Wrocław, 16.01.2020r.  
Prowadzący: dr inż. Zbigniew Buchalski

Projektowanie efektywnych algorytmów

**Projekt №2**

Implementacja i analiza efektywności algorytmu Genetucznego dla problemu komiwojażera

Nikita Stepanenko  
nr albumu 245816

# Założenia projektowe

Zadaniem projektowym była implementacja dla problemu komiwojażera algorytmów Tabu Search i Symulowanego Wyżarzania oraz porównanie efektywności obu algorytmów, a także wyznaczenie błędu względnego (porównując z najlepszym znanym rozwiązaniem).

Problem komiwojażera jest jednym z najbardziej znanych i wymagających zagadnień optymalizacyjnych dla informatyka. Polega on na znalezieniu ścieżki o najmniejszej sumie wag przechodzącej przez każdy wierzchołek grafu dokładnie jeden raz i wracającej do wierzchołka początkowego. Jest to problem NP-trudny.

Istota i nazwa problemu bierze się z zagwozdki, jaką komiwojażer może mieć przy układaniu trasy, na której odwiedzi określone z góry miasta – zależy mu bowiem na jak najkrótszej ścieżce. Rozróżniamy symetryczne (*STSP* – ang. *Symmetric Traveling Salesman Problem*) i asymetryczne (*ATSP* – ang. *Asymmetric Traveling Salesman Problem*) przypadki tego problemu. *STSP* polegają na tym, że waga krawędzi od wierzchołka *A* do *B* jest taka sama, jak od wierzchołka *B* do *A*. Asymetryczna wersja nie spełnia warunku dla każdej takiej pary krawędzi.

W tym projekcie przedstawione zostaną rozwiązanie za pomocą algorytmu Genetycznego

Ten algorytm dążą do rozwiązania problemu poprzez wykorzystanie pewnych pojęć zdefiniowanych przez dany algorytm. Nie daje gwarancji znalezienia najlepszego możliwego rozwiązania, ale wymagają znacznie mniejszej złożoności obliczeniowej, co powoduje w wielu wypadkach, że okazują się znacznie użyteczniejsze od dokładnego rozwiązania.

# Stecyfikacja techniczna

* Progrm zrealizowany w języku programowania C++
* Struktury przechowujące dane alokowane są dynamicznie, zależnie od rozmiaru problemu
* Program posiada możliwość wczytywania danych z pliku
* Program posiada możliwość wprowadzenia kryterium stopu
* Algorytmy zostały zaimplementowane zgodnie z paradygmatami programowania obiektowego
* Czas wykonania algorytmu mierzony był z dokładnością do nanosekund, przy wykorzystaniu bibliotek systemowych (chrono)

# Opis programu

Poniżej przedstawiony został wykorzystany w programie pseudokod dla rozwiązania problemu Komiwojażera przy pomocy algorytmu genetycznego:



Na początku określamy rozmiar *popsize* populacji, tworzymy zbiór danych reprezentujących ją i wypełniamy ją losowymi rozwiązaniami z przestrzeni wszystkich rozwiązań problemu. *Popsize* jest drugim parametrem wywołania programu, który użytkownik wprowadza z klawiatury.

Kolejnym krokiem jest utworzenie pętli głównej programu, która może być wykonywana aż do znalezienia optymalnego rozwiązania, upływu zadanego czasu, bądź przez określoną liczbę iteracji. W prezentowanym programie wykorzystane zostało ostatnie rozwiązanie, a liczbę iteracji pętli użytkownik zadaje z klawiatury jako pierwszy parametr wywołania programu.

Na początku pętli konieczne jest ocenienie jakości wszystkich osobników populacji i zapamiętanie najlepszego dotychczasowego rozwiązania. Jakość osobników będzie miała znaczenie dla selekcji rodziców. Zapamiętanie najlepszego osobnika jest konieczne, aby program mógł pod koniec działania zwrócici go nam – w końcu poszukujemy najlepszego możliwego do znalezienia rozwiązania.

