

**WYDZIAŁ ZARZĄDZANIA**

**Sprawozdanie**

**Szeregowanie zadań w permutacyjnym systemie przepływowym**

Autorzy:

Kamil Wcislo, Patryk Zapała, Maciej Ziętal

1. **Solver**

|  |  |  |  |
| --- | --- | --- | --- |
| **Dane** | 50x10 | 100x20 | 200x20 |
| **Najlepszy wynik** | 3004 | 6660 | 12675 |

W przypadku Solvera najlepsze otrzymane wyniki to: 3004 dla danych 50x10, 6660 dla danych 100x20 oraz 12675 dla danych 12675. Rezultaty te zostały wykorzystane jako punkty odniesienia dla napisanych algorytmów.

1. **NEH**

|  |  |  |  |
| --- | --- | --- | --- |
| **Dane** | 50x10 | 100x20 | 200x20 |
| **Najlepszy wynik** | 3445 | 7042 | 13578 |

Algorytm NEH w przypadku każdego uruchomienia, na tych samych danych wejściowych, osiąga taki sam rezultat. Jest to spowodowane faktem, że algorytm ten nie posiada elementu losowości. Dodatkowo wynik algorytmu nie jest zależny od parametrów wejściowych, ponieważ funkcja, oprócz zbioru danych, ich nie posiada. Otrzymane wyniki to kolejno 3445 dla najmniejszego, 7042 dla średniego oraz 13578 dla największego zbioru danych.

Otrzymane wyniki są w znaczącym stopniu gorsze niż te uzyskane dzięki Solverowi. Jest to prawdopodobnie spowodowane tym, że NEH nie wybiera kolejnych zadań na podstawie łącznego czasu wykonania wszystkich zadań, tylko na podstawie ustalonej z góry kolejności. Przez to algorytm wybiera jedynie optymalne miejsce dla konkretnego zadania w danym uszeregowaniu, a nie w całym procesie.

1. **Iteracyjna wspinaczka**

Algorytm wspinaczki posiada jeden parametr n – który opisuje jak duże ma być sąsiedztwo, w którym poszukujemy sąsiednich rozwiązań. W poniższym badaniu, sprawdzono jaki wpływ ma wielkość sąsiedztwa na wynik. Do wykonania algorytmu wykorzystano funkcję multiStart, porównano również jak prezentują się wyniki dla różnej liczby startów. Algorytm ma na celu ustawić zadania w takiej kolejności, aby ich czas wykonania był jak najkrótszy.

* 1. Wpływ wielkości sąsiedztwa na uzyskane wyniki.

Przy tych badaniach liczba startów jest równa i wynosi 3, natomiast zmienia się parametr n.

Dla danych z 50 zadaniami:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n | 3 | 10 | 25 | 50 |
| najlepszy wynik | 3516 | 3485 | 3392 | **3065** |

Dla danych ze 100 zadaniami:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| n | 3 | 10 | 25 | 50 | 75 | 100 |
| najlepszy wynik | 7598 | 7784 | 7347 | 7398 | 7039 | **6578** |

Dla danych z 200 zadaniami:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| n | 3 | 10 | 25 | 50 | 75 |
| najlepszy wynik | 13342 | 13603 | 13588 | 13587 | **12933** |

Na podstawie powyższych tabeli można stwierdzić, że wraz ze wzrostem wielkości sąsiedztwa program uzyskuje coraz lepsze wyniki. Najlepsze wyniki program uzyskał dla największego parametru n, wynoszącego kolejno 50, 100 oraz 75. Nie zawsze jednak większa liczba sąsiednich rozwiązań oznaczała lepszy wynik. Przykładowo dla przeprowadzonych badań z największym zestawem zadań, czas wykonywania zadań dla n=3, był krótszy niż dla n=50, może być to związane z elementem losowości, który występuje w algorytmie. Pierwsze rozwiązanie, od którego zależą kolejne kroki jest wybierane losowo. Ma to wpływ na ostateczny wynik oraz szybkość wykonania się programu.

* 1. Wpływ liczby startów na uzyskane wyniki.

Przy tych badaniach parametr dotyczący sąsiednich rozwiązań był równy i wynosił 3. Zmieniała się natomiast liczba startów, która wynosiła kolejno 1, 3, 5, 10. Dodatkowo każdy program został uruchomiony 3 razy.

