Planowanie tras przejazdów wózków widłowych po zakładzie produkcyjnym przy pomocy algorytmu ewolucyjnego

Piotr Zając

Łukasz Bednarek

Dawid Biel

1. Wprowadzenie do problemu

1.1. Opis analizowanego zadania

Przedmiotem optymalizacji jest dobór tras zadanej liczby wózków widłowych pomiędzy stanowiskami w zakładzie produkcyjnym, których położenie na hali można przybliżyć do punktów w odpowiednio zadanym układzie współrzędnych. Odległość między stanowiskami jest zadana metryką Manhattan. Każde stanowisko ma przypisany czas, jaki wózek widłowy musi poświęcić na jego obsługę (może to być np. wymiana ciężkiego kręgu blachy, która jest półproduktem lub podniesienie ciężkiej części aparatury produkcyjnej w celu dokonania dziennego przeglądu czujników, pobrania materiału do testów, itp.). Założeniem problemu jest, że wózki rozpoczynają pracę na początku układu współrzędnych, czyli w punkcie (0,0) i każdy ma określone tempo wykonywania pracy przy stanowisku. Pomiędzy stanowiskami wózki poruszają się z jednakową prędkością ze względu na ograniczenie prędkości. Celem optymalizacji jest takie dobranie tras wózków widłowych pomiędzy stanowiskami, aby czas pracy ostatniego pracującego wózka był **minimalny**.

Opisany wcześniej problem można zastosować do szeregu podobnych zadań. Jako przykład może posłużyć zadanie możliwie najszybszego dokonania rutynowego przeglądu poprawności działania systemów pomiarowych i innej aparatury w zakładzie produkcyjnym przez zespół automatyków lub metrologów o różnych zdolnościach (i co za tym idzie tempie wykonywania pracy).

1.2. Przyjęte uproszczenia

Na potrzeby ułatwienia rozwiązania problemu metodą przybliżoną przyjęto następujące uproszczenia, które nie zmieniają ogólnych założeń zadania:

- Odległość między stanowiskami jest zadana metryką prostokątną (która jest najbardziej sensowna w przypadku hali produkcyjnej).
- Wózki pomiędzy stanowiskami poruszają się z jednakową prędkością, a zużycie paliwa jest takie samo lub nie wpływa na postać rozwiązania. Jedyna istotna różnica występuje podczas obsługi stanowisk (prędkość ich obsługi jest dana współczynnikiem wózka).
- Stanowiska mają równy priorytet obsługi, nie występują kary ani nagrody za obsłużenie ich w jakiejś określonej kolejności.
- Odległość między stanowiskami jest zadana metryką prostokątną, nie są przewidziane awarie wózków, wzajemne się ich blokowanie, a drogi do stanowisk są na tyle szerokie, że wózki mogą się wyminąć.
- Każde stanowisko musi być obsłużone dokładnie raz i nie może być obsługiwane przez dwa wózki jednocześnie (np. w celu przyspieszenia).
- Wszystkie wózki rozpoczynają pracę w początku układu współrzędnych (0,0), który jest garażem i nie ma znaczenia, gdzie skończą pracę.
- Obliczana wartość na podstawie odległości między stanowiskami jest czasem potrzebnym do przejechania i obsługa stanowiska jest również dana czasem, dlatego minimalizacji poddawany jest czas pracy wózków.
- Stanowiska z przypisanym czasem obsługi równym 0 nie trzeba obsługiwać.

2. Model matematyczny

2.1. Dane wejściowe

Do danych wejściowych, na bazie których będzie optymalizowany problem należą:

- Liczba wózków widłowych obsługujących halę produkcyjną wraz z ich współczynnikami prędkości. Format danej wejściowej jest następujący:

  $$\alpha_{1,2,\ldots,n} = [1.2, 0.8, 1.1, 1, 1.3]$$

  Gdzie każda kolejna wartość oznacza współczynnik wózka widłowego. Liczba wózków jest równa długości wektora, w tym przypadku n = 5. Plik wsadowy jest plikiem tekstowym z wartościami rozdzielonymi znakiem spacji.

- Macierz reprezentująca rozkład stanowisk zakładu, gdzie współrzędne odpowiadają ich położeniu, natomiast wartość jest nominalnym czasem potrzebnym do obsługi stanowiska przez wózek o współczynniku równym 1. Opisywana macierz może mieć przykładowo postać:

4	0	0	0	0	K_6=50
3	0	K_5=30	0	0	0
2	K_4=25	0	0	0	0
1	0	0	0	K_3=5	0
0	Miejsce startu	0	K_1=10	0	K_2=15
y/x	***0***	***1***	***2***	***3***	***4***

Gdzie $K_{1,\ldots,6}$ są stanowiskami wymagającymi obsłużenia z nominalnymi czasami obsługi zawartymi w macierzy.

Plik tekstowy z zawartą macierzą wejściową ma następującą postać:

```
0 0 10 0 15
0 0 0 5 5
25 0 0 0 0
0 30 0 0 0
0 0 0 0 50
```

Gdzie numer wiersza odpowiada całkowitej wartości współrzędnej y, natomiast w jednym wierszu znajdują się kolejne wartości odpowiadające kolejnym całkowitym współrzędnym x.

2.2. Postać rozwiązania

Aby otrzymać postać rozwiązania, którą w łatwy sposób będzie można modyfikować na potrzeby algorytmu, macierz wejściowa zostaje w programie przekonwertowana na inną postać, mianowicie:

$$input = [(1, x_1, y_1, k_1), (2, x_2, y_2, k_2), \dots, (m, x_m, y_m, k_m)]$$

Gdzie m jest liczbą stanowisk z niezerowym czasem obsługi (czyli z koniecznością odwiedzenia), natomiast i-ta wartość w wektorze *input* zawiera *numer* stanowiska oraz jego współrzędne i koszt. Stanowiska zostały *ponumerowane* w ten sposób, że zaczynając od punktu (0,0), kolejne elementy wektora *input* odpowiadają stanowiskom z rosnącą współrzędną ***x*** i po osiągnięciu zakresu zwiększana jest współrzędna ***y***. Dla macierzy z wcześniejszego przykładu wektor *input* ma następującą postać:

$$[K_1 = (1,2,0,10), K_2 = (2,4,0,15), K_3 = (3,3,1,5), (4,0,2,25), (5,1,3,30), K_6 = (6,4,4,50)]$$

Przy takiej reprezentacji stanowisk można wprowadzić następującą postać rozwiązania:

$$X \in R^{(m+n)} = [0, k_{1_{1_1}}, k_{1_{2_1}}, \dots, k_{1_{s_1}}, 0, \dots, k_{n_{s_n}}]$$

Gdzie **0** jest znacznikiem nowego wózka, to znaczy wartości między dwoma znacznikami lub między znacznikiem, a końcem wektora są numerami stanowisk obsłużonych przez dany wózek odpowiadającymi numerom stanowisk we wcześniejszej reprezentacji *input*. Na podstawie takiej postaci rozwiązania można z wektora X i listy *input* stanowiącej o stanowiskach i ich czasach obsłużenia obliczyć funkcję celu. Taka postać rozwiązania została wprowadzona, aby można było na niej przeprowadzać operacje mutacji i krzyżowania typowe dla algorytmów ewolucyjnych.