Najważniejszą częścią algorytmu jest druga pętla *for* w głównej pętli programu. To w niej wywoływane mamy funkcje selekcji, krzyżowania i mutacji, które określają przebieg poszukiwań minimum globalnego.

Funkcja *SelectWithReplacement* została zaimplementowana jako selekcja turniejowa. Jest to prosty algorytm pozwalający na pseudolosowe poszukiwanie rozwiązań w zależności od rozmiaru turnieju. Rozmiar turnieju decyduje o tym, czy selekcja jest w pełni (pseudo)losowa, czy też faworyzowane są osobniki o najlepszych cechach oraz jak bardzo są faworyzowane. Rozmiar ten jest trzecim parametrem programu wprowadzanym przez użytkownika.

Za pomocą funkcji selekcji wybieramy parę rodziców i możemy przystąpić do ich krzyżówki. Krzyżowanie *Crossover* zostało zaimplementowane za pomocą operatora *OX.* Jest to efektywny sposób pozwalający na uzyskanie dobrych wyników końcowych. Do rozmnażania jednak nie musi dojść w każdym przypadku. Przeważnie określa się je za pomocą parametru jego prawdopodobieństwa i tak też dzieje się w przedstawionym programie. Parametr ten jest czwartym argumentem wprowadzanym przy uruchamianiu programu.

Funkcja *Mutate* została zaimplementowana za pomocą operatora *transposition,* znanego również jako *swap*. Jest to prosta mutacja nieodbiegająca dalece od przypadków rzeczywistych, na których wzorowany jest algorytm - taka mutacja nie zmienia znacznie, lecz delikatnie genotyp mutowanego osobnika. Można powiedzieć, że spełnia zasadę podobieństwa sąsiadów. Mutacja zachodzi z określonym prawdopodobieństwem – jest to piąty, ostatni parametr programu.

Zaimplementowany został również operator mutacji *invert*. Choć większość pomiarów przedstawionych w tym sprawozdaniu została wykonana przy pomocy *transposition*, ostatecznie najlepsze wyniki uzyskane zostały przy pomocy inwersji. Metoda ta została wybrana z powodu lepszych rezultatów uzyskiwanych przez nią.

# Testy i wykresy

Po każdej zmianie wspóczynnika byli przeprowadzone testy 10 raz.

## 47 miast („ftv47.atsp”)

* Mutacja 1: wymiana 2-ch losowych punktów miejscami
* Zmiana populacji

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 10 | 80 | 1 | 2540 | 43,01801802 |
| 20 | 80 | 1 | 2580 | 45,27027027 |
| 50 | 80 | 1 | 2434 | 37,04954955 |
| 100 | 80 | 1 | 2560 | 44,14414414 |
| 200 | 80 | 1 | 2593 | 46,00225225 |

Zmiana krzyżowania

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 50 | 50 | 1 | 2765 | 55,68693694 |
| 50 | 70 | 1 | 2642 | 48,76126126 |
| 50 | 80 | 1 | 2434 | 37,04954955 |
| 50 | 90 | 1 | 2614 | 47,18468468 |

* Zmiana mutacji

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 50 | 80 | 1 | 2540 | 43,01801802 |
| 50 | 80 | 2 | 2633 | 48,2545045 |
| 50 | 80 | 5 | 2453 | 38,11936937 |
| 50 | 80 | 10 | 2774 | 56,19369369 |

* Mutacja 2: wymiana 2-ch losowych punktów miejscami
  + Zmiana populacji

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 10 | 80 | 1 | 2621 | 47,57882883 |
| 20 | 80 | 1 | 2419 | 36,20495495 |
| 50 | 80 | 1 | 2434 | 37,04954955 |
| 100 | 80 | 1 | 2413 | 35,86711712 |
| 200 | 80 | 1 | 2435 | 37,10585586 |

