu, tym uzyskiwane rozwiązanie jest gorsze.

Najlepsze uzyskane rozwiązanie:

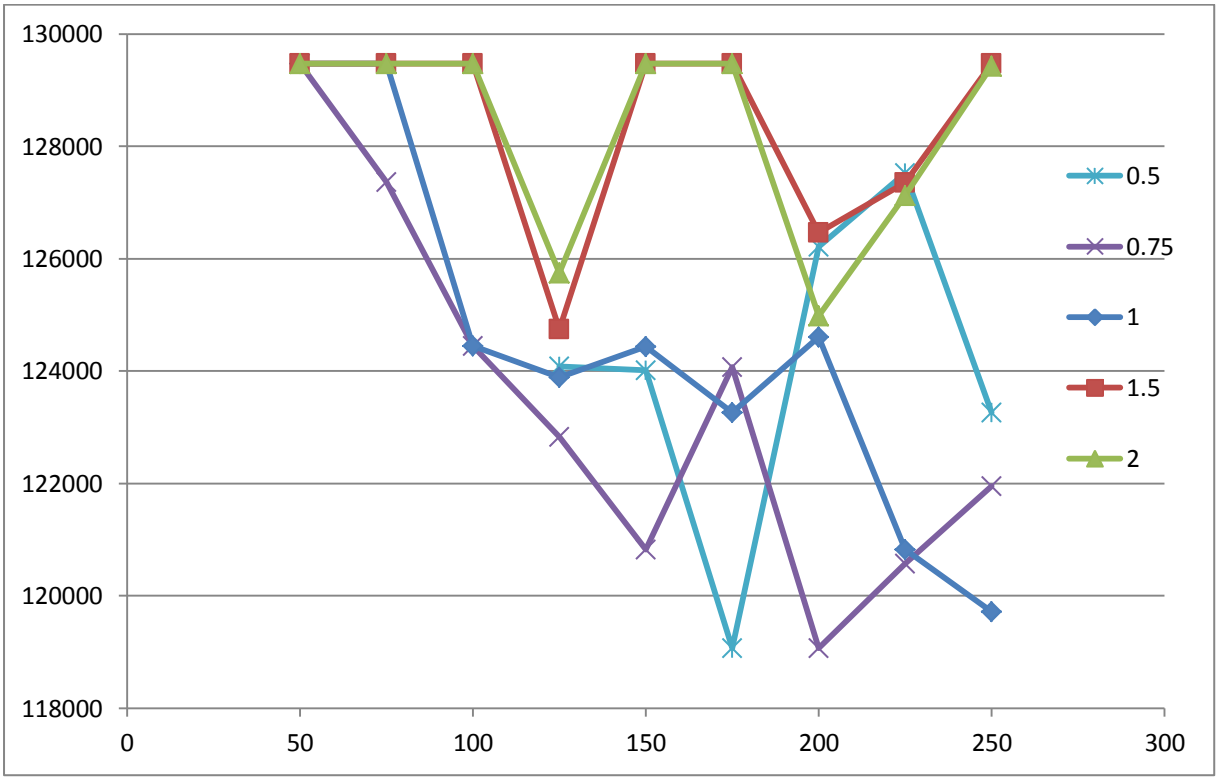


Test 2

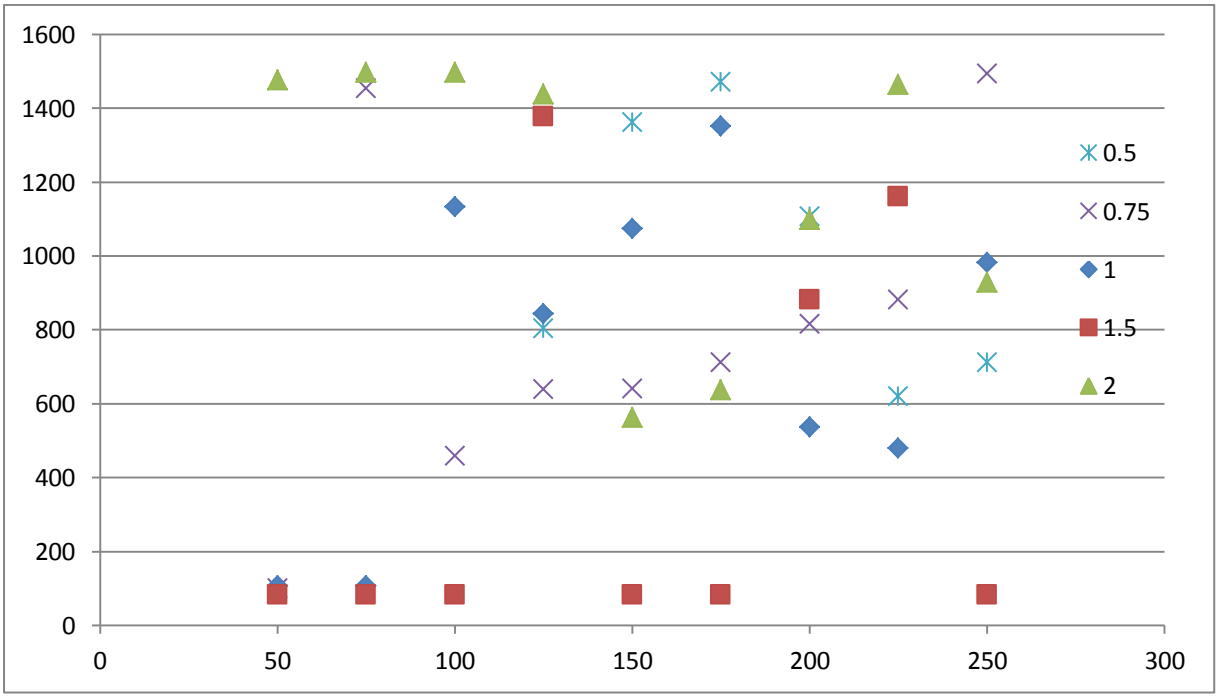
Analizowana konfiguracja:



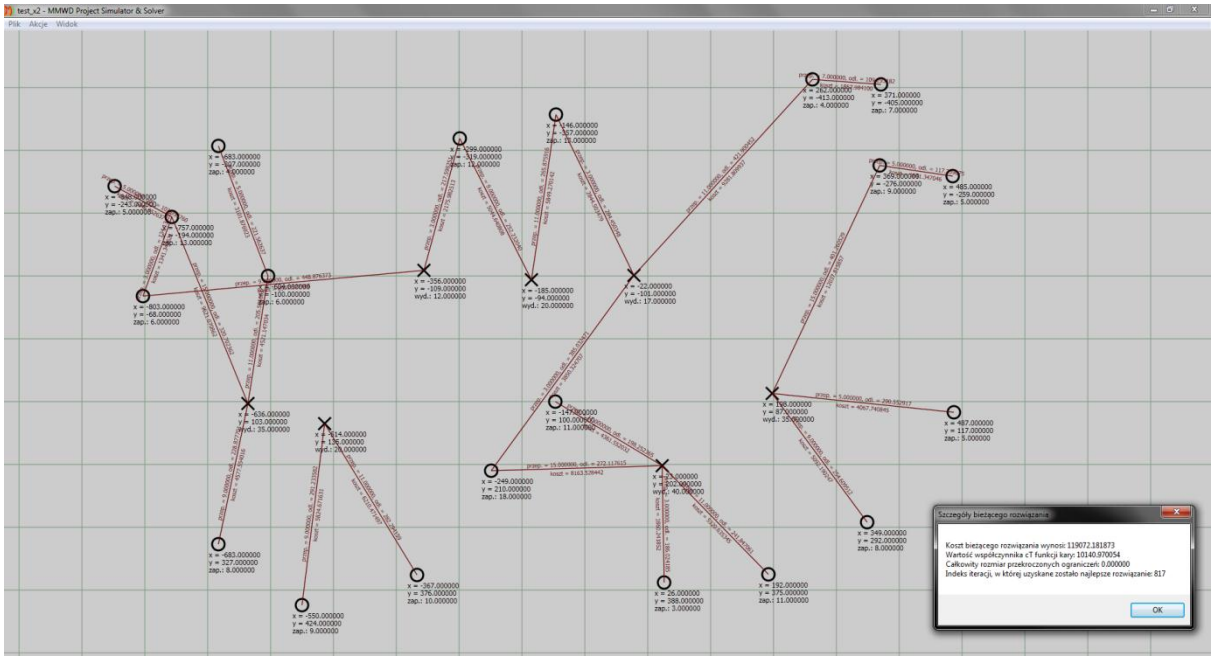
Zależność uzyskanego najlepszego rozwiązania od współczynnika k_f i rozmiaru listy tabu (1500 iteracji):



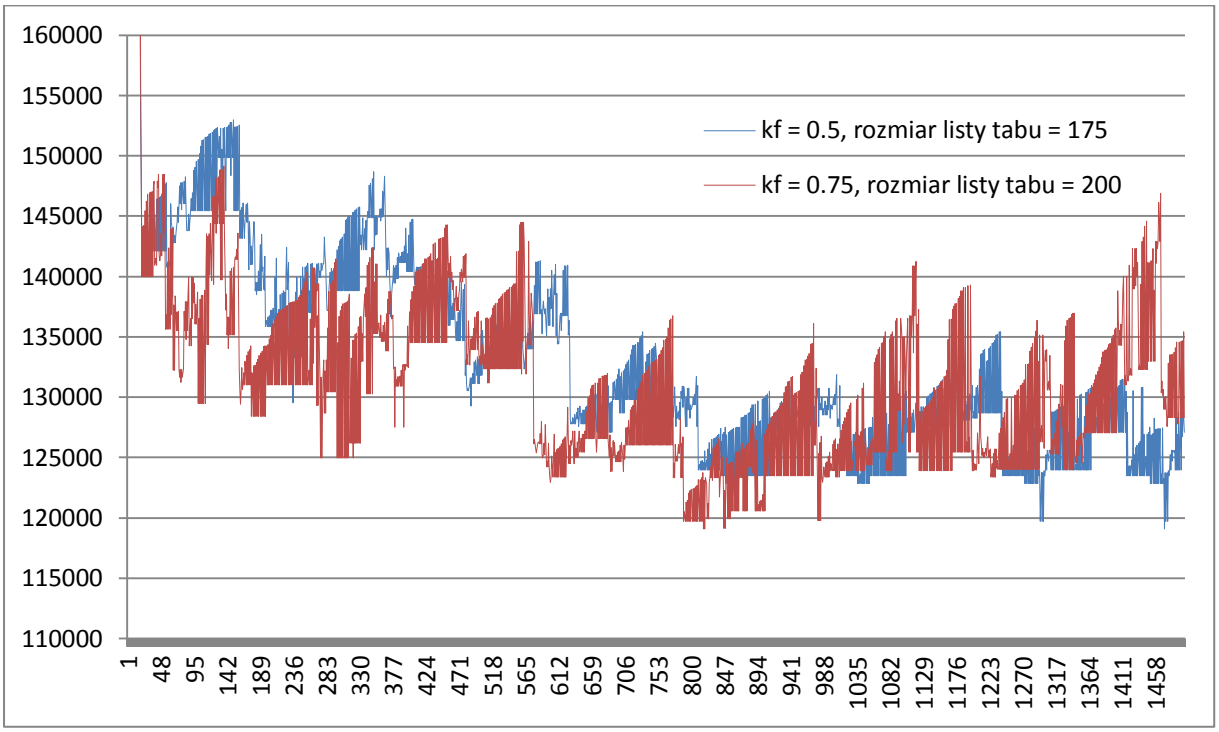
Zależność między rozmiarem listy tabu, a indeksem iteracji, w której uzyskano najlepsze rozwiązanie:



