AISDI - ćwiczenie 2

8 kwietnia 2026

1 Sprawy organizacyjne

1. Ćwiczenie realizowane jest samodzielnie.
2. Ćwiczenie wykonywane jest w języku Python.
3. Ćwiczenie powinno zostać oddane najpóźniej do 06.05.2026. W ramach oddawania ćwiczenia należy zademonstrować prowadzącemu działanie kodu oraz utworzyć pull request (z kodem oraz raportem) który prowadzący będzie mógł komentować.
4. Rozwiązanie powinno wykorzystywać optymalne dla tego problemu struktury danych oraz algorytmy.
5. Kod oraz raport powinien być umieszczony na repozytorium `gitlab-stud.elka.pw.edu.pl`, a pull request utworzony do konta prowadzącego: **@djagodzi**
6. Raport powinien być w postaci pliku .pdf, .html albo być częścią notebooka jupyterowego. Powinien zawierać koncepcję rozwiązania i testowania, opis eksperymentów, uzyskane wyniki wraz z komentarzem oraz wnioski.
7. Na ocenę wpływa poprawność oraz jakość kodu i raportu.
8. Implementacja algorytmu powinna być ogólna. W szczególności powinna być możliwa do zastosowania dla dowolnego zbioru danych oraz dowolnej wielkości pamięci.

2 Ćwiczenie

Celem ćwiczenia jest zaprojektowanie i implementacja systemu do optymalizacji tras w sieci dróg miejskich, uwzględniającego zmienność ruchu w ciągu dnia.

2.1 Dane

Zadana jest sieć drogowa miasta w postaci w postaci grafu: city-big.csv Znaczenie poszczególnych kolumn:

od ID skrzyżowania początkowego

do ID skrzyżowania końcowego

odleglosc Fizyczna odległość między skrzyżowaniami

czas-rano Czas przejazdu w godzinach 7:00-9:00

czas-popoludnie Czas przejazdu w godzinach 10:00-16:00

czas-wieczor Czas przejazdu w godzinach 17:00-19:00

Graf jest nieskierowany - każda krawędź działa w obie strony (od→do i do→od). W godzinach nocnych (20:00→6:00) można założyć że czas przejazdu wynosi 75% czasu czas-popoludnie

2.2 Program

Program powinien składać się z optymalnie zaimplementowanych części, które:

1. Wczytują dane z pliku csv i budują reprezentację w postaci grafu.
2. Znaleźć najkrótszą trasę między dowolnymi dwoma skrzyżowaniami dla wybranej pory dnia.
3. Wyznaczyć sieć głównych arterii - minimalny zbiór dróg łączący wszystkie skrzyżowania (minimalizując całkowity czas przejazdu)

Wymagania techniczne:

- Samodzielna implementacja struktur danych (bez użycia gotowych bibliotek do grafów jak networkx)
- Dopuszczone jest uzycie standardowych struktur Pythona np. list, dict, set

2.3 Badania

Po zaimplementowaniu algorytmu systemu do optymalizacji tras w sieci dróg miejskich, uwzględniającego zmienność ruchu w ciągu dnia, zmierz i porównaj czas wyznaczenia sieci głównych arterii (minimalny zbiór dróg łączący wszystkie skrzyżowania) dla zbiorów: city-big.csv(1000 skrzyżowań), mid-big.csv(200 skrzyżowań) oraz city-small.csv(50 skrzyżowań), dla każdych pór dnia.