Przykładowa postać rozwiązania problemu jest permutacją wartości od 1 do m, gdzie m jest liczbą stanowisk, znacznika pierwszego wózka na początku wektora i n-1 znaczników nowego wózka, gdzie n jest liczbą wózków.

$$X = [\mathbf{0}, 2, 4, 6, 8, \mathbf{0}, 3, 1, 7, \mathbf{0}, 9, 10, 11, \mathbf{0}, 14, 12, 13, 5]$$

2.3. Funkcja celu

Funkcja celu w rozważanym problemie ma być maksymalnym czasem osiągniętym przez wszystkie wózki i jest ona **minimalizowana**.

$$f(X) = \max\left[X_{i=1,..,n}: X_i = \sum_{j=1}^{s_j}\left(\propto_i \cdot k_{ij} + g(j, j-1)\right)\right] \rightarrow min$$

i = 1,…,n – dostępne wózki widłowe
j = 1,…,s_j – stanowiska obsłużone przez i-ty wózek
α_i – współczynnik szybkości obsługiwania stanowisk przez i-ty wózek
k_{ij} – czas obsługi j-tego stanowiska na liście i-tego wózka
g(j, j-1) – czas dojazdu do j-tego stanowiska ze stanowiska j-1

$$g(j, j-1) = \begin{cases} x_j + y_j, dla\ j = 1 \\ |x_j - x_{j-1}| + |y_j - y_{j-1}|, dla\ j \neq 1 \end{cases}$$

3. Rozwiązanie problemu algorytmem ewolucyjnym

3.1. Wstęp teoretyczny

W celu znalezienia dobrego rozwiązania rozważanego problemu zastosowano algorytm ewolucyjny w wydaniu klasycznym, czyli niewykorzystujący wiedzy o rozwiązywanym problemie. Algorytmy ewolucyjne przeszukują przestrzeń rozwiązań w celu znalezienia najlepszych. Działanie tych algorytmów wzorowane jest ewolucją biologiczną, gdzie najlepiej przystosowane osobniki (rozwiązania) utrzymają się w populacji i/lub dokonują między sobą krzyżowania/mutacji. W ten sposób populacja rozwiązań problemu z biegiem iteracji jest coraz bardziej przystosowana, to znaczy rozwiązania są lepsze z punktu widzenia problemu. W przypadku naszego zagadnienia, gdzie przestrzeń rozwiązań jest duża już dla małego rozmiaru problemu (około 25!*15000 rozwiązań dla macierzy wymiaru 5 i 3 wózków) algorytm ewolucyjny wydaje się być rozsądnym wyborem.

3.2. Genotyp i populacja początkowa

Każdy osobnik populacji reprezentuje jakieś rozwiązanie dopuszczalne postawionego problemu oraz posiada zbiór informacji stanowiący genotyp, który jest podstawą do utworzenia fenotypu – ocenianego funkcją oceny. W przypadku rozważanego przez nas problemu genotyp i fenotyp są równoważne, a funkcją oceny jest funkcja celu. Postać genotypu rozwiązania została przedstawiona w punkcie 2.2. i reprezentuje kolejność odwiedzania danych stanowisk przez poszczególne wózki.

Populacja początkowa naszego problemu jest tworzona w sposób całkowicie losowy. Losowy osobnik populacji jest tworzony przez losową permutację liczb z zakresu 1..m (m to liczba „niezerowych" stanowisk) i wstawienie w losowe miejsca tej permutacji n - 1 znaczników nowego wózka (znacznik pierwszego wózka jest zawsze na początku rozwiązania). Optymalny rozmiar populacji jest przedmiotem badania w punkcie *Testy*.

Populacja w ramach działania algorytmu ma stały rozmiar, to znaczy w każdej iteracji nie jest tworzona nowa na bazie poprzedniej, tylko niektóre osobniki są zastępowane nowymi. Takie podejście zmniejsza złożoność obliczeniową problemu.

3.3. Metoda selekcji

Metoda selekcji określa, w jaki sposób zostaną wybrane rozwiązania (osobniki), które wezmą udział w reprodukcji i mutacji. Ważne przy wyborze metody selekcji jest to, aby w gronie rodziców znajdowały się rozwiązania najlepiej przystosowane (z najmniejszą wartością funkcji celu w naszym modelu). W naszym algorytmie zastosowaliśmy do wyboru rodziców **metodę turniejową**. W wybranej losowo w grupie turniejowej osobniki są porównywane względem siebie na podstawie obliczonej i przypisanej do nich wcześniej funkcji celu.

Następnie dwoje najlepszych osobników staje się rodzicami i z użyciem **operatora krzyżowania** tworzone jest dwoje potomków (z uwagi na specyfikę problemu potomek jest zawsze bardziej podobny do jednego z rodziców) i lepszy z nich zastępuje najgorszego osobnika w grupie turniejowej. Zaletą takiego podejścia jest to, że można wybrać rodziców bez żadnych informacji statystycznych opisujących populację. Dzięki wykorzystaniu wartości funkcji celu do porównania nie ma potrzeby wyliczania przy każdym porównaniu funkcji przystosowania, co wpływa pozytywnie na złożoność obliczeniową algorytmu.

Rozmiar turnieju jest stały podczas działania algorytmu i stanowi przedmiot badania w punkcie *Testy*.

3.4. Czas działania algorytmu/liczba iteracji

Aby zakończyć działanie algorytmu i podać najlepsze znalezione rozwiązanie można zaimplementować jakieś wymaganie względem otrzymanego rozwiązania, które jeśli jest spełnione, to algorytm kończy działanie i podaje wynik. W naszym modelu mogłaby to być określona z góry lub estymowana wartość funkcji celu, której przekroczenie skutkowałoby zakończeniem działania. Jednak ze względu na specyfikę problemu i zastosowany algorytm niezwykle trudne byłoby podanie takiej wartości. Ponadto algorytm mógłby nigdy jej nie osiągnąć działać w nieskończoność. Możliwe jest również przerwanie pracy w przypadku, gdy do populacji od jakiegoś czasu nie dodawane są lub dodawane jest bardzo mało zmian. W takiej sytuacji zdecydowaliśmy się jednak podnieść współczynnik mutacji i kontynuować działanie, natomiast liczbę iteracji narzucić z góry.

Wyznaczenie optymalnej liczby iteracji algorytmu jest przedmiotem badań w rozdziale *Testy*.

3.5. Operatory krzyżowania

Rdzeniem algorytmu ewolucyjnego jest krzyżowanie osobników lepiej przystosowanych ze sobą. W tym celu zaimplementowaliśmy funkcję realizującą operację krzyżowania dwóch osobników, która bierze od jednego rodzica wycinek genu i wstawia go do genu drugiego rodzica w tym samym miejscu, z którego został wyjęty. Ze względu na permutacyjny charakter rozwiązania dziecko jest bardziej podobne do jednego z rodziców, dlatego przyjęliśmy, że przeżywa to przystosowane lepiej. Poniższy przykład przedstawia sposób, w jaki tworzony jest nowy osobnik.

[0, 2, 3, 4, 0, 9, 5, 0, 6, 8, 7, 1] x [0, 8, 7, 6, 5, 0, 3, 2, 1, 0, 9, 4] =

[0, 8, 7, 3, 9, 0, 5, 6, 2, 0, 1, 4] – podobny bardziej do rodzica 2

[0, 8, 7, 6, 5, 0, 3, 2, 1, 0, 9, 4] x [0, 2, 3, 4, 0, 9, 5, 0, 6, 8, 7, 1] =

[0, 4, 9, 6, 0, 5, 3, 0, 2, 8, 7, 1] – podobny bardziej do rodzica 2

Realizacja operatora jest zrealizowana w ten sposób, że po usunięciu znaczników wybierany jest fragment genu o długości ustalanej programowo w losowym miejscu i w tym samym miejscu wstawiane jest w drugim genomie. Wartości przed i po tym wstawionym fragmencie ustawiane są z pozostałych rodzica drugiego z zachowaniem ich kolejności. Następnie wstawiane są znaczniki nowego wózka w miejscach, w których były u rodzica 2. W ten sposób potomek zachowuje podobieństwo do rodzica 1 pod względem wybranego wycinka i jego miejsca w genie, natomiast znaczniki nowego wózka i kolejności dziedziczy po rodzicu 2.