Dla danych z 50 zadaniami:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | start=1 | start=3 | start=5 | start=10 |
| 1. | 3554 | 3635 | 3614 | 3539 |
| 2. | 3837 | **3453** | 3516 | 3549 |
| 3. | 3824 | 3674 | 3579 | 3551 |
| Średnia | 3783 | 3587 | 3570 | 3546 |
| Od.st. | 130 | 96 | 41 | 5 |

Dla danych ze 100 zadaniami:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | start=1 | start=3 | start=5 | start=10 |
| 1. | 7934 | 7598 | 7606 | 7576 |
| 2. | 7806 | 7608 | 7589 | 7555 |
| 3. | 7839 | 7708 | 7662 | **7534** |
| Średnia | 7860 | 7638 | 7619 | 7555 |
| Od.st. | 54 | 50 | 31 | 17 |

Dla danych z 200 zadaniami:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | start=1 | start=3 | start=5 | start=10 |
| 1. | 13621 | 13646 | 13656 | 13400 |
| 2. | 13695 | 13606 | 13599 | 13581 |
| 3. | 13487 | **13342** | 13503 | 13540 |
| Średnia | 13601 | 13531 | 13586 | 13507 |
| Od.st. | 86 | 135 | 63 | 77 |

W przypadku zestawu danych z 50 zadaniami, najlepszy wynik uzyskano dla startu wynoszącego 3, tak samo jak dla danych z 200 zadaniami, dla danych ze 100 zadaniami najlepiej poradził sobie program z 10 startami.

Dla danych małych i średnich możemy zaobserwować, że średnia oraz odchylenie standardowe maleje wraz ze wzrostem liczby startów. Im większa liczba startów, tym większe szanse na to, że pierwsze rozwiązanie, które jest losowane w algorytmie będzie miało krótszy czas i doprowadzi nas do lepszego wyniku. Dla dużego zestawu danych zależności te nie są już tak bardzo widoczne, jednak średnia wyników, pokazuje, że najlepiej z zadaniem radzi sobie program, który ma tych startów najwięcej.

* 1. Wnioski

Analiza wyników algorytmu wspinaczki pokazuje, że im większe sąsiedztwo oraz im większa liczba startów, tym wynik jest lepszy. Najlepsze wyniki we wszystkich przypadkach uzyskano dla programów, w których wielkość sąsiedztwa była największa (dla zestawów z 50, 100 oraz 200 zadaniami, kolejno: n=50, n=100, n=75), a liczba startów wynosiła 3.

1. **Tabu Search**

Algorytm Tabu Search przyjmuje 4 argumenty wejściowe – zbiór danych, liczbę następnych zadań, które należą do sąsiedztwa (N), długość listy tabu (S), liczba iteracji (I). Podczas badań sprawdzono wpływ wszystkich argumentów na wyniki działania algorytmu.

* 1. Wpływ wielkości sąsiedztwa na uzyskane wyniki.

Dane 50x10:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| N | 10 | 20 | 30 | 40 | 50 |
| S | 5 | 5 | 5 | 5 | 5 |
| I | 30 | 30 | 30 | 30 | 30 |
| Wynik | 3608 | 3542 | 3375 | 3264 | 2985 |

W powyższej tabeli pokazano wyniki działania algorytmu przy stałej długości listy tabu – 5 oraz stałej liczbie iteracji – 30. Wielkość sąsiedztwa zmieniała się natomiast od 10 do 50, co 10. Na tej podstawie stwierdzono, że im większe sąsiedztwo tym lepsze rezultaty można uzyskać. Przy małym sąsiedztwie otrzymywano bardzo słabe wyniki. Jednak wielkość sąsiedztwa w bardzo dużym stopniu wpływa także na czas wykonania algorytmu.

