

Analizy sieciowe w QGIS

ALGORYTMY GRAS V.NET: ALLPAIRS, ISO, SALESMAN, STEINER;
WTYCZKA QNEAT3 I CHINESE POSTMAN SOLVER

wer. 2020.05.11

Paweł Zmuda-Trzebiatowski

DOKUMENT ROZPOWSZECHNIANY NA LICENCJI CC BY-SA 3.0



Spis treści

Wstęp	2
1. Narzędzia processingu: wektor – analiza sieciowa (QGIS 3.x)	2
1.1. Obliczanie najkrótszej ścieżki	2
1.2. Wyznaczanie obszaru obsługi	5
2. Wtyczka QNEAT3 (QGIS 3.x).....	8
2.1. Generowanie macierzy „źródło-cel” podróży	8
2.2. Generowanie Izolinii (Iso-areas)	12
3. Algorytmy sieciowe GRASS – V.NET (QGIS 2.18 oraz 3.X).....	15
3.1. v.net.allpairs.....	15
3.2. v.net.iso.....	17
3.3. v.net.salesman	18
3.4. v.net.steiner	19
4. Inspekcja sieci z wtyczką Chinese Postman Solver.....	20

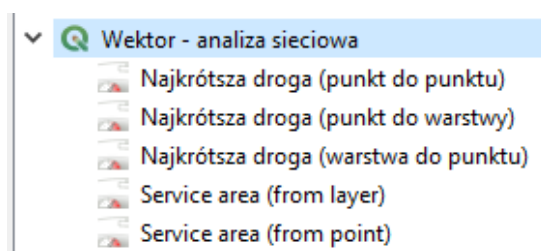
WSTĘP

W niniejszym samouczku opisano wybrane analizy sieciowe, które są dostępne w QGISie natywnie, przez algorytmy GRASS oraz wtyczki QNEAT3 i Chinese Postman Solver. Analizy sieciowe różnią się od opisanych w innych częściach samouczka analiz przestrzennych tym, że odległości mierzone są w metryce miejskiej na sieci drogowej, a nie w metryce euklidesowej (w linii prostej). Stąd kluczowym ich elementem są liniowe warstwy reprezentujące tę sieć. Wczytaj warstwę *drogi_poz92.shp*, która zawiera informacje o głównych ulicach zlokalizowanych w obrębie miasta Poznań. Ponadto do wykonania części ćwiczeń będziesz potrzebować warstwy punktowej – *straz_pozarna.shp*. Warstwa ta zawiera lokalizacje Jednostek Ratunkowo-Gaśniczych Państwowej Straży Pożarnej oraz jednej jednostki Ochotniczej Straży Pożarnej, które zlokalizowane są w Poznaniu. Obie warstwy zostały wyodrębnione z danych projektu OpenStreetMap. Do celów poglądowych możesz wczytać także warstwę *powiatMPO-znanCS92.shp*, która określa granice administracyjne Poznania i została wyodrębniona z danych Państwowego Rejestru Granic. Wszystkie warstwy zostały zapisane z kodowaniem UTF-8 w układzie współrzędnych EPSG:2180.

1. NARZĘDZIA PROCESSINGU: WEKTOR – ANALIZA SIECIOWA (QGIS 3.X)

Uwaga: narzędzia opisane w tym rozdziale są niedostępne w QGISie 2.18. Tamta wersja programu zawierała nieco prostsze wersje narzędzie *RoadGraph*, które zostało opisane na stronach 90-93 poświęconego jej samouczka.

W QGISie 3 udostępnia natywnie zestaw algorytmów sieciowych, które pozwalają obliczać najkrótszą ścieżkę oraz obszar obsługi (*service area*). Narzędzia te dostępne są przez panel processingu w grupie *Wektor – analiza sieciowa*.