Szansa na wywołanie krzyżowania dla wybranych rodziców jest stała przez cały czas działania algorytmu i jest przedmiotem badania w punkcie *Testy*.

3.6. Operatory mutacji

Operatory mutacji są ważnym elementem algorytmu ewolucyjnego. Wprowadzają one losowe, drobne zmiany w genie mutującego osobnika. Zapewniają one różnorodność populacji, a ich wpływ w polepszaniu rozwiązania jest szczególnie duży w końcowych iteracjach, gdzie populacja znajduje się w okolicy minimum lokalnego. W zaimplementowanej przez nas wersji algorytmu osobnik jest zastępowany swoją mutacją tylko wtedy, gdy poprawia ona wynik. Takie podejście nie zwiększa aż tak różnorodności, jednak nie występuje szansa zepsucia dobrego rozwiązania. Zdecydowaliśmy się na takie podejście ze względu na specyfikę zagadnienia i typ zastosowanych przez nas operatorów.

- Zamiana całej kolejki jednego wózka z kolejką innego:

  Zastosowanie tego operatora wydaje się być dobrym rozwiązaniem ze względu na występowanie w modelu współczynników prędkości wózków. Zastąpienie wózka z długą kolejką innym, szybciej pracującym wózkiem może sprawić, że funkcja celu będzie miała mniejszą wartość. Poniżej zaprezentowane jest działanie opisanego operatora:

  [0, **1, 2, 3, 4**, 0, **5, 6, 7**, 0, 8, 9] → [0, **5, 6, 7**, 0, **1, 2, 3, 4**, 0, 8, 9]

- Przesunięcie znacznika nowego wózka w prawo lub w lewo

  Ta mutacja zmienia nieznacznie genotyp, jednak może znacząco wpłynąć na funkcję oceny (funkcje celu). Poniżej jej działanie:

  [0, 1, 2, 3, **4**, **0**, 5, 6, 7, 0, 8, 9] → [0, 1, 2, 3, **0**, **4**, 5, 6, 7, 0, 8, 9]

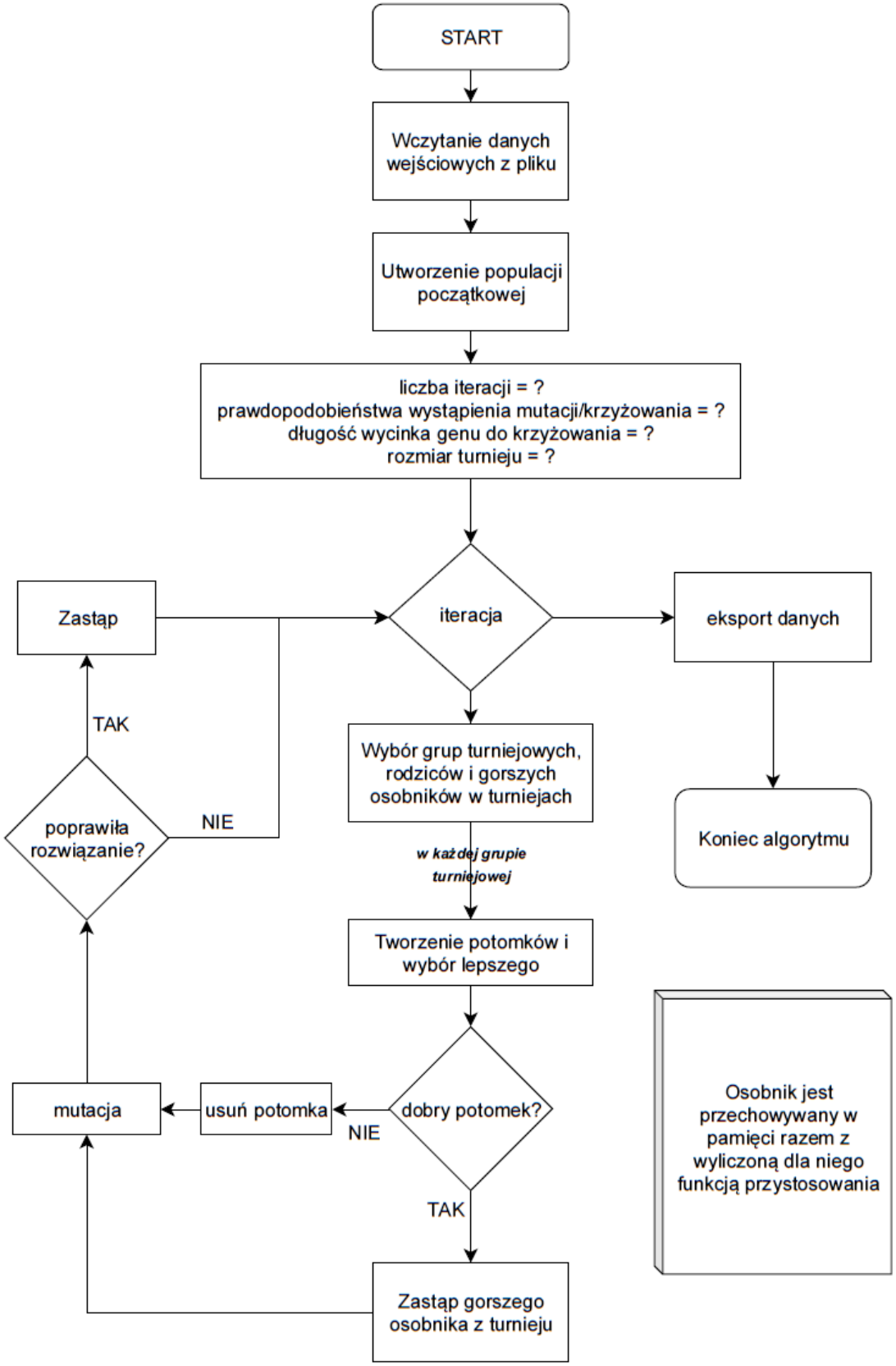
- Losowa permutacja stanowisk w obrębie listy jednego wózka

  [0, **1**, **2**, **3**, **4**, 0, 5, 6, 7, 0, 8, 9] → [0, **3**, **2**, **4**, **1**, 0, 5, 6, 7, 0, 8, 9]

Powyższe operatory nie mogą wystąpić jednocześnie, są wprowadzane tylko wtedy, kiedy poprawią wartość rozwiązania, a ich udział we wszystkich mutacjach, jak i ogólne prawdopodobieństwo wystąpienia mutacji jest obiektem analizy w punkcie *Testy*.

Oprócz zastosowania operatorów w algorytmie, zbadaliśmy również wpływ stopniowego zwiększania z biegiem iteracji szansy ich wystąpienia ponad normę oraz gwałtownego podniesienia szansy na mutację w sytuacji, w której od wielu iteracji różnorodność populacji się nie zmienia. Oba te działania pozytywnie wpłynęły zarówno na szybkość osiągnięcia optymalnego rozwiązania, jak i na jego wartość, dlatego wprowadziliśmy je na stałe do naszego algorytmu.

3.7. Uproszczony schemat blokowy



3.8. Oprogramowanie użyte do realizacji projektu

Algorytm został przez nas napisany w języku **Python**, z interpreterem w wersji **3.6.** Wybraliśmy ten język programowania ze względu na dostępny szeroki zakres bibliotek, łatwość programowania oraz to, że jest całkowicie darmowy. Wadą zastosowania tego języka jest jednak to, że jest on wolniejszy w porównaniu do innych języków wysokiego poziomu, jak np. C++. Nie przeszkodziło to jednak w płynnym przeprowadzeniu testów ze względu na nie tak duże rozmiary problemów oraz dyspozycją wydajnego sprzętu. Implementując algorytm korzystaliśmy głównie z biblioteki ***random*** oraz ***math*** pozwalających wprowadzać elementy losowości i bardziej zaawansowanych obliczeń matematycznych. Pomocna okazała się również biblioteka ***matplotlib*** służąca do rysowania wykresów.