Dane 100x20:

|  |  |  |  |
| --- | --- | --- | --- |
| N | 10 | 50 | 80 |
| S | 5 | 5 | 5 |
| I | 20 | 20 | 20 |
| Wynik | 7445 | 7255 | 6864 |

Dane 200x20:

|  |  |  |  |
| --- | --- | --- | --- |
| N | 20 | 50 | 80 |
| S | 3 | 3 | 3 |
| I | 20 | 20 | 20 |
| Wynik | 13465 | 13286 | 12946 |

Identyczne wnioski można także wysnuć na podstawie dwóch powyższych tabel. Podobnie jak w przypadku pierwszej tabeli, zaprezentowano wpływ zmian w wielkości sąsiedztwa, przy stałej długości listy tabu oraz liczbie iteracji na wyniki algorytmu. We wszystkich 3 przypadkach wraz z zwiększeniem sąsiedztwa malał całkowity czas wykonania zadań.

* 1. Wpływ wielkości listy tabu na uzyskane wyniki.

Dane 50x10:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N | 40 | 40 | 40 | 40 |
| S | 1 | 3 | 5 | 10 |
| I | 30 | 30 | 30 | 30 |
| Wynik | 3254 | 3165 | 3264 | 3326 |

Dane 100x20:

|  |  |  |  |
| --- | --- | --- | --- |
| N | 80 | 80 | 80 |
| S | 3 | 5 | 10 |
| I | 20 | 20 | 20 |
| Wynik | 6972 | 6864 | 7148 |

Dane 200x20:

|  |  |  |  |
| --- | --- | --- | --- |
| N | 20 | 20 | 20 |
| S | 1 | 3 | 10 |
| I | 20 | 20 | 20 |
| Wynik | 13461 | 13465 | 13344 |

W powyższych 3 tabelach zaprezentowano minimalne wyniki dla wybranych danych wejściowych, przy stałej wielkości sąsiedztwa i liczbie iteracji oraz zmieniającej się długości listy tabu. W przypadku danych 50x10 badane długości listy tabu to 1, 3, 5, 10. Najlepszy wynik – 3165 otrzymano dla listy tabu wynoszącej 3. Zbliżone wyniki przy długościach 1 oraz 5, a najsłabszy przy długości 10. Na tej podstawie można stwierdzić, że długość listy tabu nie może być zbyt duża, ponieważ wtedy pozostają na niej zadania, które mogłyby poprawić wynik, ale także nie może być zbyt krótka, ponieważ wtedy algorytm może ciągle wykonywać te same zadania, które nie poprawiają rezultatu. Podobne wyniki otrzymano przy danych średniej wielkości. Spośród długości 3, 5 i 10 najlepszy wynik odnotowano przy długości 5, najsłabszy natomiast gdy lista tabu była najdłuższa, nieproporcjonalnie długa w stosunku do liczby iteracji. Natomiast w przypadku danych 200x20 ciężko dopatrzeć się konkretnej zależności, ponieważ w każdym przypadku otrzymano bardzo zbliżone rezultaty. Prawdopodobnie jest to spowodowane faktem, że w zaprezentowanym przykładzie wielkość sąsiedztwa jest bardzo mała w stosunku do łącznej liczby danych, więc zdecydowanie większy wpływ na wynik mógł mieć element losowości algorytmu, który wprowadził wynik do minimum lokalnego.

* 1. Wpływ liczby iteracji na uzyskane wyniki.

Dane 50x10:

|  |  |  |  |
| --- | --- | --- | --- |
| N | 30 | 30 | 30 |
| S | 5 | 5 | 5 |
| I | 10 | 20 | 30 |
| Wynik | 3480 | 3417 | 3375 |

Dane 100x20:

|  |  |  |  |
| --- | --- | --- | --- |
| N | 10 | 10 | 10 |
| S | 5 | 5 | 5 |
| I | 10 | 20 | 30 |
| Wynik | 7517 | 7445 | 7383 |

Dane 200x20:

|  |  |  |
| --- | --- | --- |
| N | 50 | 50 |
| S | 5 | 5 |
| I | 10 | 20 |
| Wynik | 13107 | 13092 |

W 3 powyższych tabelach zaprezentowano wybrane wyniki algorytmu w przypadku, gdy stałe są wielkość sąsiedztwa oraz długość listy tabu, a zmienia się liczba iteracji. Wnioskiem, który można wyciągnąć, ze wszystkich tabel jest to, że większa liczba iteracji ma pozytywny wpływ na wyniki algorytmu. Jest to więc zgodne z logiką, im dłużej algorytm będzie pracował, tym bardziej prawdopodobne, że osiągnie dobre wyniki.

* 1. Wnioski.