1.1. Obliczanie najkrótszej ścieżki

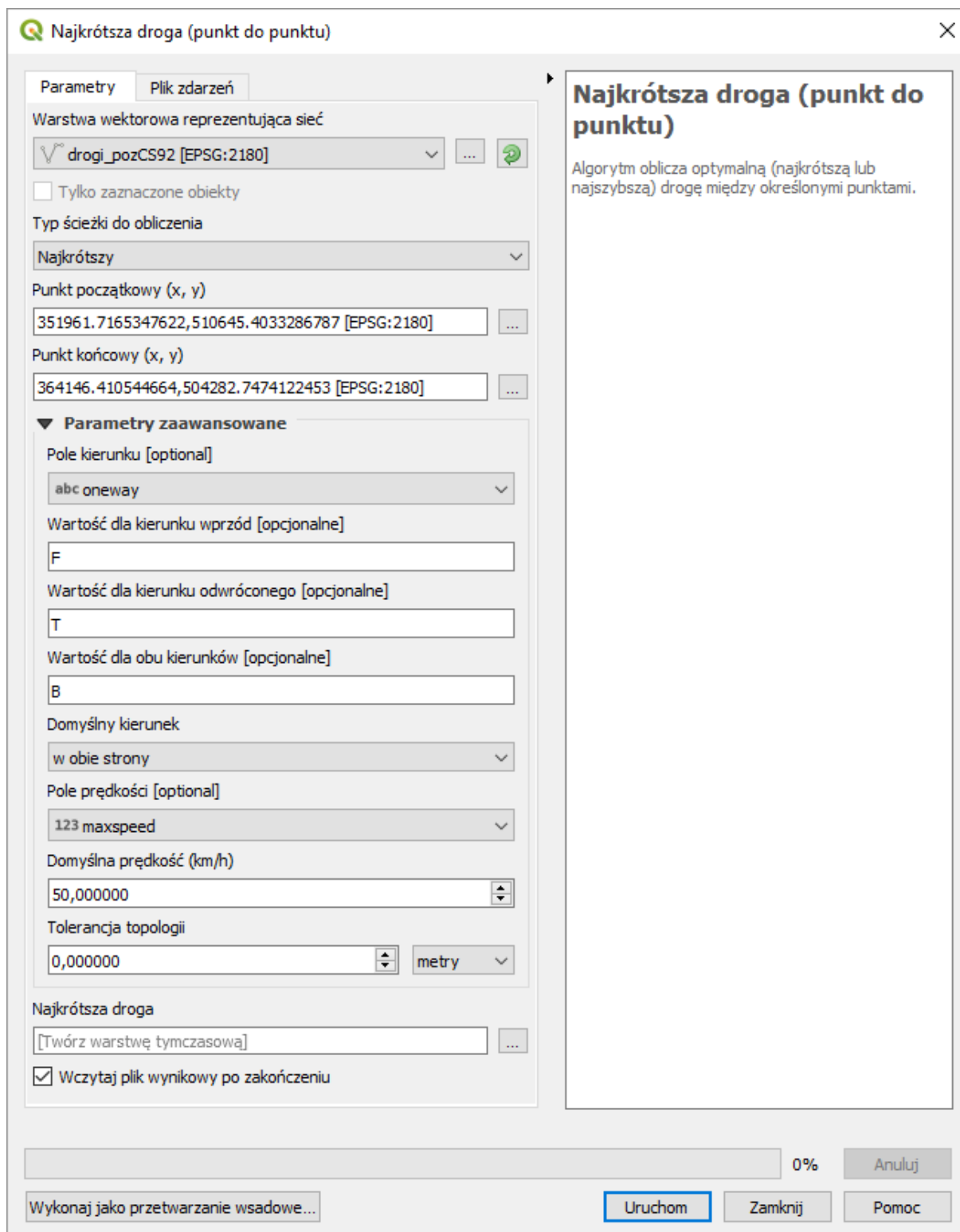
QGIS pozwala obliczyć najkrótszą ścieżkę dla trzech rodzajów danych:

- pomiędzy dwoma wskazanymi punktami (*punkt do punktu*)
- pomiędzy wskazanym punktem początkowym, a wszystkimi punktami docelowymi określonymi w warstwie (*punkt do warstwy*)
- pomiędzy wszystkimi punktami początkowymi wskazanymi w warstwie, a punktem docelowym (*warstwa do punktu*).