4. Testy

4.1. Sprawdzenie poprawności działania algorytmu

Pierwszym testem, który wykonaliśmy było sprawdzenie dla małej instancji testowej, czy algorytm działa poprawnie, to znaczy czy znalezione rozwiązania są dopuszczalne i wydają się być logiczne. W tym celu utworzyliśmy problem małego rozmiaru, wykonaliśmy na nim algorytm i przedstawiliśmy najlepsze uzyskane rozwiązanie graficznie.

n = 3 – liczba wózków				
3	3	0	3	3
2	5	3	1	0
1	0	3	3	3
0	1	4	0	2
y/x	**0**	**1**	**2**	**3**

Rozmieszczenie stanowisk fabryki z kosztami ich obsługi.

Przegląd zupełny tego problemu wymagałby przeanalizowania około 14! możliwych rozwiązań. Rozwiązaliśmy go naszym algorytmem z rozmiarem populacji 20 i liczbą iteracji 100. Pozostałe parametry, które są mniej istotne, ustawiliśmy na rozsądne wartości. Otrzymaliśmy następujące rozwiązanie, dla którego funkcja celu wynosi 17,8:

Wózek nr 1, α = 1.18				
Wózek nr 2, α = 0.8				
Wózek nr 3, α = 1.08				
3	3_3	0	3_3	3_4
2	5_1	3_2	1_2	0
1	0	3_4	3_1	3_3
0	1_1	4_5	0	2_2
y/x	**0**	**1**	**2**	**3**

Gdzie kolorami są zaznaczone stanowiska obsłużone przez dany wózek w danej kolejności:

1) (0,0) → (2,1) → (2,2) → (2,3) → (3,3)
2) (0,0) → (0,0) → (3,0) → (3,1) → (1,1) → (1,0)
3) (0,0) → (0,2) → (1,2) → (0,3)

Funkcja celu policzona „ręcznie” wynosi:

$$\max(11.8 + 6,\ 10.4 + 7,\ 11.88 + 5) = \max(17.8, 17.4, 16.88) = 17.8$$

i zgadza się ona z obliczoną programowo. Z powyższej tabeli i obliczeń wynika, że rozwiązanie znaleziona algorytmicznie wysyła każdy wózek do innej strefy, przez co minimalizowany jest czas przejazdu. Ponadto kolejka wózka z najmniejszym współczynnikiem α jest najdłuższa, a z największym najkrótsza. Skutkuje to równomiernym czasem pracy każdego wózka, co z kolei zmniejsza wartość funkcji celu. Na podstawie przedstawionej analizy można stwierdzić, że algorytm działa poprawnie i należy jedynie dobrać parametry tak, aby działał najoptymalniej.

W pesymistycznym przypadku (krzyżowania i mutacje następowały zawsze) podczas działania algorytmu przeglądnięte zostało 4000 rozwiązań, co w porównaniu do przeglądu zupełnego daje 4,6*10^-6%.

4.2. Dobór parametrów algorytmu na podstawie testów dla różnych instancji testowych

Ważnym elementem tworzenia algorytmu ewolucyjnego jest odpowiedni dobór parametrów sterujących. W przypadku naszego algorytmu są nimi: rozmiar turnieju, prawdopodobieństwa krzyżowań i mutacji (wraz z udziałem każdej mutacji) oraz długość części genu przenoszonej od jednego rodzica podczas krzyżowania. Oprócz tego, przy okazji wykonywania testów sprawdziliśmy, czy zaimplementowane funkcjonalności, takie jak powolny wzrost prawdopodobieństwa wystąpienia mutacji z biegiem iteracji i gwałtowny jego wzrost przy małym zróżnicowaniu populacji wpływają pozytywnie na działanie algorytmu.

Przygotowaliśmy 8 instancji testowych i dla każdej z nich badaliśmy średnią wartość funkcji celu dla wielu powtórzeń działania algorytmu z tą samą liczbą iteracji, populacją początkową i różnych zestawów parametrów. Rozmiar populacji i liczba iteracji w każdej instancji są stosunkowo małe, aby wyniki dla różnych zestawów parametrów różniły się i było łatwo wyciągnąć wnioski z tego, jaki jest ich wpływ na szybkość znajdowania rozwiązania.

4.2.1. Instancja testowa nr 1

W tej instancji macierz wejściowa była rozmiaru 4x4, fabryka dysponowała 3 wózkami, populacja początkowa wynosiła 50 osobników, a algorytm wykonywał 30 iteracji. Poniżej znajduje się macierz wejściowa wraz ze współczynnikami wózków.

Tabela 1. Warunki początkowe dla 1 instancji testowej.

$\alpha_1 = 1.18, \alpha = 0.8, \alpha = 1.08$				
3	1	4	0	2
2	0	3	3	3
1	5	3	1	0
0	3	0	3	3
y/x	**0**	**1**	**2**	**3**

Tabela 2. Wyniki testów instancji nr 1.

Rozmiar turnieju	Długość wycinka	Prawd. Krzyż.	Prawd. mutacji	Dominujący udział mutacji nr	Powolny wzrost prawd. mutacji	Gwałtowny wzrost prawd. mutacji	Średnia wartość f celu z 30 powtórzeń
2	**3**	**1**	**0.1**	**1**	**4**	**Nie**	**18.641**
2	**3**	**1**	**0.1**	**2**	**4**	**Nie**	**18.641**
2	**3**	**1**	**0.1**	**3**	**4**	**Nie**	**18.641**
3	**losowa**	**0.5**	**0.5**	**-**	**1**	**Tak**	**19.41**
3	3	0.5	0.5	-	1	Nie	19.526
4	4	0.2	0.2	-	3	Nie	21.412
5	**losowa**	**1**	**0.05**	**-**	**10**	**Nie**	**18.947**
8	7	0.5	0.4	1	1	Nie	20.083
8	7	0.5	0.4	2	1	Nie	19.955
8	7	0.5	0.4	3	1	Nie	20.367
10	2	0.2	0.05	-	5	Nie	22.721
12	**4**	**0.75**	**0.2**	**-**	**2**	**Tak**	**19.858**
12	4	0.75	0.2	-	2	Nie	20.34
15	3	0.5	0.3	-	1	Nie	20.597
20	losowa	1	0.5	-	2	Nie	20.396
50	3	1	0.75	-	1	Nie	21.795

Na podstawie tej instancji (a następnymi można to potwierdzić) stwierdziliśmy, że turniej wielkości 2 daje najlepsze rozwiązania, co jest zrozumiałem, albowiem przeszukiwany jest wtedy największy obszar rozwiązań. Z racji złożoności obliczeniowej jaka jest implikowana tak małym turniejem warto również rozpatrzyć parametry, w których turniej był większy i funkcja celu miała nieco większą wartość. Gwałtowny wzrost mutacji poprawiał wynik dla obu zestawów pozostałych parametrów, w których był użyty. Na podstawie zestawów parametrów 8-11 dominujący wpływ mutacji nr 2 wpływa najkorzystniej na wynik.

4.2.2. Instancja testowa nr 2

W tym przypadku macierz wejściowa miała wymiar 5, zakład dysponował 3 wózkami, a rozmiar populacji i liczba iteracji wynosiły odpowiednio 50 i 30.

Tabela 3. Warunki początkowe dla 2 instancji testowej.