* + Zmiana krzyżowania

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 20 | 50 | 1 | 3480 | 95,94594595 |
| 20 | 70 | 1 | 3143 | 76,97072072 |
| 20 | 80 | 1 | 2419 | 36,20495495 |
| 20 | 90 | 1 | 3137 | 76,63288288 |

* + Zmiana mutacji

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 20 | 80 | 1 | 2419 | 36,20495495 |
| 20 | 80 | 2 | 2934 | 65,2027027 |
| 20 | 80 | 5 | 3265 | 83,84009009 |
| 20 | 80 | 10 | 3058 | 72,18468468 |

## 170 miast („ftv170.atsp”)

* Mutacja 1: wymiana 2-ch losowych punktów miejscami
* Zmiana populacji

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| * populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 10 | 80 | 1 | 4800 | 94,72616633 |
| 20 | 80 | 1 | 4815 | 95,3346856 |
| 50 | 80 | 1 | 4711 | 91,11561866 |
| 100 | 80 | 1 | 4813 | 95,2535497 |
| 200 | 80 | 1 | 4796 | 94,56389452 |

* Zmiana krzyżowania

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 50 | 50 | 1 | 6233 | 126,2431942 |
| 50 | 70 | 1 | 6020 | 118,5117967 |
| 50 | 80 | 1 | 6364 | 130,9981851 |
| 50 | 90 | 1 | 6456 | 134,3375681 |

* Zmiana mutacji

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 50 | 80 | 1 | 6440 | 133,7568058 |
| 50 | 80 | 2 | 6007 | 118,0399274 |
| 50 | 80 | 5 | 5927 | 115,1361162 |
| 50 | 80 | 10 | 6319 | 129,3647913 |

* Mutacja 2: wymiana 2-ch losowych punktów miejscami
  + Zmiana populacji

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 10 | 80 | 1 | 6129 | 122,4682396 |
| 20 | 80 | 1 | 6090 | 121,0526316 |
| 50 | 80 | 1 | 6434 | 133,53902 |
| 100 | 80 | 1 | 6431 | 133,430127 |
| 200 | 80 | 1 | 6300 | 128,6751361 |

* + Zmiana krzyżowania

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 20 | 50 | 1 | 5959 | 116,2976407 |
| 20 | 70 | 1 | 5845 | 112,1597096 |
| 20 | 80 | 1 | 5890 | 113,7931034 |
| 20 | 90 | 1 | 5919 | 114,845735 |

* + Zmiana mutacji

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 20 | 80 | 1 | 6540 | 137,3865699 |
| 20 | 80 | 2 | 6636 | 140,8711434 |
| 20 | 80 | 5 | 6610 | 139,9274047 |
| 20 | 80 | 10 | 6685 | 142,6497278 |

## 403 miasta („rbg403.atsp”)

* Mutacja 1: wymiana 2-ch losowych punktów miejscami
* Zmiana populacji

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| * populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 10 | 80 | 1 | 6440 | 133,7568058 |
| 20 | 80 | 1 | 6548 | 137,676951 |
| 50 | 80 | 1 | 6364 | 130,9981851 |
| 100 | 80 | 1 | 6862 | 149,0744102 |
| 200 | 80 | 1 | 6432 | 133,4664247 |

* Zmiana krzyżowania

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| * populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 50 | 50 | 1 | 4951 | 100,851927 |
| 50 | 70 | 1 | 4979 | 101,9878296 |
| 50 | 80 | 1 | 4711 | 91,11561866 |
| 50 | 90 | 1 | 5022 | 103,7322515 |

* Zmiana mutacji

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 50 | 80 | 1 | 4800 | 94,72616633 |
| 50 | 80 | 2 | 4643 | 88,35699797 |
| 50 | 80 | 5 | 4553 | 84,70588235 |
| 50 | 80 | 10 | 4682 | 89,93914807 |