Spróbujmy znaleźć najkrótszą ścieżkę pomiędzy dwoma wskazanymi punktami. Wybierz algorytm *Najkrótsza droga (punkt do punktu)*. W oknie dialogowym algorytmu należy ustawić kolejno:

- *Warstwa wektorowa reprezentująca sieć* – wybierz *drogi_pozCS92*
- *Typ ścieżki do obliczenia*, do wyboru są *najkrótszy* (odległość) lub *najszybszy* (czas).
- *Punkt początkowy (x, y)* oraz *punkt końcowy (x, y)* – można ręcznie wpisać lub zaznaczyć myszą po kliknięciu przycisków [...] znajdujących się po prawej stronie pól. Wybierz drugi sposób. Zaznacz dwa punkty zlokalizowane gdzieś w obrębie Poznania. Nie musisz zaznaczać punktów znajdujących się

dokładnie na warstwie uliv. Algorytm automatycznie dociągnie punkt do najbliższej (w linii prostej). Obszar poszukiwań można powiększyć zmieniając parametr *Tolerancja topologii* (pozostaw 0).



Aby opcja *Typ ścieżki do obliczenia* rozróżniała ścieżki najkrótsze pod względem odległościowym od czasowego należy w parametrach zaawansowanych określić, który atrybut informuje o prędkości (*Pole prędkości*). W przypadku danych pobranych z serwisu geofabrik.de jest to „maxspeed”. W polu *domyślna prędkość* można określić wartość, która QGIS przyjmie w przypadku braku informacji w tabeli atrybutów. Można pozostawić 50km/h.

Ponadto w parametrach zaawansowanych można wskazać, który atrybut informuje o kierunkowości ulicy. W naszym przypadku jest to „*oneway*”. Kierunek „wprzód” (od początku linii do końca) jest w danych geofabrik oznaczany literą „F”. Kierunek odwrócony (stosowany np. do opisanego odcinków sieci, na których obowiązuje tymczasowa organizacja ruchu przeciwna do standardowej) jest oznaczany literą „T”, natomiast ruch dwukierunkowy „B”. W kolejnym polu można wskazać domyślny kierunek ruchu na wypadek braku danych w tabeli atrybutów (w obie strony). W przypadku danych wykorzystywanych w tym ćwiczeniu uwzględnienie kierunkowości najłatwiej jest zauważyć dla dróg dwujezdniowych.

Po uruchomieniu algorytmu zostanie utworzona nowa warstwa reprezentująca najkrótszą ścieżkę, względnie zwrócona zostanie informacja, że taka ścieżka nie istnieje. Jej tabela atrybutów będzie zawierać informacje o współrzędnych punktu początkowego i końcowego oraz wyznaczonym koszcie przejazdu.



Efekt działania algorytmu Najkrótsza ścieżka (punkt do punktu)

W analogiczny sposób spróbuj wyznaczyć najkrótsze trasy dojazdu straży pożarnej do wybranego punktu w centrum. Wykorzystaj algorytm *Najkrótsza ścieżka (warstwa do punktu)*. Zauważ, że jednostki straży pożarnej także nie muszą znajdować się na ulicach. Ma to miejsce w tym ćwiczeniu, gdyż warstwa ulic nie zawiera informacji o drogach dojazdowych.



Efekt działania algorytmu Najkrótsza ścieżka (warstwa do punktu)

1.2. Wyznaczanie obszaru obsługi

Natywne algorytmy sieciowe QGISa pozwalają wyznaczyć także tzw. obszary obsługi (*service area*), tj. wszystkie drogi osiągalne z punktów startowych w określonym czasie lub limicie odległości. Algorytm podobnie do najkrótszej ścieżki występuje w dwóch wariantach:

- Wyznaczanie obszarów obsługi dla wszystkich punktów zapisanych w warstwie – *service area (from layer)*
- Wyznaczanie obszarów obsługi dla wybranego punktu – *service area (from point)*.