α = 0.81, α = 0.81, α = 1.14					
4	2	9	1	4	1
3	7	7	7	10	6
2	3	1	7	0	6
1	6	9	0	7	4
0	3	9	1	5	0
y/x	**0**	**1**	**2**	**3**	**4**

Tabela 4. Wyniki testów instancji nr 2.

Rozmiar turnieju	Długość wycinka	Prawd. Krzyż.	Prawd. mutacji	Dominujący udział mutacji nr	Powolny wzrost prawd. mutacji	Gwałtowny wzrost prawd. mutacji	Średnia wartość f celu z 30 powtórzeń
2	**3**	**1**	**0.1**	**1**	**4**	**Nie**	**52.489**
2	**3**	**1**	**0.1**	**2**	**4**	**Nie**	**52.489**
2	**3**	**1**	**0.1**	**3**	**4**	**Nie**	**52.489**
3	**losowa**	**0.5**	**0.5**	**-**	**1**	**Tak**	**54.175**
3	**3**	**0.5**	**0.5**	**-**	**1**	**Nie**	**54.438**
4	4	0.2	0.2	-	3	Nie	58.06
5	**losowa**	**1**	**0.05**	**-**	**10**	**Nie**	**52.794**
8	7	0.5	0.4	1	1	Nie	55.708
8	7	0.5	0.4	2	1	Nie	55.427
8	**7**	**0.5**	**0.4**	**3**	**1**	**Nie**	**54.716**
10	2	0.2	0.05	-	5	Nie	59.807
12	4	0.75	0.2	-	2	Tak	55.432
12	4	0.75	0.2	-	2	Nie	56.558
15	3	0.5	0.3	-	1	Nie	57.32
20	losowa	1	0.5	-	2	Nie	56.283
50	3	1	0.75	-	1	Nie	58.712

Gwałtowny wzrost poprawia rozwiązanie, powolny wzrost dla prawdopodobieństwa mutacji równego 0.05 też daje bardzo dobry wynik. Na podstawie zestawów parametrów 8-11 najkorzystniej na wynik wpływa dominująca rola mutacji nr 3.

4.2.3. Instancja testowa nr 3

W tym przypadku macierz wejściowa miała wymiar 6, zakład dysponował 4 wózkami, a rozmiar populacji i liczba iteracji wynosiły odpowiednio 50 i 30.

Tabela 5. Warunki początkowe dla 3 instancji testowej.

α = 0.94, α = 1.08, α = 1.11, α = 1.15						
5	2	9	1	4	1	7
4	7	7	10	6	3	1
3	7	0	6	6	9	0
2	7	4	3	9	1	5
1	0	0	0	10	8	0
0	6	10	3	6	0	8
y/x	**0**	**1**	**2**	**3**	**4**	**5**

Tabela 6. Wyniki testów instancji nr 3.

Rozmiar turnieju	Długość wycinka	Prawd. Krzyż.	Prawd. mutacji	Dominujący udział mutacji nr	Powolny wzrost prawd. mutacji	Gwałtowny wzrost prawd. mutacji	Średnia wartość f celu z 30 powtórzeń
2	**3**	**1**	**0.1**	**1**	**4**	**Nie**	**72.984**
2	**3**	**1**	**0.1**	**2**	**4**	**Nie**	**72.984**
2	**3**	**1**	**0.1**	**3**	**4**	**Nie**	**72.984**
3	**losowa**	**0.5**	**0.5**	**-**	**1**	**Tak**	**72.836**
3	3	0.5	0.5	-	1	Nie	76.756
4	4	0.2	0.2	-	3	Nie	84.923
5	**losowa**	**1**	**0.05**	**-**	**10**	**Nie**	**73.54**
8	7	0.5	0.4	1	1	Nie	78.838
8	7	0.5	0.4	2	1	Nie	78.13
8	**7**	**0.5**	**0.4**	**3**	**1**	**Nie**	**76.59**
10	2	0.2	0.05	-	5	Nie	94.596
12	4	0.75	0.2	-	2	Tak	78.457
12	4	0.75	0.2	-	2	Nie	79.429
15	3	0.5	0.3	-	1	Nie	81.935
20	losowa	1	0.5	-	2	Nie	77.599
50	3	1	0.75	-	1	Nie	84.514

Po raz kolejny bardzo dobry wynik daje zestaw parametrów z turniejem wielkości 5. Gwałtowny wzrost poprawia rozwiązanie, a najlepszy wpływ ma mutacja nr 3.

4.2.4. Instancja testowa nr 4

W tym przypadku macierz wejściowa miała wymiar 7, zakład dysponował 4 wózkami, a rozmiar populacji i liczba iteracji wynosiły odpowiednio 75 i 30.

Tabela 7. Warunki początkowe dla 4 instancji testowej.

α = 0.86, α = 0.91, α = 0.98, α = 0.87							
6	4	18	2	8	3	15	4
5	15	20	12	6	3	15	15
4	12	13	19	0	14	8	12
3	18	3	10	0	0	0	18
2	17	0	12	6	13	0	17
1	7	14	15	17	7	11	7
0	7	14	9	0	13	17	7
y/x	**0**	**1**	**2**	**3**	**4**	**5**	**6**

Tabela 8. Wyniki testów instancji nr 4.

Rozmiar turnieju	Długość wycinka	Prawd. Krzyż.	Prawd. mutacji	Dominujący udział mutacji nr	Powolny wzrost prawd. mutacji	Gwałtowny wzrost prawd. mutacji	Średnia wartość f celu z 30 powtórzeń
2	**3**	**1**	**0.1**	**1**	**4**	**Nie**	**149.636**
2	**3**	**1**	**0.1**	**2**	**4**	**Nie**	**149.636**
2	**3**	**1**	**0.1**	**3**	**4**	**Nie**	**149.636**
3	**losowa**	**0.5**	**0.5**	**-**	**1**	**Tak**	**153.951**
3	**3**	**0.5**	**0.5**	**-**	**1**	**Nie**	**154.756**
4	4	0.2	0.2	-	3	Nie	163.978
5	**losowa**	**1**	**0.05**	**-**	**10**	**Nie**	**150.35**
8	7	0.5	0.4	1	1	Nie	155.439
8	7	0.5	0.4	2	1	Nie	156.082
8	7	0.5	0.4	3	1	Nie	156.422
10	2	0.2	0.05	-	5	Nie	168.297
12	4	0.75	0.2	-	2	Tak	157.262
12	4	0.75	0.2	-	2	Nie	159.178
15	3	0.5	0.3	-	1	Nie	163.139
20	**losowa**	**1**	**0.5**	**-**	**2**	**Nie**	**154.151**
50	3	1	0.75	-	1	Nie	163.709

Po raz kolejny bardzo dobry wynik daje zestaw parametrów z turniejem wielkości 5. Gwałtowny wzrost poprawia rozwiązanie, a najlepszy wpływ ma mutacja nr 1. Duży turniej z losową długością wycinka daje bardzo dobre rozwiązanie biorąc pod uwagę niską złożoność obliczeniową tej opcji.

4.2.5. Instancja testowa nr 5

W tym przypadku macierz wejściowa miała wymiar 8, zakład dysponował 5 wózkami, a rozmiar populacji i liczba iteracji wynosiły odpowiednio 75 i 30.

Tabela 9. Warunki początkowe dla 5 instancji testowej.

α = 1.05, α = 1.17, α = 0.82, α = 1.1, α = 0.95								
7	4	18	2	8	3	15	4	18
6	20	12	6	3	15	0	20	12
5	19	0	14	8	7	18	19	0
4	0	0	0	20	17	0	0	0
3	13	0	16	7	14	15	13	0
2	11	7	7	14	9	0	11	7
1	20	3	5	20	9	3	20	3
0	13	16	6	9	9	18	13	16
y/x	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**

Tabela 10. Wyniki testów instancji nr 5.

