Karolina Pawlak 325698
Maja Kret 325693
Maja Płaciszewska 325699

Semestr 5 grupa 2 GI

Rok akad. 2024/2025

**Politechnika Warszawska**
**Wydział Geodezji i Kartografii**

Data oddania ćwiczenia:

Zaliczenie:

# PROGRAMOWANIE APLIKACJI GEOINFORMACYJNYCH 2

## Raport – Wyznaczanie trasy w sieci drogowej

## 1. CEL PROJEKTU I WYBRANE ROZSZERZENIE

Celem projektu jest stworzenie i implementacja algorytmu wyznaczającego najkrótszą trasę w sieci drogowej dla danych z BDOT10k. Wyznaczana trasa ma być rozumiana jako najkrótsza pod względem długości geometrii trasy lub długości czasu jej przebycia.

Dodatkowym rozszerzeniem projektu jest implementacja dodatku w środowisku ESRI, który dodaje elementy interfejsu umożliwiające wykonanie procesu wyznaczania trasy i prezentacji wyników.

## 2. ETAPY REALIZACJI PROJEKTU I NAPOTKANE PROBLEMY

- Odczyt sieci drogowej w wybranym środowisku

  Projekt wykonywany jest w środowisku ESRI ArcGIS Pro. Jako dane wejściowe pobrano warstwę SKJZ z BDOT10k dla powiatu toruńskiego oraz miasta Toruń, a następnie wczytano je do programu.

- Utworzenie grafu z danych źródłowych

  W danych źródłowych są tylko krawędzie, dlatego głównym problemem jest wygenerowanie wierzchołków. Kolejnym problemem jaki się pojawia jest sposób identyfikowania wierzchołków oraz powiązany z nim problem topologii sieci (drogi leżące bardzo blisko siebie nie są połączone).

  Za pomocą kursora, który iteruje po każdym obiekcie w pliku .shp, pobrano dane o drogach (geometria, ID, klasa drogi), a także punkt początkowy i końcowy każdej z nich (współrzędne w układzie 92 zaokrąglane są do liczb całkowitych, dzięki czemu zapewniona może być spójność topologii). W ten sposób punkty te są dodawane do grafu jako wierzchołki o identyfikatorze równemu zaokrąglonemu (x,y).

  Jako wynik uzyskano:
  - kolekcję wierzchołków grafu o atrybutach**: id**, **x, y, edges** (id krawędzi wychodzących z danego wierzchołka)
  - kolekcję krawędzi grafu o atrybutach: **id, from** (id wierzchołka początkowego, **to** (id wierzchołka końcowego), **road_id, length** (długość geometrii jezdni w metrach) i **speed** (prędkość określana na podstawie klasy drogi [Tabela 1]).

*Tabela 1 - Prędkości dróg w zależności od klasy*

| Klasa drogi | Oznaczenie | Przyjęta prędkość [km/h] |
|---|---|---|
| Autostrada | A | 140 |
| Droga ekspresowa | S | 120 |
| Droga główna ruchu przyspieszonego | GP | 70 |
| Droga główna | G | 60 |
| Droga zbiorcza | Z | 50 |
| Droga lokalna | L | 40 |
| Droga dojazdowa | D | 20 |
| Inna | I | 20 |

*Fragmenty wyeksportowanych plików z krawędziami i wierzchołkami grafu:*

| id | x | y | edges |
|---|---|---|---|
| 0 | 477173.0 | 576037.0 | 1,5028,6867,6870 |
| 1 | 477068.0 | 575902.0 | 1,763,785 |
| 2 | 473526.0 | 570465.0 | 2,10066,10069,10156 |
| 3 | 473504.0 | 570461.0 | 2,7135,9780,10064,10067 |
| 4 | 473510.0 | 570445.0 | 3,9779,10065,10067,10215,11404 |
| 5 | 473534.0 | 570437.0 | 3,10068,10070,10157 |
| 6 | 470546.0 | 572083.0 | 4,5,7 |
| 7 | 470542.0 | 572115.0 | 4,6,9886,9899 |
| 8 | 470577.0 | 571991.0 | 5,6047,6050 |
| 9 | 470546.0 | 572143.0 | 6,8256,8258 |