Okno dialogowe algorytmu pozwala na ustalenie zbliżonych parametrów do algorytmów obliczających najkrótszą ścieżkę. Dodatkowa opcja *Koszt podróży* pozwala na określenie maksymalnego dopuszczalnego kosztu podróży. Jest on wyrażony w metrach lub sekundach.

Wynik może być przedstawiony zarówno jako zbiór linii (*Service area (lines)*), tj. ulic objętych zasięgiem, jak i punktów (*Service area (boundary nodes)*) – w tym przypadku zaznaczane są tylko węzły graniczne danych odcinków sieci. Domyślnie generowany jest zbiór linii.

Service area (from layer)
✕

Parametry
Plik zdarzeń

Warstwa wektorowa reprezentująca sieć

📍 drogi_pozCS92 [EPSG:2180]
⌵
⋮
🔄

Tylko zaznaczone obiekty

Warstwa wektorowa z punktami początkowymi

📍 straz_pozarna [EPSG:2180]
⌵
⋮
🔄

Tylko zaznaczone obiekty

Typ ścieżki do obliczenia

Najkrótszy
⌵

Koszt podróży (odległość dla "Najkrótszy", czas dla "Najszybszy")

2500,000000
✕
⌵

▼ Parametry zaawansowane

Pole kierunku [optional]

abc oneway
⌵

Wartość dla kierunku drogi [opcjonalne]

F

Wartość dla kierunku odwróconego [opcjonalne]

T

Wartość dla obu kierunków [opcjonalne]

B

Domyślny kierunek

w obie strony
⌵

Pole prędkości [optional]

123 maxspeed
⌵

Domyślna prędkość (km/h)

50,000000
✕
⌵

Tolerancja topologii

0,000000
⌵
metry
⌵

Include upper/lower bound points

Service area (lines)

[Twórz warstwę tymczasową]
⋮

Wczytaj plik wynikowy po zakończeniu

Service area (boundary nodes)