Rozmiar turnieju	Długość wycinka	Prawd. Krzyż.	Prawd. mutacji	Dominujący udział mutacji nr	Powolny wzrost prawd. mutacji	Gwałtowny wzrost prawd. mutacji	Średnia wartość f celu z 30 powtórzeń
2	**3**	**1**	**0.1**	**1**	**4**	**Nie**	**204.7**
2	**3**	**1**	**0.1**	**2**	**4**	**Nie**	**204.7**
2	**3**	**1**	**0.1**	**3**	**4**	**Nie**	**204.7**
3	**losowa**	**0.5**	**0.5**	**-**	**1**	**Tak**	**210.744**
3	3	0.5	0.5	-	1	Nie	221.8
4	4	0.2	0.2	-	3	Nie	242.495
5	**losowa**	**1**	**0.05**	**-**	**10**	**Nie**	**205.534**
8	7	0.5	0.4	1	1	Nie	224.25
8	7	0.5	0.4	2	1	Nie	225.508
8	7	0.5	0.4	3	1	Nie	223.619
10	2	0.2	0.05	-	5	Nie	251.92
12	4	0.75	0.2	-	2	Tak	226.471
12	4	0.75	0.2	-	2	Nie	230.427
15	3	0.5	0.3	-	1	Nie	238.701
20	**losowa**	**1**	**0.5**	**-**	**2**	**Nie**	**214.34**
50	3	1	0.75	-	1	Nie	234.726

Zestaw z rozmiarem turnieju 5 ponownie zadziałał bardzo dobrze. Najlepszy wpływ ma mutacja nr 3, gwałtowny wzrost poprawia rozwiązanie, a dla dużego rozmiaru turnieju ponownie sprawdziła się losowa długość wycinka, prawdopodobieństwo krzyżowania równe 1 i mutacji równe 0.5. Prawdopodobnie ma to związek z długością rozwiązania która w tej instancji testowej jest duża i dziedziczenie po rodzicu wycinka genu długości 3 lub 4 ma małą skuteczność.

4.2.6. Instancja testowa nr 6

W tym przypadku macierz wejściowa miała wymiar 12, zakład dysponował 7 wózkami, a rozmiar populacji i liczba iteracji wynosiły odpowiednio 150 i 50.

Tabela 11. Warunki początkowe dla 6 instancji testowej.

α = 1.19, α = 1.17, α = 1.17, α = 1.05, α = 0.9, α = 0.9, α = 1.12												
11	4	18	27	25	24	2	4	18	27	25	24	2
10	20	12	25	6	3	15	20	12	25	6	3	15
9	24	24	0	22	14	8	24	24	0	22	14	8
8	28	10	0	0	0	20	28	10	0	0	0	20
7	6	13	23	0	16	7	6	13	23	0	16	7
6	11	7	21	7	24	14	11	7	21	7	24	14
5	29	17	29	20	3	5	29	17	29	20	3	5
4	10	28	23	22	16	29	10	28	23	22	16	29
3	6	9	9	18	28	15	6	9	9	18	28	15
2	1	15	7	23	25	12	1	15	7	23	25	12
1	22	24	21	23	11	2	22	24	21	23	11	2
0	16	26	12	11	15	23	16	26	12	11	15	23
y/x	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**

Tabela 12. Wyniki testów instancji nr 6.

Rozmiar turnieju	Długość wycinka	Prawd. Krzyż.	Prawd. mutacji	Dominujący udział mutacji nr	Powolny wzrost prawd. mutacji	Gwałtowny wzrost prawd. mutacji	Średnia wartość f celu z 30 powtórzeń
2	**3**	**1**	**0.1**	**1**	**4**	**Nie**	**584.163**
2	**3**	**1**	**0.1**	**2**	**4**	**Nie**	**584.163**
2	**3**	**1**	**0.1**	**3**	**4**	**Nie**	**584.163**
3	**losowa**	**0.5**	**0.5**	**-**	**1**	**Tak**	**622.089**
3	3	0.5	0.5	-	1	Nie	644.313
4	4	0.2	0.2	-	3	Nie	675.72
5	**losowa**	**1**	**0.05**	**-**	**10**	**Nie**	**584.148**
8	7	0.5	0.4	1	1	Nie	638.755
8	7	0.5	0.4	2	1	Nie	637.638
8	7	0.5	0.4	3	1	Nie	636.94
10	2	0.2	0.05	-	5	Nie	690.463
12	4	0.75	0.2	-	2	Tak	653.857
12	4	0.75	0.2	-	2	Nie	662.795
15	3	0.5	0.3	-	1	Nie	673.272
20	**losowa**	**1**	**0.5**	**-**	**2**	**Nie**	**611.612**
50	3	1	0.75	-	1	Nie	669.872

Można zauważyć, że dla dużych problemów najkorzystniejsza jest losowa długość wycinka genu i prawdopodobieństwo krzyżowania równe 1. W tej instancji najlepiej wypadła mutacja nr 3, a gwałtowny wzrost prawdopodobieństwa mutacji korzystnie wpływał na wynik.

4.2.7. Instancja testowa nr 7

W tym przypadku macierz wejściowa miała wymiar 20, zakład dysponował 10 wózkami, a rozmiar populacji i liczba iteracji wynosiły odpowiednio 300 i 50.

Tabela 13. Warunki początkowe dla 7 instancji testowej.

α = 1.14, α = 1.19, α = 1.17, α = 1.18, α = 0.95, α = 0.94, α = 0.81, α = 0.95, α = 1.05																				
19	2	9	1	4	1	7	7	7	6	3	1	7	0	6	6	9	0	7	4	3
18	9	1	5	0	0	0	8	0	6	3	6	0	8	3	7	7	8	3	5	3
17	3	7	4	0	6	8	1	2	4	1	5	8	6	8	3	4	4	9	7	8
16	6	9	0	7	3	6	6	2	5	8	5	1	7	8	1	2	8	6	5	7
15	0	7	0	4	9	9	9	6	2	2	8	3	0	3	8	8	3	6	8	5
14	9	5	7	4	8	9	0	6	8	2	8	8	3	6	0	7	5	9	8	3
13	8	6	7	5	6	5	0	8	8	9	9	5	7	9	0	3	2	8	9	2
12	1	8	4	0	1	1	0	7	0	4	3	4	1	9	2	5	4	1	2	2
11	4	8	2	4	4	7	5	7	7	1	0	4	6	5	6	3	4	1	4	8
10	3	9	6	0	3	0	6	2	0	2	7	8	6	8	3	8	7	3	8	0
9	6	9	5	6	0	4	2	3	0	4	1	1	4	4	2	6	9	4	2	0
8	8	0	9	3	9	7	2	9	8	0	6	3	5	1	3	9	6	9	3	7
7	1	6	4	8	7	0	5	9	6	4	0	2	3	5	9	2	5	6	3	4
6	1	6	8	5	8	7	8	3	1	0	1	2	2	2	8	3	4	5	9	8
5	4	5	5	5	1	4	3	9	7	2	9	8	1	5	0	6	1	6	2	2
4	5	1	9	9	6	1	9	8	3	9	1	4	5	4	9	8	1	7	4	1
3	0	4	0	9	0	1	6	1	0	3	3	9	6	2	1	7	2	3	2	1
2	6	6	8	4	8	4	7	5	1	3	5	0	0	0	4	9	5	7	6	5
1	6	1	1	5	9	7	1	4	3	9	8	7	5	4	2	8	3	4	3	3
0	5	1	4	1	7	1	9	5	3	6	4	0	5	2	5	9	4	3	5	1
y/x	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**

Tabela 14. Wyniki testów instancji nr 7.