* Mutacja 2: wymiana 2-ch losowych punktów miejscami
  + Zmiana populacji

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 10 | 80 | 1 | 6735 | 173,2251521 |
| 20 | 80 | 1 | 6515 | 164,3002028 |
| 50 | 80 | 1 | 6644 | 169,5334686 |
| 100 | 80 | 1 | 6635 | 169,168357 |
| 200 | 80 | 1 | 6588 | 167,2616633 |

* + Zmiana krzyżowania

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 20 | 50 | 1 | 6616 | 168,3975659 |
| 20 | 70 | 1 | 6504 | 163,8539554 |
| 20 | 80 | 1 | 6515 | 164,3002028 |
| 20 | 90 | 1 | 6547 | 165,5983773 |

* + Zmiana mutacji

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| populacja | krzyżowanie [%] | mutacja [%] | najlepsza droga | błąd [%] |
|  |  |  |  |  |
| 20 | 80 | 1 | 6515 | 164,3002028 |
| 20 | 80 | 2 | 6368 | 158,336714 |
| 20 | 80 | 5 | 6674 | 170,7505071 |
| 20 | 80 | 10 | 6717 | 172,494929 |

# Porównanie Tabu Search oraz Algorytmu Genetycznego

Najlieprzą ze wszystkich parametrów jest metoda mutacji 1 oraz pprawdopodobieństwo mutacji = 0,05.

Plik ftv47.atsp:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| max czas [s] | droga Tabu | droga Ge | blad Tabu % | blad Ge % |
|  |  |  |  |  |
| 30 | 2119 | 2441 | 19,31306306 | 37,44369369 |
| 60 | 2037 | 2446 | 14,69594595 | 37,72522523 |
| 120 | 1965 | 2435 | 10,64189189 | 37,10585586 |
| 180 | 2037 | 2455 | 14,69594595 | 38,23198198 |
| 240 | 1965 | 2505 | 10,64189189 | 41,0472973 |
| 300 | 2049 | 2518 | 15,37162162 | 41,77927928 |
| 360 | 2012 | 2520 | 13,28828829 | 41,89189189 |

Plik ftv170.atsp:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| max czas [s] | droga Tabu | droga Ge | blad Tabu % | blad Ge % |
|  |  |  |  |  |
| 30 | 5017 | 6176 | 82,10526316 | 124,1742287 |
| 60 | 4954 | 5967 | 79,8185118 | 116,5880218 |
| 120 | 4873 | 5449 | 76,8784029 | 97,78584392 |
| 180 | 4826 | 5441 | 75,17241379 | 97,49546279 |
| 240 | 4865 | 5366 | 76,58802178 | 94,77313975 |
| 300 | 4982 | 5034 | 80,83484574 | 82,72232305 |
| 360 | 3652 | 4844 | 32,55898367 | 75,82577132 |

Plik rbg403.atsp:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| max czas [s] | droga Tabu | droga Ge | blad Tabu % | blad Ge % |
|  |  |  |  |  |
| 30 | 2814 | 7816 | 14,15821501 | 217,0791075 |
| 60 | 2786 | 7389 | 13,02231237 | 199,7565923 |
| 120 | 2786 | 7193 | 13,02231237 | 191,8052738 |
| 180 | 2786 | 7491 | 13,02231237 | 203,8945233 |
| 240 | 2786 | 7397 | 13,02231237 | 200,0811359 |
| 300 | 2786 | 7329 | 13,02231237 | 197,3225152 |
| 360 | 2781 | 7328 | 12,81947262 | 197,2819473 |

# Wnioski

Ten algorytm był bardzo przyjemny do napisania. Niestety ten algorym jest dużo gorszy niż Tabu Search, ale daje możliwość dopasować parametry do konkretnego zadania. Jest dużo szypszy od Brute Force, ale, jak i inne algorytmy zrobione w tym semestrze, nie daje 100% pewności że wynik jest najlepszy, można powidzieć więcej, prawie zawsze ma nie najleprzą odpowiedź, ale są dość szybkie.