Podsumowując, prawdopodobnie największy wpływ na wynik uzyskany przez algorytm Tabu Search ma wielkość sąsiedztwa, które jest przeszukiwane w poszukiwaniu rozwiązań. Dodatkowo ważne jest także, aby dobrać odpowiednio długą listę tabu. W przypadku, gdy będzie ona zbyt krótka możliwe jest wpadnięcie w ciąg powtarzających się zamian, natomiast jeżeli byłaby ona zbyt długa blokowaliśmy możliwość poprawy algorytmu. Kolejnym czynnikiem wpływającym na wynik jest liczba iteracji algorytmu. Gdy algorytm wykonuje się zbyt mało razy możliwe jest, że nie wykorzysta on pełni potencjału danego rozwiązania i nie dokona wszystkich możliwych popraw.

1. **Algorytmy genetyczne.**

Zaimplementowany algorytm genetyczny ma 6 danych wejściowych: zbiór danych, liczbę iteracji, wielkość populacji, liczba krzyżowań, liczba mutacji oraz długość przekazywanego genotypu dziecku. W badaniu sprawdzono jaki wpływ na wyniki mają 3 z nich: wielkość populacji, liczba krzyżowań oraz długość genotypu. Natomiast pozostałe dane wejściowe były stałe i wynosiły odpowiednio, liczba iteracji – 50, liczba mutacji – 5.

* 1. Wpływ wielkości populacji na otrzymane wyniki.

Dane 50x10:

|  |  |  |  |
| --- | --- | --- | --- |
| Population | 50 | 100 | 200 |
| Crosses | 25 | 50 | 100 |
| MaskLen | 20 | 20 | 20 |
| Wynik | 3458 | 3531 | 3428 |

Dane 100x20:

|  |  |  |  |
| --- | --- | --- | --- |
| Population | 50 | 100 | 200 |
| Crosses | 25 | 50 | 100 |
| MaskLen | 40 | 40 | 40 |
| Wynik | 7477 | 7368 | 7342 |

Dane 200x20:

|  |  |  |
| --- | --- | --- |
| Population | 30 | 60 |
| Crosses | 15 | 30 |
| MaskLen | 80 | 80 |
| Wynik | 13313 | 13334 |

W powyższych tabelach zaprezentowano, jak zmieniały się wyniki, przy stałej liczbie krzyżowań wynoszącej 50% liczby populacji oraz stałej długości genu przekazywanego potomkowi i zmieniającej się wielkości populacji. Można zauważyć, że im większa populacja, tym mniejsza wartość funkcji celu. Wyniki potwierdzające to odnotowano w danych 50x10 oraz 100x20. W przypadku danych 200x20 wyniki były bardzo zbliżone do siebie, więc mógł mieć wpływ na to również fakt, że populacja była dość mała.

* 1. Wpływ liczby krzyżowań na otrzymane wyniki.

Dane 50x10:

|  |  |  |  |
| --- | --- | --- | --- |
| Population | 200 | 200 | 200 |
| Crosses | 30 | 50 | 100 |
| MaskLen | 20 | 20 | 20 |
| Wynik | 3531 | 3410 | 3327 |

Dane 100x20:

|  |  |  |
| --- | --- | --- |
| Population | 100 | 100 |
| Crosses | 25 | 50 |
| MaskLen | 40 | 40 |
| Wynik | 7407 | 7368 |

Dane 200x20:

|  |  |  |
| --- | --- | --- |
| Population | 60 | 60 |
| Crosses | 20 | 30 |
| MaskLen | 80 | 80 |
| Wynik | 13332 | 13334 |

Podobnie jak w przypadku badania wpływu wielkości populacji zbadano również wpływ liczby krzyżówek na wynik końcowy, przy stałych wielkościach populacji oraz długości genu. Jak można zauważyć w przypadku pierwszych i drugich danych wraz ze zwiększeniem liczby krzyżówek spadł wynik końcowy. Natomiast w przypadku danych największych ponownie otrzymano bardzo zbliżone wyniki, jednak tutaj również wielkość populacji nie była zbyt wysoka, więc dużą rolę odgrywa tutaj czynnik losowy algorytmu.

* 1. Wpływ długości przekazywanego genotypu na otrzymane wyniki.