Rozmiar turnieju	Długość wycinka	Prawd. Krzyż.	Prawd. mutacji	Dominujący udział mutacji nr	Powolny wzrost prawd. mutacji	Gwałtowny wzrost prawd. mutacji	Średnia wartość f celu z 30 powtórzeń
2	**3**	**1**	**0.1**	**1**	**4**	**Nie**	**857.666**
2	**3**	**1**	**0.1**	**2**	**4**	**Nie**	**857.666**
2	**3**	**1**	**0.1**	**3**	**4**	**Nie**	**857.666**
3	**losowa**	**0.5**	**0.5**	**-**	**1**	**Tak**	**905.771**
3	3	0.5	0.5	-	1	Nie	915.237
4	4	0.2	0.2	-	3	Nie	970.5
5	**losowa**	**1**	**0.05**	**-**	**10**	**Nie**	**863.009**
8	7	0.5	0.4	1	1	Nie	919.985
8	7	0.5	0.4	2	1	Nie	928.588
8	7	0.5	0.4	3	1	Nie	934.781
10	2	0.2	0.05	-	5	Nie	994.223
12	**4**	**0.75**	**0.2**	**-**	**2**	**Tak**	**912.836**
12	4	0.75	0.2	-	2	Nie	923.701
15	3	0.5	0.3	-	1	Nie	939.675
20	**losowa**	**1**	**0.5**	**-**	**2**	**Nie**	**878.429**
50	3	1	0.75	-	1	Nie	927.402

Ponownie wygrały losowe długości wycinka, gwałtowny wzrost polepszał rozwiązanie, natomiast najlepiej na f celu wpłynęła mutacja nr 1.

4.2.8. Instancja testowa nr 8

W tej instancji testowej macierz była wymiaru 50x50, zakład dysponował 50 wózkami, populacja i liczba iteracji wynosiły 200 i 15. Średnia została wyciągnięta z 5 powtórzeń algorytmu.

Tabela 13. Wyniki testów instancji nr 8.

Rozmiar turnieju	Długość wycinka	Prawd. Krzyż.	Prawd. mutacji	Dominujący udział mutacji nr	Powolny wzrost prawd. mutacji	Gwałtowny wzrost prawd. mutacji	Średnia wartość f celu z 30 powtórzeń
2	**3**	**1**	**0.1**	**1**	**4**	**Nie**	**4071.868**
2	**3**	**1**	**0.1**	**2**	**4**	**Nie**	**4071.868**
2	**3**	**1**	**0.1**	**3**	**4**	**Nie**	**4071.868**
3	**losowa**	**0.5**	**0.5**	**-**	**1**	**Tak**	**4256.194**
3	3	0.5	0.5	-	1	Nie	4264.008
4	4	0.2	0.2	-	3	Nie	4365.35
5	**losowa**	**1**	**0.05**	**-**	**10**	**Nie**	**4123.114**
8	7	0.5	0.4	1	1	Nie	4273.94
8	7	0.5	0.4	2	1	Nie	4267.546
8	7	0.5	0.4	3	1	Nie	4281.04
10	2	0.2	0.05	-	5	Nie	4372.57
12	**4**	**0.75**	**0.2**	**-**	**2**	**Tak**	**4194.494**
12	4	0.75	0.2	-	2	Nie	4265.898
15	3	0.5	0.3	-	1	Nie	4305.282
20	**losowa**	**1**	**0.5**	**-**	**2**	**Nie**	**4119.506**
50	3	1	0.75	-	1	Nie	4243.618

W tej instancji również można zaobserwować pozytywny wpływ gwałtownego wzrostu prawdopodobieństwa mutacji oraz losowej długości wycinka genu. Najlepiej sprawdziła się mutacja nr 2.

4.3. Ogólne wnioski z przeprowadzonych testów

- W ogólnym zastosowaniu najlepiej sprawdza się losowa długość wycinka genu branego od jednego rodzica do drugiego podczas krzyżowania. Znając wymiar rozważanego problemu należałoby zbadać dla niego odpowiednią długość.
- W większości przypadków testowych osiągnięto najlepsze wyniki dla turnieju rozmiaru 2. Wynika to ze znacznie większego zakresu przeanalizowanych rozwiązań w porównaniu do pozostałych. Biorąc pod uwagę złożoność obliczeniową i osiągane wyniki, najrozsądniejsze jest wybranie rozmiaru turnieju 5, który nie jest tak obciążający i daje bardzo zadowalające wyniki.
- Najlepsze wyniki daje algorytm wywołany z prawdopodobieństwem krzyżowania równym 1 i prawdopodobieństwem mutacji równym 0.05, przy czym prawdopodobieństwo to rośnie do 0.5 w późniejszych iteracjach. W takiej sytuacji algorytm na początku skupia się bardziej na krzyżowaniu między sobą dobrych

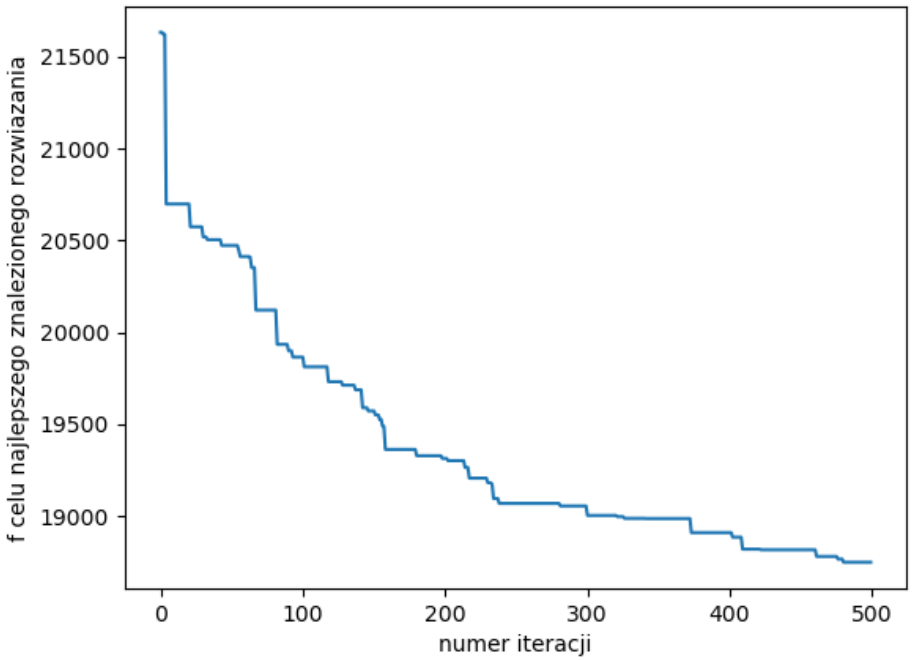
rozwiązań, a dopiero pod koniec działania udoskonala je wprowadzając mutacje. W każdym badanym przypadku testowym dwukrotne podniesienie prawdopodobieństwa mutacji w sytuacji, kiedy populacja jest mało zróżnicowana (i przywróceniu go do pierwotnego stanu kiedy wróci do normy) się opłaciło i otrzymano lepsze wyniki.

- Badanie, których mutacji lepiej używać częściej, a których rzadziej, nie dało jednoznacznej odpowiedzi, dlatego każdej mutacji będziemy używać w algorytmie w takim samym stopniu.
- Dla małych problemów najkorzystniej jest wybrać rozmiar turnieju 2, sztywno określoną długość wycinka genu do krzyżowania, prawdopodobieństwo krzyżowania równe 1 i mutacji równe 0.1 (powoli rosnące). Jednak dla dużych problemów najrozsądniejsze wydaje się być ustawienie rozmiaru turnieju na 5, 20 lub nawet większą wartość, ale z losową długością genu do krzyżowania i prawdopodobieństwem krzyżowania 1 i mutacji 0.5.
- Najbardziej optymalne parametry w zależności od wielkości problemu znajdują się w zestawieniu poniżej.