Najlepsze uzyskane rozwiązanie:



Porównanie przebiegów funkcji celu dla różnych parametrów, które niezależnie doprowadziły do powyższego rozwiązania:



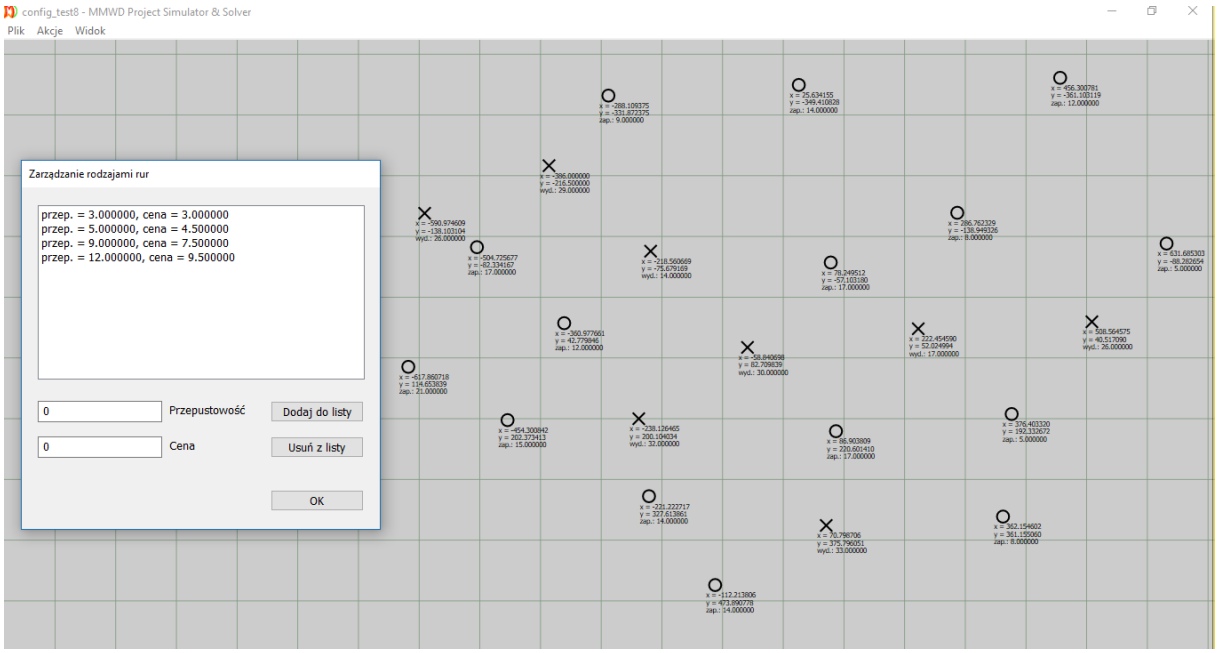
Na podstawie wyników testu można wywnioskować, że zmniejszanie współczynnika k_f polepsza działanie algorytmu. Przesadne zmniejszenie tej wartości powoduje jednak, że algorytm ma trudności z uzyskaniem rozwiązania dopuszczalnego.

Zależność indeksu iteracji, w której uzyskano najlepsze rozwiązanie od rozmiaru listy tabu wykazuje bardzo słabą korelację obu tych wartości. Współczynnik korelacji liniowej tych wartości wyniósł 0,16.