*Rysunek 1 - fragment pliku vertices.txt*

| id | from | to | road_id | length [m] | speed [m/s] |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 175.36713156142068 | 5.555555555555555 |
| 1 | 2 | 3 | 1 | 23.137294539213176 | 19.444444444444446 |
| 2 | 4 | 5 | 2 | 25.85698367048788 | 19.444444444444446 |
| 3 | 6 | 7 | 3 | 32.42771671805166 | 13.88888888888889 |
| 4 | 8 | 6 | 4 | 97.71947691250426 | 13.88888888888889 |
| 5 | 7 | 9 | 5 | 28.230190944329756 | 13.88888888888889 |
| 6 | 6 | 10 | 6 | 40.13102420621265 | 13.88888888888889 |
| 7 | 11 | 12 | 7 | 107.96179351445133 | 13.88888888888889 |
| 8 | 12 | 13 | 8 | 33.97962786197702 | 13.88888888888889 |
| 9 | 14 | 11 | 9 | 15.614803873242622 | 13.88888888888889 |

*Rysunek 2 - fragment pliku edges.txt*

- Implementacja algorytmu A*

  Pierwszym krokiem jest wyznaczenie heurystyki. Ze względu na 2 możliwe warianty wyznaczanych tras heurystyka jest inna dla obu z nich:

  - **dla trasy najkrótszej** - heurystyka to odległość euklidesowa między danym wierzchołkiem a wierzchołkiem końcowym (docelowym)
  - **dla trasy najszybszej -** heurystyka to czas pokonania odcinka (odległości euklidesowej) między danym wierzchołkiem a wierzchołkiem końcowym, przy założeniu, że samochód porusza się z maksymalną prędkością (zakładamy, że maksymalna prędkość to 140 km/h, obowiązująca na autostradzie)

  Algorytm wykorzystuje kolejkę priorytetową, która przechowuje wierzchołki do odwiedzenia, uporządkowane według wartości funkcji kosztu f(x). Algorytm pobiera wierzchołek o najniższym koszcie f(x):

  $$f(x) = g(x) + h(x)$$

  g(x) – koszt dotarcia do danego wierzchołka od wierzchołka początkowego
  h(x) – obliczona heurystyka (dostosowywana w zależności od wybranego wariantu trasy)

Następnie iteruje po sąsiadach tego wierzchołka, oblicza dla nich koszt, wybiera sąsiada o najniższym koszcie i dodaje go do ścieżki.

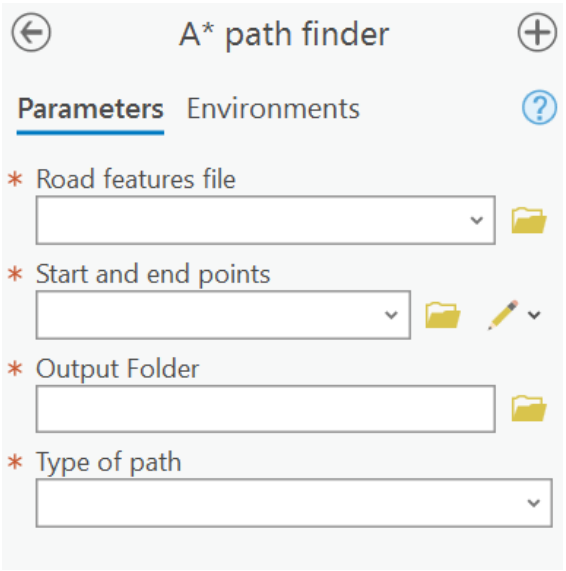
Algorytm dodatkowo przechowuje ilość przejrzanych wierzchołków oraz liczbę wierzchołków w kolejce priorytetowej, co pozwala ocenić szybkość jego działania.