[Pomiń dane wyjściowe]
⋮

Wczytaj plik wynikowy po zakończeniu

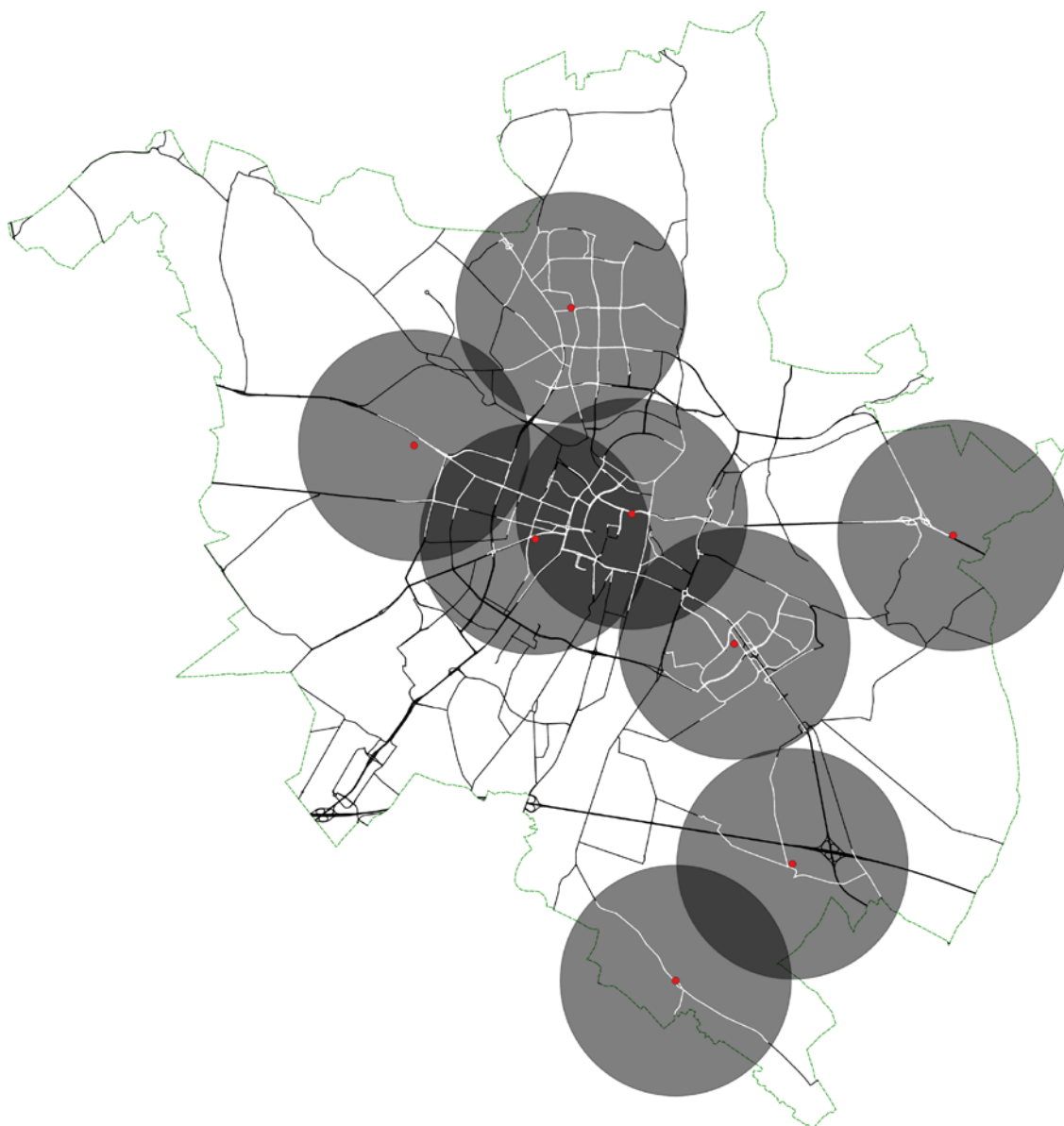
0%
Anuluj

Wykonaj jako przetwarzanie wsadowe...
Uruchom
Zamknij
Pomoc

Poleceniem *Service area (from layer)* wyznacz zasięg straży pożarnej dla czasu dojazdu wynoszącego 300 sekund (5 minut) oraz dla odległości wynoszącej 2500 metrów. Załóż, że przepisy są przestrzegane. Aby poprawnie wykonać zadanie dla czasu dojazdu musisz usunąć ograniczenia wynoszące 0 km/h dla niektórych segmentów dróg, tzn. zamienić wartość atrybutu „maxspeed” z 0 na NULL¹. Wyniki przedstaw jako linie.

Przyglądając się wynikom można zauważyć, że obszary obsługi mogą częściowo na siebie nachodzić. Jednak algorytm dodaje w tabeli atrybutów pole „start”, które pozwala np. na zróżnicowanie koloru ulic będących w zasięgu poszczególnych jednostek straży albo wyodrębnienie ich do nowych warstw.

Dla porównania możesz też utworzyć bufory wskazujące na zasięgi, które byłyby osiągalne, jeśli pominąć sieć drogową.



¹ Uruchom kalkulator pól w tabeli atrybutów i przelicz pole „maxspeed” poleceniem: `if ("maxspeed" = 0, NULL, "maxspeed")` Więcej o kalkulatorze pól przeczytasz w podstawowym module samouczka.

2. WTYCZKA QNEAT3 (QGIS 3.X)

Analizy sieciowe w QGIS 3 mogą być rozwinięte przez instalację wtyczki QNEAT3. Wtyczka udostępnia trzy podstawowe rodzaje algorytmów:

- Generowanie macierzy typu źródło-cel podróży (*Distance Matrices*)
- Generowanie Izolinii (*Iso-areas*)
- Znajdowanie najkrótszej ścieżki od punktu do punktu (*Routing*)

2.1. Generowanie macierzy „źródło-cel” podróży

Macierze typu „źródło-cel” podróży (*OD-Matrix*) w przypadku QNEAT3 wskazują odległości pomiędzy każdą parą punktów źródłowych i docelowych. Algorytm jest zatem odpowiednikiem algorytmu przestrzennego *Macierz odległości* (menu [→Wektor→Narzędzia analizy]), przy czym odległości liczone są na sieci drogowej, a nie w linii prostej. Istnieją dwie wersje tego algorytmu:

- dla jednej warstwy punktowej – *OD-Matrix from Points*
- dla niezależnych warstw źródeł i celów podróży – *OD Matrix from Layers*

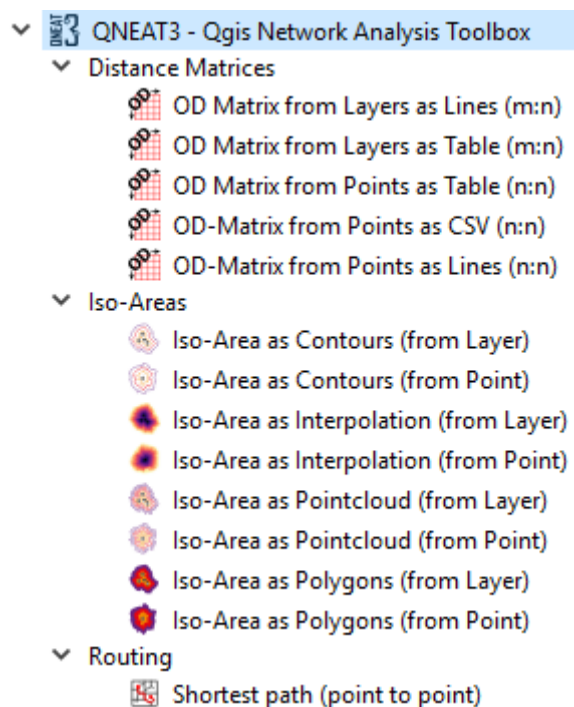
Ponadto dla każdej z tych wersji można wybrać opcję generowania wyników, jako linii (*as Lines*) lub tabeli (*as Table*). W przypadku wykorzystywania tylko jednej warstwy punktowej można też wygenerować od razu warstwę CSV (*as CSV*).

Skorzystaj z tej ostatniej opcji i wygeneruj macierz odległości pomiędzy jednostkami straży pożarnej.

Okno dialogowe algorytmu zawiera zbliżone parametry do algorytmów opisanych wcześniej. Nie zostały one jednak przetłumaczone na język polski (w nawiasach podano oczekiwane w tym ćwiczeniu wartości):

- *Network Layer* – warstwa sieci drogowej (drogi_pozCS92.shp)
- *Point Layer* – warstwa punktowa (straz_pozarna.shp)
- *Unique Point ID Field* – unikalne pole ID – po nim będą rozpoznawane łączone pary punktów (full.id)
- *Optimization Cirterion* (kryterium optymalizacji) – odległość (*shortest*) lub czas (*fastest*)
- *Direction Field* – pole kierunku (oneway)
- *Value for*:
 - *Forward direction* – kierunek drogi “wprost” (F)
 - *Backward direction* – kierunek odwrócony (T)
 - *Both directions* – obydwa kierunków (B)
- *Default direction* – domyślny kierunek (Both directions)
- *Speed field* – pole prędkości (maxspeed)
- *Default speed* – domyślna prędkość (50km/h)
- *Topology tolerance* – tolerancja topologii (0)
- *Output OD matrix* - wyjściowa macierz OD (kliknij [...] i wpisz nazwę pliku w oczekiwanym folderze)

Po kliknięciu [Uruchom] zostanie utworzony plik CSV.



OD-Matrix from Points as CSV (n:n)
✕

Parametry
Plik zdarzeń

Network Layer

▼ drogi_pozCS92 [EPSG:2180]
...
🔄

Tylko zaznaczone obiekty

Point Layer

◦ straz_pozarna [EPSG:2180]
...
🔄

Tylko zaznaczone obiekty

Unique Point ID Field

abc full_id
▼

Optimization Criterion

Shortest
▼

▼ **Parametry zaawansowane**

Direction field [optional]

abc oneway
▼

Value for forward direction [opcjonalne]

F

Value for backward direction [opcjonalne]

T

Value for both directions [opcjonalne]

B

Default direction

Both directions
▼

Speed field [optional]

123 maxspeed
▼

Default speed (km/h)

50,000000
✕
▲▼

Topology tolerance

0,000000
▲▼

Output OD Matrix

C:/Users/Pawel Zmuda/Desktop/aaa/a.csv
...