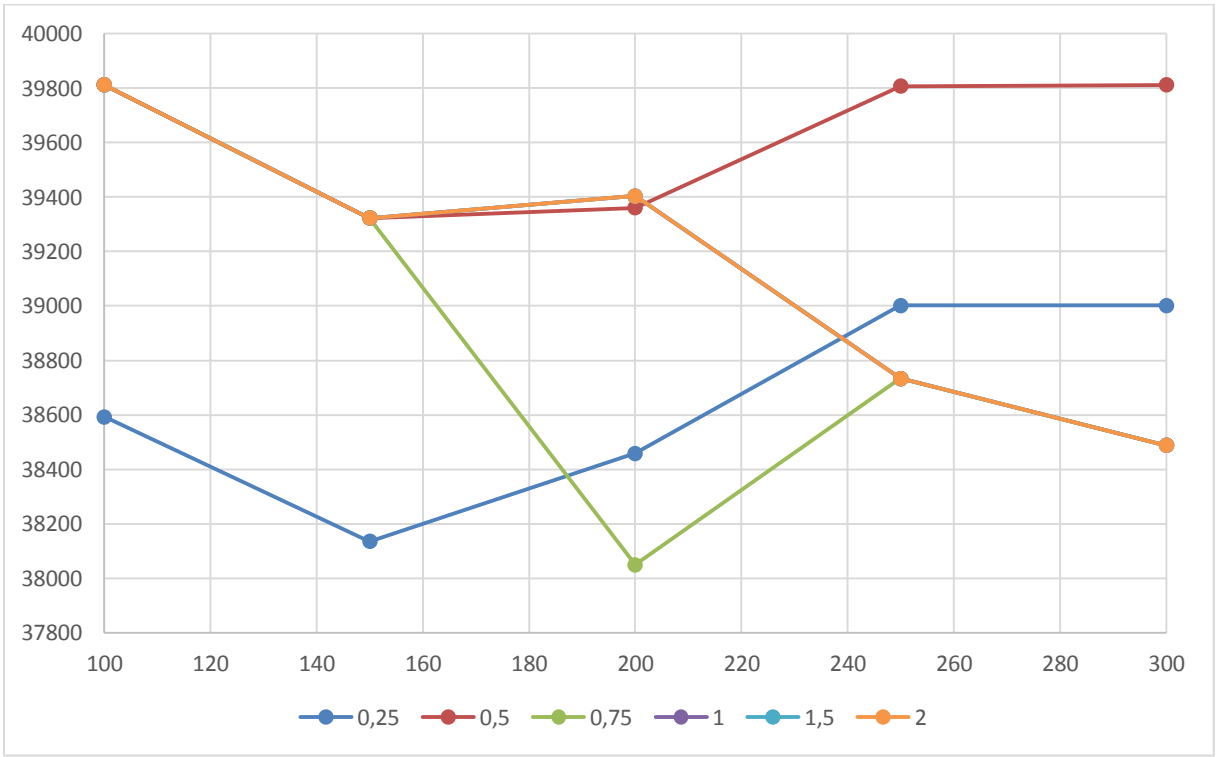
Wykresy przedstawiające przebieg funkcji celu prezentują dobrze sposób działania algorytmu, a szczególnie miejsca, w których algorytm próbuje wydostać się z minimum lokalnego (naprzemienne wartości rosnące i wartość w minimum lokalnym), a następnie gwałtowanie zmienia wartość ("przechodzi" do innego minimum lokalnego).

Test 3

Analizowana konfiguracja:

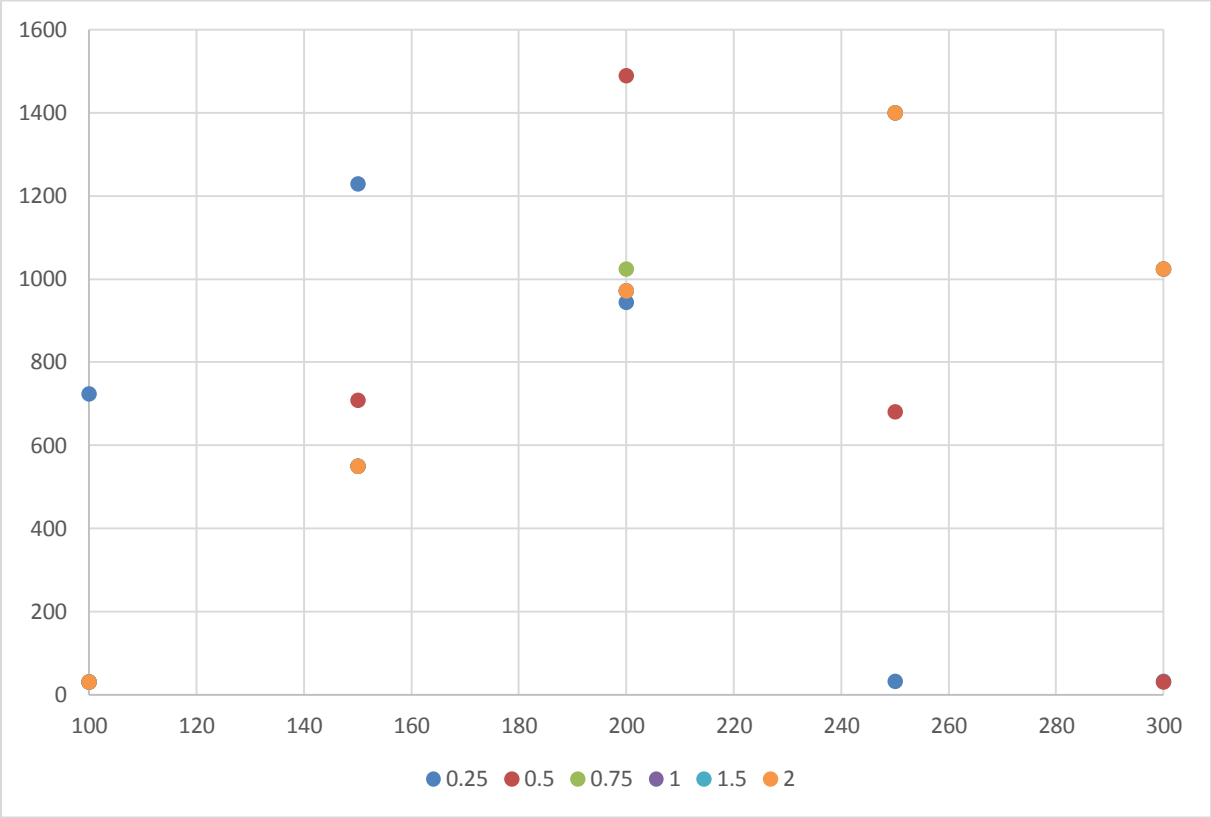


Zależność uzyskanego najlepszego rozwiązania od współczynnika k_f i rozmiaru listy tabu (1500 iteracji):

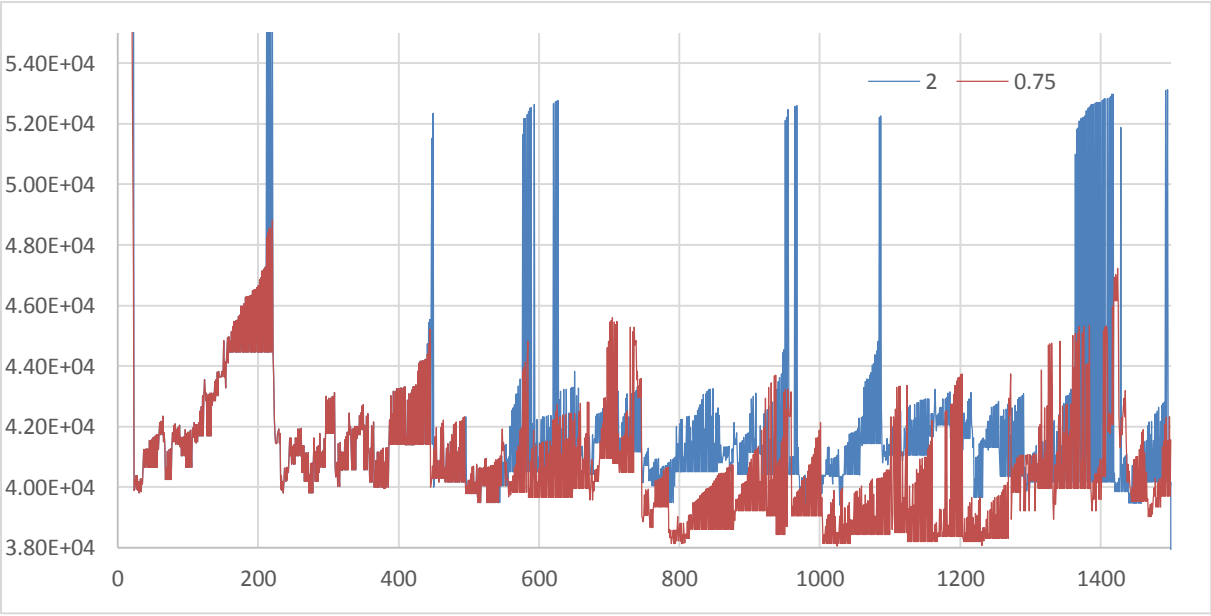


Dla wartości współczynnika k_f mniejszej niż 0.25 algorytm miał problem ze znalezieniem rozwiązania dopuszczalnego. Gdy lista tabu ma niewielki rozmiar, mały współczynnik k_f pozwala algorytmowi uzyskiwać dobre rozwiązania, co pozwala otrzymać wynik szybciej (większa lista tabu spowalnia program).

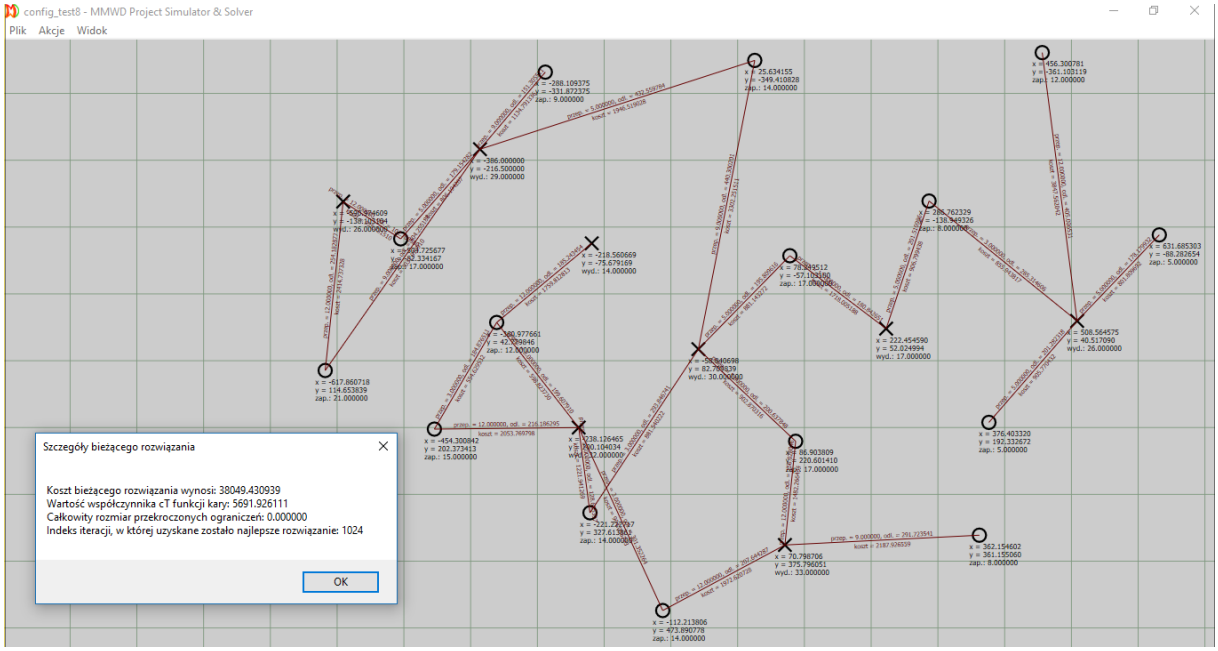
Zależność iteracji, w której osiągnięto najlepsze rozwiązanie od rozmiaru listy tabu:



Porównanie wartości funkcji celu dla rozmiaru listy tabu równego 200 oraz współczynników $k_f = 0.75$ oraz $k_f = 2$:



Najlepsze uzyskane rozwiązanie dla współczynnika k_f = 0.75:

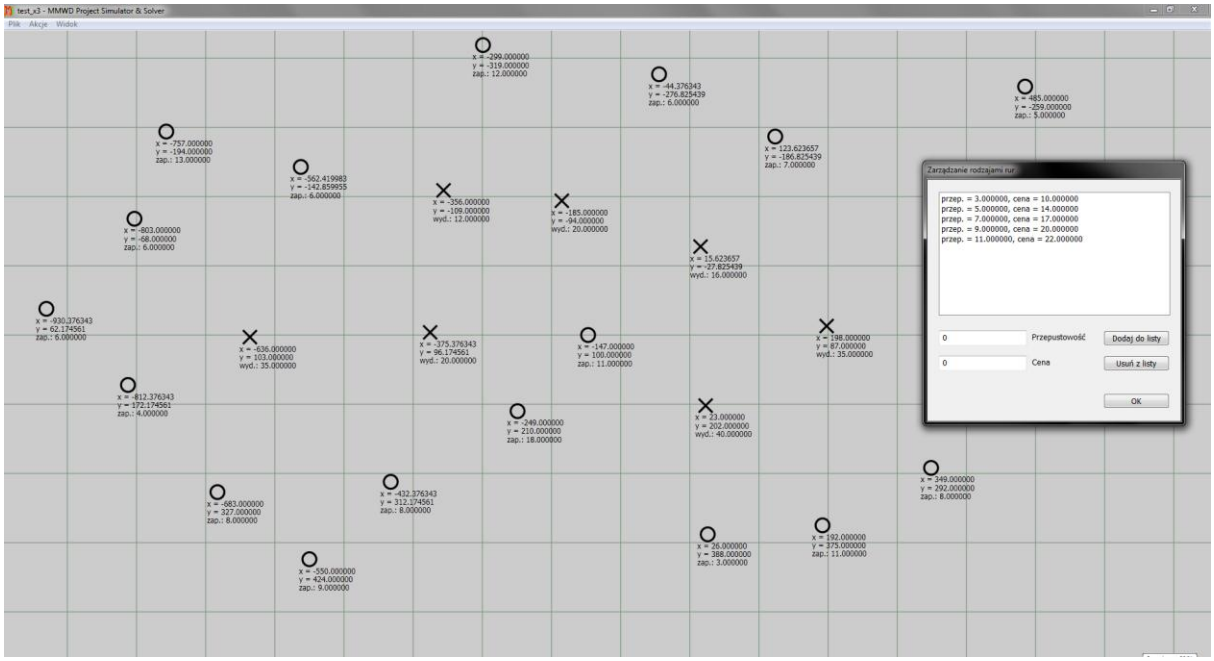


Mniejszy współczynnik k_f pozwolił uzyskać lepsze rozwiązanie. Z porównania dwóch przebiegów funkcji celu widać, że do momentu wypełnienie listy tabu (o rozmiarze 200) w obu przypadkach algorytm przebiegał niemal identycznie. Zauważalne duże skoki wartości funkcji celu dla współczynnika 2.0 pojawiają się w momencie osiągnięcia pewnego rozwiązania niedopuszczalnego – suma przekroczeń jest mnożona właśnie przez 2.