- Uwzględnianie kierunkowości dróg
  Ze względu na brak dostępu do danych zawierających informacje o kierunkowości dróg, do warstwy SKJZ został dodany „sztuczny" atrybut mówiący o kierunkowości [Tabela 2]. Kilka przykładowych obiektów uznano za jednokierunkowe lub nieprzejezdne, a reszta pozostała bez ograniczeń.

*Tabela 2 - Wartości do oznaczania kierunkowości*

| Wartość atrybutu | Kierunkowość |
|---|---|
| 0 | Bez ograniczeń (dwukierunkowa) |
| 1 | Jednokierunkowa zgodnie z kierunkiem geometrii |
| 2 | Jednokierunkowa przeciwnie do kierunku geometrii |
| 3 | Nieprzejezdna |

- Integracja ze środowiskiem ESRI i implementacja rozszerzenia
  Jako rozszerzenie do projektu utworzony został Toolbox umożliwiający uruchomienie programu [Rysunek 3].




*Rysunek 3 - Toolbox w ArcGIS Pro*

Parametry wejściowe obejmują:
- **Road features file** – warstwa z siecią dróg SKJZ
- **Start and end point** – użytkownik podaje warstwę zawierającą dokładnie 2 punkty lub tworzy je na mapie, program posiada dodatkowe zabezpieczenie przed warstwami posiadającymi geometrię inną niż punktową oraz warstwami z więcej niż 2 punktami.
- **Output Folder**

- **Type of path** – użytkownik wybiera między dwoma wariantami – shortest path lub fastest path

## 3. URUCHOMIENIE PROGRAMU I WYNIKI

Toolbox uruchomiono dla przykładowych punktów w powiecie toruńskim. Na początku wyznaczono trasę najkrótszą, następnie najszybszą. Podczas wykonywania algorytmu zwracane są takie informacje, jak liczba wierzchołków w zbiorze S, liczba wszystkich odwiedzonych wierzchołków, całkowita długość trasy oraz czas jej przejazdu. Poniżej przedstawiono wyniki algorytmu dla obu rodzajów tras.

```
Vertex count: 31058
Edge count: 38559
Finding path...
Number of vertices in S set: 148
Total number of visited vertices: 2871
Path found.

--------------- Route ---------------
Total length of the route: 18km 528m
Total time of the route: 32min 23sec
Writing path to files...

Path added to display.
Succeeded at wtorek, 3 grudnia 2024
22:23:12 (Elapsed Time: 12,36 seconds)
```