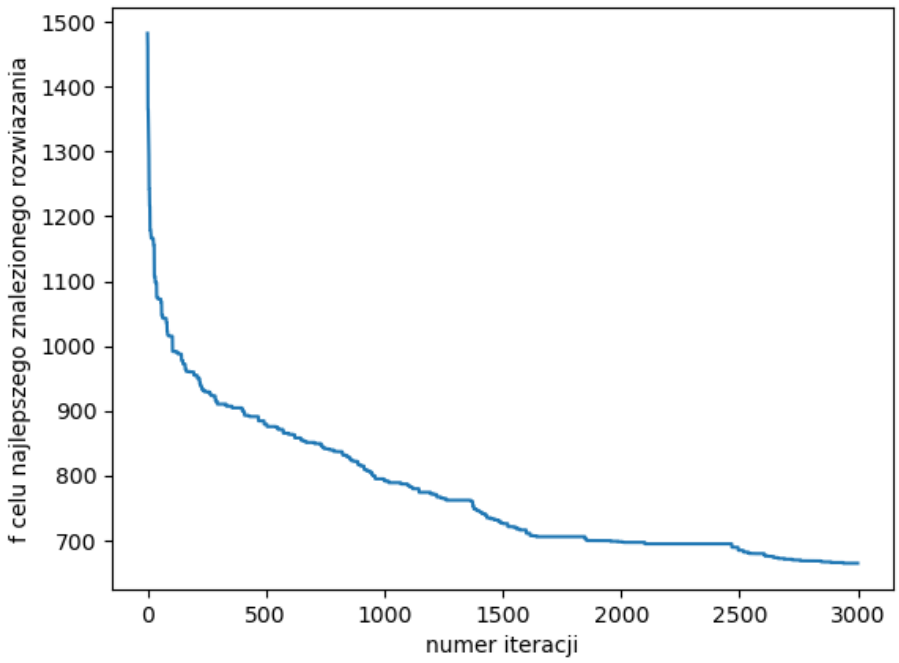
Rozmiar problemu	Rozmiar turnieju	Długość wycinka	Prawd. Krzyż.	Prawd. mutacji	Udział poszczególnych typów mutacji	Powolny wzrost prawd. mutacji	Gwałtowny wzrost prawd. mutacji
Mały (2x2 – 6x6)	2	3	1	0.1	równy	Do 4-krotnie większej	włączony
	5	Losowa	1	0.05	równy	Do 10-krotnie większej	włączony
Średni (7x7 – 15x15)	3	Losowa	1	0.5	równy	wyłączony	włączony
	5	Losowa	1	0.05	równy	Do 10-krotnie większej	włączony
Duży (>15x15)	12	Losowa	1	0.2	równy	Do 2-krotnie większej	włączony
	20	Losowa	1	0.5	równy	Do 2-krotnie większej	włączony

5. Rozwiązanie przykładowego problemu

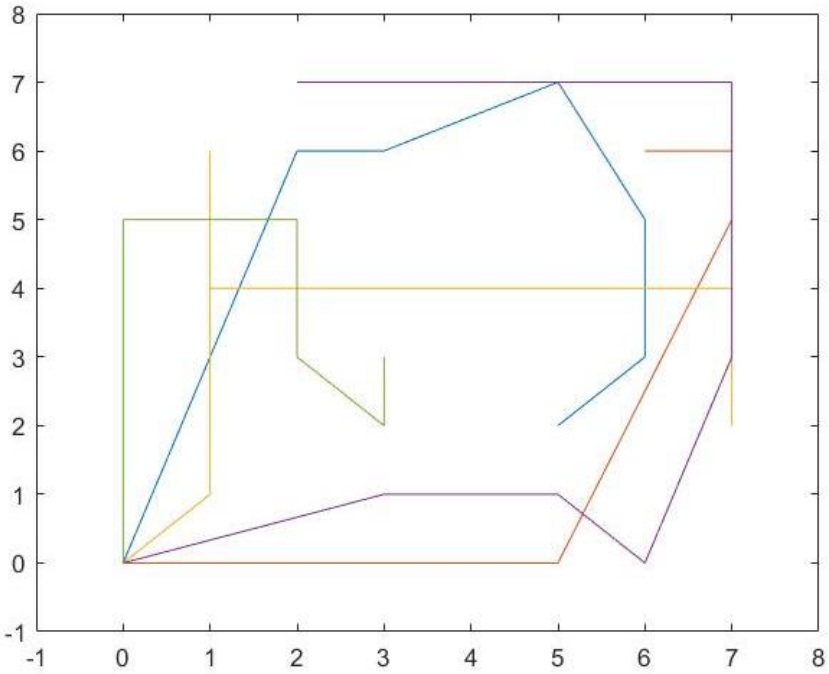
W tym przykładzie rozwiązany zostanie problem, którego wymiar wynosi 100x100, jednak macierz wejściowa będzie miała około połowę stanowisk niewymagających obsługi. Dostępnych w zakładzie wózków będzie 50, a liczba iteracji i populacja początkowa będzie ustawiona na wartości pozwalające przeszukać możliwie dużą przestrzeń rozwiązań w rozsądnym czasie. Poniższy wykres przedstawia przebieg najlepszego znalezionego rozwiązania od iteracji dla populacji liczącej 200 osobników.



W kolejnym przykładzie problemem jest macierz wejściowa 30x30, 30 wózków, a algorytm rozwiązuje go w 3000 iteracji z populacją początkową równą 1000.



Poniżej znajduje się również wizualizacja tras poszczególnych wózków dla nieco mniejszego problemu (macierz 8x8).



Wizualizacja tras wózków dla przykładowego problemu (oś pozioma – x, oś pionowa – y)

Na powyższym wykresie widać, że wózki raczej nie wracają się do wcześniej odwiedzonych stref, mają mniej więcej równe kolejki i (tego już bezpośrednio z wykresu nie widać) obejmują wszystkie wymagane punkty.

6. Wnioski

- Algorytm działa poprawnie i w sposób typowy dla algorytmów ewolucyjnych przeszukuje przestrzeń rozwiązań w celu znalezienia optymalnego rozwiązania.
- Zazwyczaj po osiągnięciu pewnej iteracji (zależnej od problemu) funkcja celu nie zmienia się lub zmienia się nieznacznie.
- Po przeanalizowaniu otrzymanego rozwiązania dla małego problemu można ocenić, że znalezione rozwiązanie wydaje się logiczne (najlepsze wózki otrzymują najbardziej obciążające kolejki i wózki nie wracają się w miejsce, w którym były już wcześniej).
- Po przeanalizowaniu operatorów mutacji różnego typu ich wpływu w różnych fazach działania algorytmu można stwierdzić, że najbardziej przyczyniają się one do poprawy rozwiązania w końcowych iteracjach.
- Najbardziej obciążające obliczeniowo było zmniejszenie rozmiaru turnieju, a co za tym idzie przeszukanie większej liczby rozwiązań, czyli obliczanie funkcji celu i krzyżowania. Jak się jednak okazało, nie zawsze mniejszy turniej dawał lepsze rozwiązania, dla większych problemów odpowiedni był rozmiar 5 i 20.
- Dla bardzo dużych problemów algorytm poprawia funkcję celu i wpada w jakieś minimum, jednak ten typu problemu (problem komiwojażera z kilkoma podróżnikami i paroma dodatkowymi założeniami) jest trudny do rozważenia i aby móc otrzymywać jeszcze lepsze wyniki warto by było utworzyć rozwiązania w jakiś inny sposób, niż losowy, np. pogrupować stanowiska znajdujące się blisko siebie strefami w liczbie odpowiadającej liczbie wózków i na tej podstawie utworzyć populację początkową.