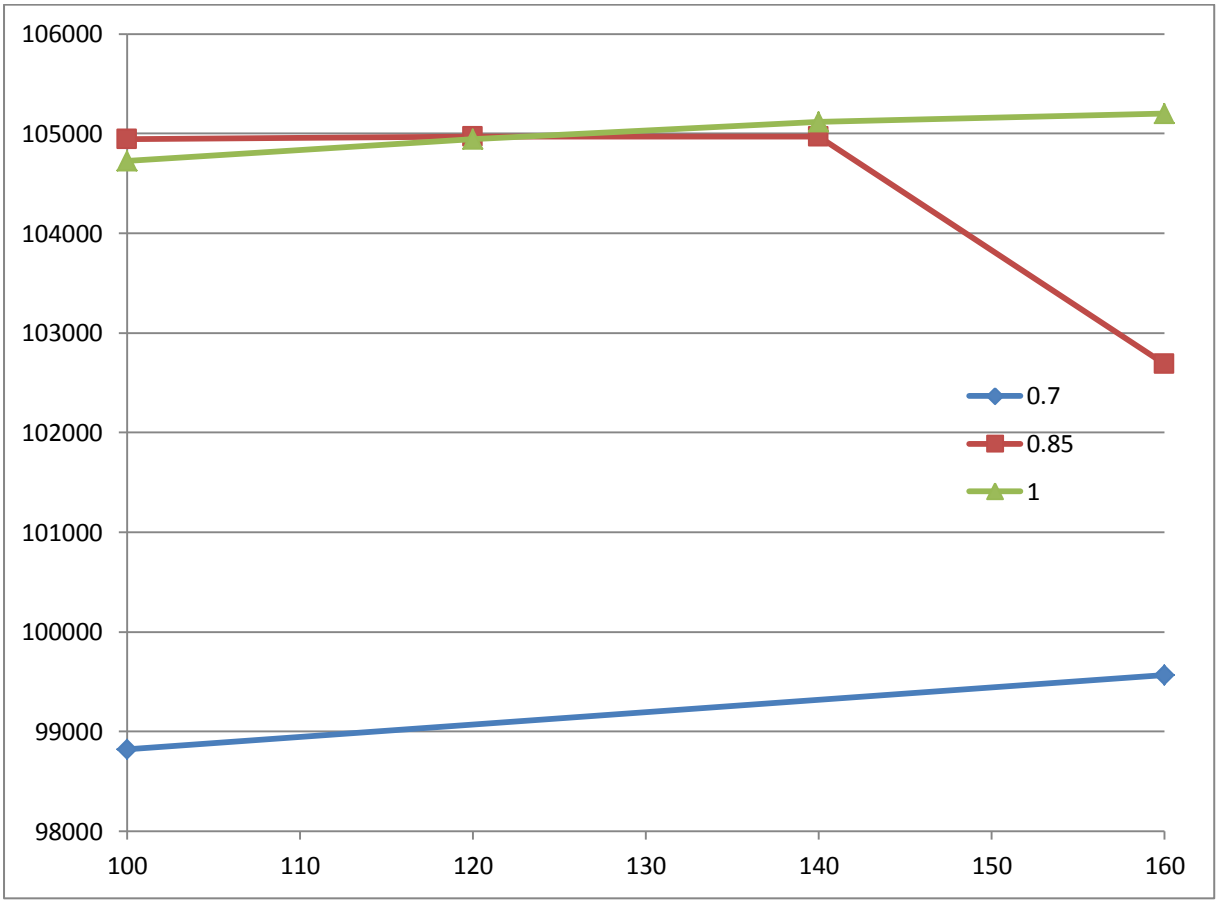
Współczynnik korelacji liniowej między indeksami iteracji, w których uzyskano najlepsze rozwiązanie, a rozmiarami listy taboo wyniósł 0,42. Podobnie jak w teście 2, korelacja między tymi wartościami jest słaba.

Test 4

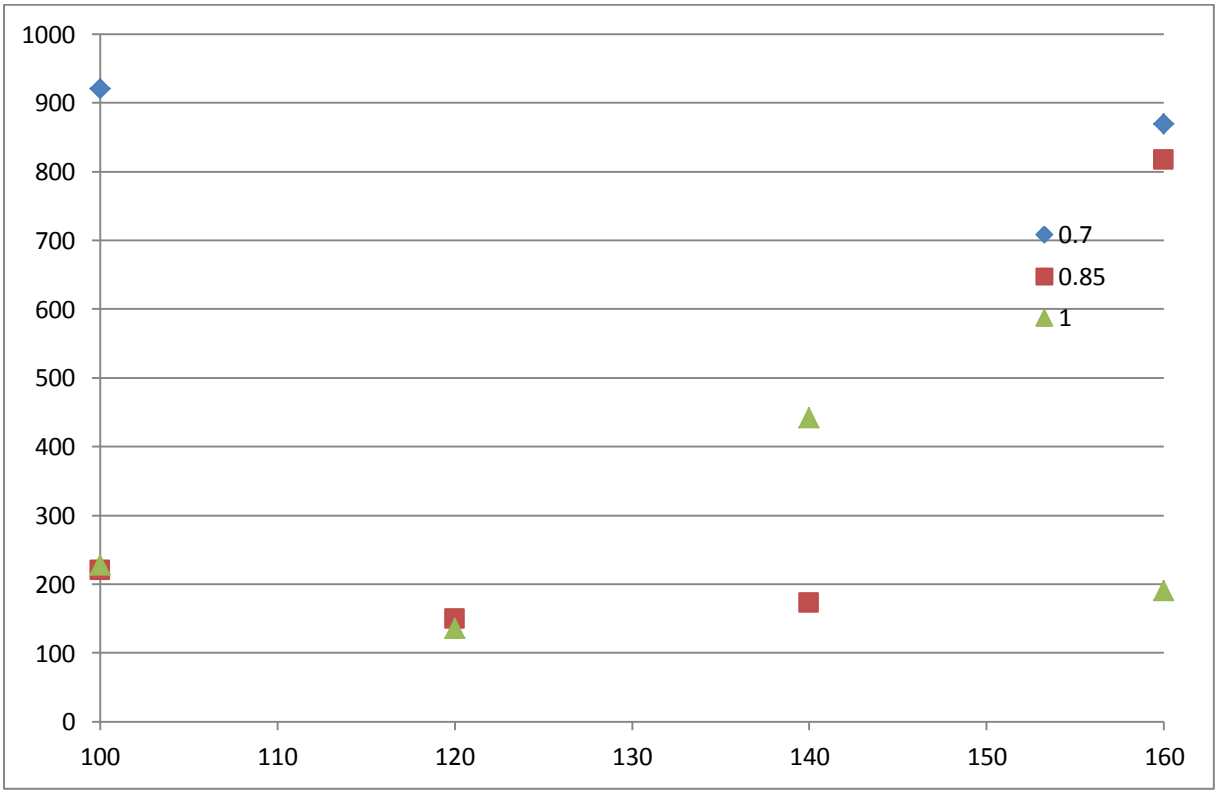
Analizowana konfiguracja:



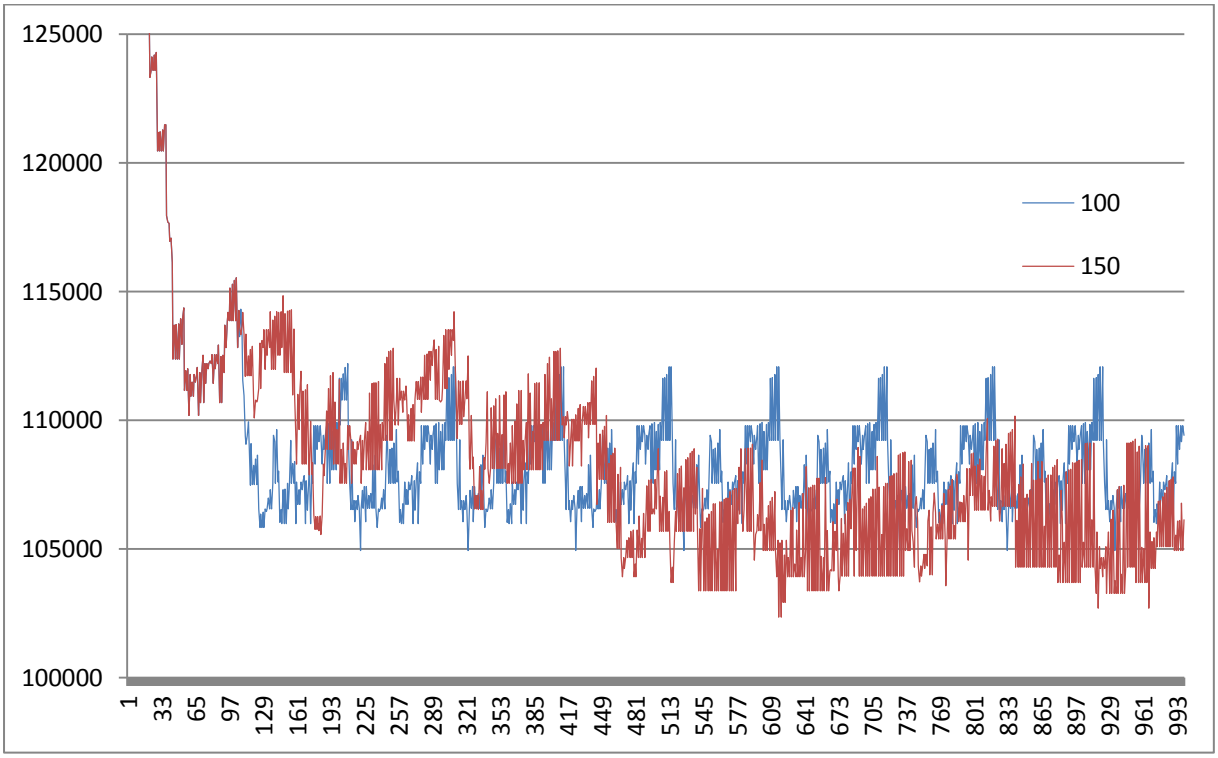
Zależność uzyskanego najlepszego rozwiązania od współczynnika k_f i rozmiaru listy tabu (1000 iteracji):



Zależność iteracji, w której osiągnięto najlepsze rozwiązanie od rozmiaru listy tabu:



Porównanie przebiegów funkcji celu dla $k_f = 0.85$ i różnych rozmiarów listy tabu:



W przedstawionym powyżej przebiegu funkcji celu dla rozmiaru listy tabu 100 widać jak od iteracji około 130 algorytm wpada w cykl. Długość tego cyklu jest równa rozmiarowi listy tabu. Jest to efekt zbyt małej listy tabu. Po zwiększeniu rozmiaru listy tabu do 150, przebieg funkcji celu w cykl już nie wpada.

Współczynnik korelacji liniowej między indeksem iteracji, w której uzyskano najlepsze rozwiązanie, a rozmiarem listy tabu wyniósł 0.25. Podobnie jak w testach 2 i 3 jest to korelacja słaba.