OD-Matrix from Points as CSV (n:n)

General:
 This algorithm implements OD-Matrix analysis to return the **matrix of origin-destination pairs as csv-file yielding network based costs on a given network dataset between the elements of one point layer(n:n)**. It accounts for **points outside of the network** (eg. *non-network-elements*). Distances are measured accounting for **ellipsoids**, entry-, exit-, network- and total costs are listed in the result attribute-table.

Parameters (required):
 Following Parameters must be set to run the algorithm:

- Network Layer
- Point Layer
- Unique Point ID Field (numerical)
- Cost Strategy

Parameters (optional):
 There are also a number of *optional parameters* to implement **direction dependent** shortest paths and provide information on **speeds** on the networks edges.

- Direction Field
- Value for forward direction
- Value for backward direction
- Value for both directions
- Default direction
- Speed Field
- Default Speed (affects entry/exit costs)
- Topology tolerance

Output:
 The output of the algorithm is one file:

- OD-Matrix as csv-file with network based distances as attributes

0%
Anuluj

Wykonaj jako przetwarzanie wsadowe...
Uruchom
Zamknij

	A	B	C	D	E	F	G
1	origin_id	destination_id	entry_cost	network_cost	exit_cost	total_cost	
2	w28647661	w28647661	0.0				
3	w28647661	w89133916	190.28560683872965	5935.347030950646	32.00745797576367	6157.640095765139	
4	w28647661	w165179800	190.28560683872965	6148.4973962973145	41.34534838655919	6380.128351522604	
5	w28647661	w210713758	190.28560683872965	17147.02336563103	16.086493044511432	17353.395465514273	

9

Tabela wynikowa składa się z 6 kolumn. Dwie pierwsze stanowią informacje o rozpatrywanym punkcie początkowym i końcowym (*origin_id* i *destination_id*). Następne cztery kolumny określają koszt: wjazdu na sieć (*entry_cost*), koszt przemieszczania się po sieci (*network_cost*), koszt zjazdu (*exit cost*) oraz koszt całkowity będący sumą poprzednich (*total_cost*).

Teraz spróbuj wykonać to samo zadanie, ale rysując między punktami linie. Skorzystaj z algorytmu *OD Matrix from Layers as Lines*. Poniżej zobaczysz efekt działania algorytmu.



OD Matrix from Layers as Lines (m:n)
✕

Parametry
Plik zdarzeń

Network Layer

▼
drogi_pozCS92 [EPSG:2180]
...
↻

Tylko zaznaczone obiekty

From-Point Layer

◦ ◦
straz_pozarna [EPSG:2180]
...
↻

Tylko zaznaczone obiekty

Unique Point ID Field

▼
abc full_jd

To-Point Layer

◦ ◦
straz_pozarna [EPSG:2180]
...
↻

Tylko zaznaczone obiekty

Unique Point ID Field

▼
abc full_jd

Optimization Criterion

▼
Shortest

▼ Parametry zaawansowane

Direction field [optional]

▼
abc oneway

Value for forward direction [opcjonalne]

F

Value for backward direction [opcjonalne]

T

Value for both directions [opcjonalne]

B

Default direction

▼
Both directions

Speed field [optional]

▼
123 maxspeed

Default speed (km/h)

50,000000
✕
↕

Topology tolerance

0,000000
↕

OD Matrix from Layers as Lines (m:n)

General:
 This algorithm implements OD-Matrix analysis to return the **matrix of origin-destination pairs as lines yielding network based costs** on a given **network dataset between two layer of points (m:n)**.
 It accounts for **points outside of the network** (eg. *non-network-elements*). Distances are measured accounting for **ellipsoids**, entry-, exit-, network- and total costs are listed in the result attribute-table.

