# WSTĘP DO SZTUCZNEJ INTELIGENCJI

*Ćwiczenie 2 – Algorytm Ewolucyjny*

MICHAŁ MIZIA 331407

# SPIS TREŚCI

## Wstęp

Ewolucja kodu:

Kod algorytmu genetycznego przeszedł serię przemian, ostatecznie powstały 2 pliki:

- ga_tournament.py
- ga_roulette.py

Na początku ciężko było selekcją ruletkową osiągnąć wynik zbliżony do selekcji turniejowej. Wypisywanie wszystkich członków populacji pod koniec pozwalało stwierdzić że algorytm za szybko schodzi do minimum lokalnego i nie potrafi z niego wyjść. Np. przy populacji 200 po 100 iteracjach wszyscy członkowie mieli dokładnie tą samą ścieżkę.

Zmiany jakie pozwoliły to naprawić:

- Zmiana obliczeń fitness z 1/(dł_trasy) na 1/(dł_trasy – min_eval + 1e-6), to skutkowało w większej losowości przy wyborze osobników.
- Zmiana selekcji ruletkowej z całej populacji na połowę populacji, następnie gdy wybierzemy nową populację o wielkości 0,5 starej, przechodzimy do funkcji crossover która zwraca populację do starej wielkości. Dla każdego osobnika albo dobiera mu losową parę w wypadku krzyżowania, albo wpisuje jego i dodatkowego osobnika do populacji.

```python
def apply_crossovers(population: List[Member], cross_chance: float) -> List[Member]:
    new_population = []
    for i in range(0, len(population)):
        parent1 = population[i]
        parent2 = population[np.random.choice(len(population))]
        if np.random.random() < cross_chance:
            # Perform crossover and get two offspring
            offspring1, offspring2 = crossover(parent1, parent2)
            new_population.extend([offspring1, offspring2])
        else:
            # add copies of parents to new population
            new_population.extend([parent1, parent2])

    return new_population
```

- Do tego między selekcją turniejową a ruletkową zaszło dużo zmian które pozwoliły na osiągnięcie w miarę sensownych wyników.

## Wybór metod

- Krzyżowanie

Jako metoda krzyżowania zostało wybrane krzyżowanie członków po kolei z innym losowym członkiem populacji przy użyciu krzyżowania OX 2 punktowego z prawdopodobieństwem 0.9

- Mutacja

Mutacja jest wykonywana przez losową zamianę 2 miast z prawdopodobieństwem 0,05.

- Selekcja

Selekcja ruletkowa przy użyciu wartości fitness = 1/(dł_trasy – min_eval + 1e-6) oraz znormalizowaniu wartości tak żeby suma wszystkich wartości fitness wynosiła 1

## Wnioski

Średnia najlepsza ewaluacja, najlepsza ewaluacja i dewiacja standardowa dla grupy 5 eksperymentów przy różnych populacjach, miasta to Warszawa i Rzeszów; prawdopodobieństwo mutacji => 0,05; prawdopodobieństwo krzyżowania => 0,9; maksymalna ilość iteracji => 200;

| Population Size | Best Eval | Avg Best Eval | Std Dev |
|---|---|---|---|
| 50 | 8806.705999999998 | 9280.0948 | 421.97132088775976 |
| 100 | 7656.2080000000005 | 8253.6018 | 645.352515098036 |
| 150 | 7187.010000000001 | 7568.591399999999 | 444.2416054394718 |
| 200 | 6791.5999999999985 | 7349.1291999999985 | 415.4400571728252 |
| 250 | 7042.079000000003 | 7427.818400000001 | 339.91751764661973 |
| 300 | 6923.763999999999 | 7459.1154 | 334.44158317565746 |
| 350 | 6553.179000000001 | 7115.272800000001 | 473.55973911024205 |
| 400 | 6410.528999999998 | 6739.8726 | 294.38306316607446 |
| 450 | 6142.025 | 6784.4396 | 546.2612543388368 |

Widzimy że wraz ze wzrostem wielkości populacji długość najkrótszej trasy spada, prawdopodobnie spadałaby dalej gdyby zwiększać ilość populacji do wartości około 5500km (najlepsza wartość jaką udało się uzyskać). Prawdopodobnie jest tak ponieważ skuteczność algorytmu jest ograniczona przez to jak szybko skonwertuje on do lokalnego minimum a przy większych populacjach konwertuje wolniej.

Poniżej mapa trasy Warszawa – Rzeszów o długości 6100km. Każdą znalezioną trasę można na bieżąco wizualizować przy pomocy wizualizatora. Generuje on plik .html z punktami z map google który następnie należy odpalić przy pomocy rozszerzenia Live Server