Dane 50x10:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Population | 200 | 200 | 200 | 200 |
| Crosses | 50 | 50 | 50 | 50 |
| MaskLen | 10 | 20 | 40 | 45 |
| Wynik | 3427 | 3410 | 3399 | 3449 |

Dane 100x20:

|  |  |  |
| --- | --- | --- |
| Population | 100 | 100 |
| Crosses | 25 | 25 |
| MaskLen | 20 | 40 |
| Wynik | 7443 | 7407 |

Dane 200x20:

|  |  |  |
| --- | --- | --- |
| Population | 60 | 60 |
| Crosses | 30 | 30 |
| MaskLen | 80 | 160 |
| Wynik | 13334 | 13318 |

W powyższych tabelach można zauważyć jak wyglądają wyniki przy stałej wielkości populacji, stałej liczbie krzyżówek oraz różnej długości genu. W przypadku danych 50x10 funkcja celu osiągnęła najniższą wartość w przypadku długości maski 40, a najwyższą natomiast w przypadku bardzo krótkiej – 10 oraz bardzo długiej 45. W dwóch pozostałych przypadkach można zauważyć, że im dłuższa maska tym lepszy rezultat. Na tej podstawie można stwierdzić, że musi zostać zachowany odpowiedni bilans w dobieraniu długości maski. Nie może ona być krótka, prawdopodobnie powinna zawierać więcej niż 50% rodzica, jednak nie może być też zbyt długa.

* 1. Wnioski.

Podsumowując, w przypadku algorytmów genetycznych każdy z badanych czynników ma wpływ na wyniki algorytmu. Im większa populacja tym większa szansa na wylosowanie dobrych rodziców, z których mogą powstać potomkowie. Przy tym warto także dobrać możliwie dużą liczbę krzyżowań w danej iteracji, tak aby jak najwięcej genów krzyżowało się ze sobą. Dodatkowo istotna jest również długość przekazywanego genu. Nie może być ona zbyt krótka, aby przekazać znaczącą część dobrego rodzica, ale także nie może być zbyt długa, aby potomek mógł zostać udoskonalony.

1. **Symulowane wyżarzanie.**

*Symulowane wyżarzanie (SA)*

Algorytm wyżarzania posiada **5** parametrów takich jak:

**T** - początkowa temperatura, T>0

**F** - temperatura zamrożenia,

**L** - ile razy ma się wykonywać pętla for

**r** - współczynnik wygaszania

**n** - jak duże ma być sąsiedztwo dla rozwiązania

W badaniu sprawdziliśmy zależność 3 parametrów takich jak **n**, który opisuje jak duże ma być sąsiedztwo, parametru **T** czyli początkowej temperatury (T>0) oraz parametru **r** czyli współczynnika wygaszania. Algorytm, tak jak inne przygotowane, ma na celu ustawić zadania w takiej kolejności aby ich czas wykonania był jak najkrótszy.

* 1. Zbadanie jak uzyskane wyniki zależą od zmiany liczby sąsiadów dla rozwiązania.

W tym kroku przyjęliśmy wartości stałe dla pozostałych parametrów, zmienia się jedynie liczba sąsiedztwa.

**Wartości stałe:**

T - 1

F - 0.3

L - 15

r - 0.9

Dla danych z 50 zadaniami:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n | 15 | 20 | 30 | **50** |
| najlepszy wynik | 3582 | 3423 | 3426 | **3280** |

Dla danych z 100 zadaniami:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| n | 15 | 20 | 30 | 50 | 80 | **100** |
| najlepszy wynik | 7549 | 7458 | 7514 | 7435 | 7228 | **7108** |

Dla danych z 200 zadaniami

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| n | 15 | 30 | 50 | 80 | 100 | **150** |
| najlepszy wynik | 13620 | 13319 | 13147 | 13011 | 12965 | **12938** |

Analizując powyższe wyniki, możemy stwierdzić, że wraz ze wzrostem liczebności sąsiedztwa program uzyskuje coraz lepsze wyniki. Możemy zauważyć, że nie zawsze większa liczba n powoduje polepszenie wyniku (np. dla danych z 100 zadaniami, parametr n=20 uzyskał lepszy wynik od parametru n=30). Może być to spowodowane elementem losowości, który występuję w algorytmie. Pierwsze rozwiązanie, od którego zależą kolejne kroki jest wybierane losowo. Ma to wpływ na ostateczny wynik oraz szybkość wykonania programu. Najlepsze wyniki uzyskaliśmy zawsze dla największego parametru, wynoszącego kolejno 50, 100 oraz 150.