*Rysunek 4 - Trasa najkrótsza*

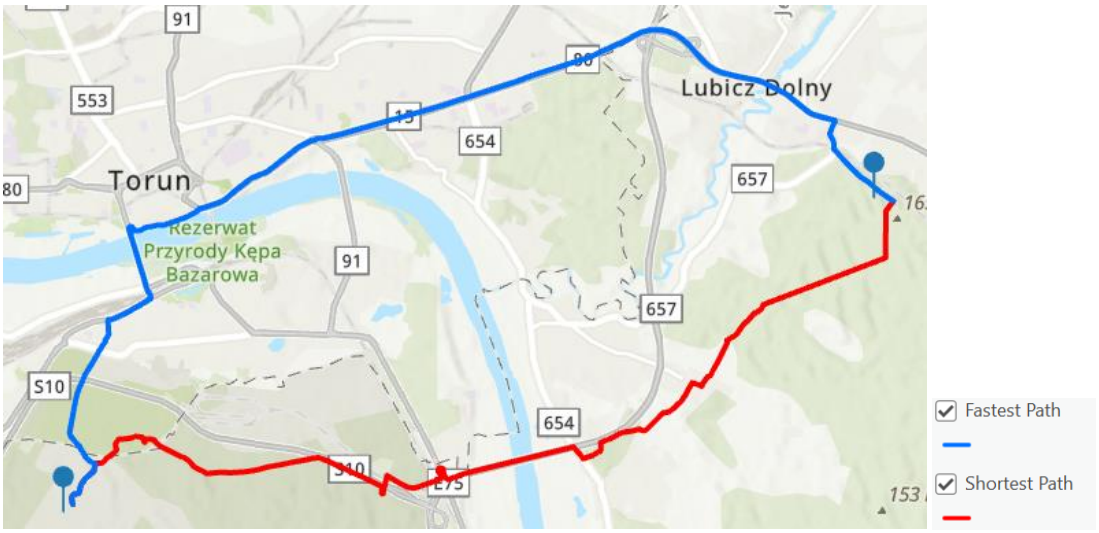
```
Vertex count: 31058
Edge count: 38559
Finding path...
Number of vertices in S set: 256
Total number of visited vertices: 15842
Path found.

--------------- Route ---------------
Total length of the route: 20km 898m
Total time of the route: 28min 40sec
Writing path to files...

Path added to display.
Succeeded at wtorek, 3 grudnia 2024
22:23:28 (Elapsed Time: 12,48 seconds)
```

*Rysunek 5 - Trasa najszybsza*

Po wykonaniu na mapie wyświetla się wyznaczona trasa. W wyjściowym folderze zapisują się pliki tekstowe: ***visited_vertices.txt*** i ***visited_roads.txt*** zawierające odwiedzone wierzchołki i krawędzie trasy.




*Rysunek 6 - Wizualizacja tras*

## 4. WNIOSKI

- Algorytm poprawnie wyznacza zarówno najkrótszą jak i najszybszą trasę. Zdarzają się przypadki, najczęściej przy niewielkiej odległości między punktem początkowym i końcowym, gdzie zwrócone przez algorytm trasy się pokrywają.

- Analizując wyniki dla przykładowego uruchomienia programu, można zauważyć, że w przypadku trasy najkrótszej odwiedzana jest mniejsza liczba wierzchołków. Nie wpływa to jednak na złożoność czasową, gdyż widzimy, że w obu przypadkach (dla trasy najszybszej i najkrótszej) algorytm wykonuje się w podobnym czasie (około 12 sekund).

- Wyznaczanie trasy najszybszej jest bardziej optymalne, ponieważ czas jej przejazdu potrafi być dużo krótszy. W wyznaczaniu tras i nawigacji w praktyce ważniejsze jest minimalizowanie czasu przejazdu, a nie jedynie długości trasy.

- Dzięki zastosowaniu podejścia heurystycznego oraz odpowiednich struktur danych (listy sąsiedztwa, kolejka priorytetowa) program jest wydajny i zapewnia stosunkowo szybkie znajdowanie tras nawet dla dużego zbioru danych.

- Mimo że sztucznie nadana kierunkowość niekoniecznie jest zgodna ze stanem rzeczywistym, udowadnia ona, że algorytm skutecznie obsługuje różne scenariusze ruchu drogowego, takie jak jednokierunkowość i ograniczenia przejazdu, które są powszechne szczególnie w obszarach miejskich.

## 5. BIBLIOGRAFIA


- https://pl.wikipedia.org/wiki/Algorytm_A*
- https://www.gov.pl/web/infrastruktura/rodzaje-drog-w-polsce
- https://pro.arcgis.com/en/pro-app/latest/arcpy/geoprocessing_and_python/defining-parameters-in-a-python-toolbox.htm
- https://pro.arcgis.com/en/pro-app/latest/tool-reference/data-management/xy-table-to-point.htm
- https://pro.arcgis.com/en/pro-app/3.3/tool-reference/analysis/near.htm
- https://pro.arcgis.com/en/pro-app/latest/arcpy/get-started/describing-data.htm