* 1. Sprawdzenie zależności wyników od zmiany początkowej temperatury (T)

W tym kroku przyjęliśmy wartości stałe dla pozostałych parametrów, zmienia się jedynie temperatura początkowa (T).

**Wartości stałe:**

F - 0.3

L - 15

r - 0.9

n - 30

Dla danych z 50 zadaniami:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| T | 5 | 10 | **20** | 30 |
| najlepszy wynik | 3596 | 3530 | **3480** | 3625 |

Dla danych z 100 zadaniami:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| T | 5 | 10 | **20** | 30 |
| najlepszy wynik | 7423 | 7328 | **7292** | 7328 |

Dla danych z 200 zadaniami

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| T | 5 | **10** | 20 | 30 |
| najlepszy wynik | 13633 | **13472** | 13800 | 13521 |

Na podstawie powyższych wyników, nie możemy jednoznacznie stwierdzić, że wyniki zwiększają się wraz ze wzrostem początkowej temperatury ani również w odwrotnej kolejności. Głównym czynnikiem odpowiadającym za uzyskanie wyników był najprawdopodobniej czynnik losowy. Mimo iż w dwóch przypadkach najlepsza wartością parametru T okazała się liczba 20 to nie mówi nam to o 100% pewności takiego wyniku w kolejnych próbach.

* 1. Sprawdzenie zależności wyników od zmiany współczynnika wygaszania (r)

W tym kroku przyjęliśmy wartości stałe dla pozostałych parametrów, zmienia się jedynie współczynnik wygaszania (r).

**Wartości stałe:**

T - 1

F - 0.3

L - 15

n – 30

Dla danych z 50 zadaniami:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| r | 0.5 | 0.8 | **0.95** | 0.99 |
| najlepszy wynik | 3602 | 3642 | **3377** | 3719 |

Dla danych z 100 zadaniami:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| r | 0.5 | 0.8 | 0.95 | **0.99** |
| najlepszy wynik | 7501 | 7516 | 7504 | **7489** |

Dla danych z 200 zadaniami:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| r | **0.5** | 0.8 | 0.95 | 0.99 |
| najlepszy wynik | **13297** | 13720 | 13561 | 13427 |

Na podstawie powyższych wyników również możemy stwierdzić, że zmiana współczynnika wygaszania (czy to zwiększanie czy zmniejszanie) nie ma większego znaczenia na najlepszy otrzymany wynik. Przyczynia się do tego głównie czynnik losowy.

* 1. Próba dobrania parametrów do uzyskania najlepszego wyniku.

W tym badaniu staraliśmy się dopasować takie wartości parametrów aby uzyskać jak najlepszy wynik z algorytmu. Wykonaliśmy w tym celu wiele prób aby wybrać najlepsze zestawienie wartości parametrów.

1. Dla danych z 50 zadaniami:

T - 10

F - 0.3

L - 20

r - 0.95

n - 50

**Otrzymaliśmy wynik 3066**

1. Dla danych z 100 zadaniami:

T - 10

F - 0.3

L - 20

r - 0.95

n – 100

**Otrzymaliśmy wynik 6848**

1. Dla danych z 200 zadaniami:

Nie udało nam się poprawić wyniku w porównaniu do wcześniej uzyskanego **12938.**

* 1. Wnioski

Analiza wyników algorytmu wyżarzania oraz analiza wpływu parametrów na te wyniki okazała się najbardziej wiarygodna w przypadku zmiany liczebności sąsiedztwa. Otrzymaliśmy wyniki, które jednoznacznie mówią nam, że najlepszy rezultat uzyskamy w próbie z największą liczbą sąsiadów. Badanie pozostałych dwóch parametrów nie ukazuje żadnych trendów w otrzymanych wynikach. Na podstawie zestawienia parametrów w celu uzyskania najlepszego wyniku, możemy zauważyć, że kluczowym aspektem było zwiększenie liczby iteracji do 20 oraz przyjęcie liczby sąsiadów odpowiednio dla największych wartości dla poszczególnych danych.