Parameters (required):
 Following Parameters must be set to run the algorithm:

- Network Layer
- From-Point Layer
- Unique From-Point ID Field (numerical)
- To-Point Layer
- Unique To-Point ID Field (numerical)
- Cost Strategy

Parameters (optional):
 There are also a number of *optional parameters* to implement **direction dependent** shortest paths and provide information on **speeds** on the networks edges.

- Direction Field
- Value for forward direction
- Value for backward direction
- Value for both directions
- Default direction
- Speed Field
- Default Speed (affects entry/exit costs)
- Topology tolerance

Output:
 The output of the algorithm is one layer:

- OD-Matrix as lines with network based distances as attributes

0%

Anuluj

Wykonaj jako przetwarzanie wsadowe...

Uruchom

Zamknij

2.2. Generowanie Izolinii (Iso-areas)

QNEAT3 udostępnia 8 wariantów algorytmu generującego izolinie. Podstawowy podział obejmuje generowanie izolinii dla jednego wskazanego punktu (*from Point*) lub dla punktów z wybranej warstwy punktowej (*from Layer*). Dla każdej z tych opcji dostępne są cztery warianty algorytmu, które różnią się sposobem reprezentacji wyniku. Może być to:

- Warstwa liniowa konturów – *as Contours*
- Warstwa poligonowa – *as Polygons*
- Warstwa punktowa (chmura punktów) – *as Pointcloud*
- Warstwa rastrowa / intropolacja – *as Interpolation*

Sposób generowania wyników jest zbliżony do wykonywanych w poprzednich ćwiczeniach. Wygenerujmy izolinie, jako poligony wskazujące odległość od jednostek straży pożarnej. Ustalmy maksymalną odległość na 2,5 km, a kolejne izolinie generujmy co 500 metrów. Wykorzystaj algorytm *Iso-Area as Polygons (from Layer)*. W oknie dialogowym algorytmu dostępne są następujące parametry:

- *Vector Layer representing network* – warstwa sieci drogowej (drogi_pozCS92.shp)
- *Starts Points* – warstwa punktowa, z której generowane są izolinie (straz_pozarna.shp)
- *Unique Point ID Field* – unikalne pole ID – po nim będą rozpoznawane izolinie (full.id)
- *Size of Iso-areas* – maksymalna wielkość odległości lub czasu izolinii (2500 metrów)
- *Contour Interval* – odległość wstawiania kolejnych izolinii (500m)
- *Cellsize of interpolation Raster* – rozmiar komórki (piksela) wynikowego pliku rastrowego (50m)
- *Path type to calculate (rodzaj izolinii)* – odległość (*shortest*) lub czas (*fastest*)
- *Direction Field* – pole kierunku (oneway)
- *Value for:*
 - *Forward direction* – kierunek drogi (F)
 - *Backward direction* – kierunek odwrócony (T)
 - *Both directions* – obydwu kierunków (B)
- *Default direction* – domyślny kierunek (Both directions)
- *Speed field* – pole prędkości (maxspeed)
- *Default speed* – domyślna prędkość (50km/h)
- *Topology tolerance* – tolerancja topologii (0)
- *Output Interpolation* – wyjściowy plik rastrowy (pozostaw warstwę tymczasową)
- *Output Polygon* – wyjściowy plik rastrowy (pozostaw warstwę tymczasową)

Q Iso-Area as Polygons (from Layer)
✕

Parametry
Plik zdarzeń

Vector layer representing network

▼ drogi_pozCS92 [EPSG:2180]
...
↻

Tylko zaznaczone obiekty

Start Points

◉ straz_pozarna [EPSG:2180]
...
↻

Tylko zaznaczone obiekty

Unique Point ID Field

abc full_id
▼

Size of Iso-Area (distance or time value)

2500,000000
↕

Contour Interval (distance or time value)

500,000000
↕

Cellsize of interpolation raster

50
✕
↕

Path type to calculate

Shortest
▼

▼ Parametry zaawansowane

Direction field [optional]

abc oneway
▼

Value for forward direction [opcjonalne]

F

Value for backward direction [opcjonalne]

T

Value for both directions [opcjonalne]

B

Default direction

Both directions
▼

Speed field [optional]

123 maxspeed
▼

Default speed (km/h)

50,000000
✕
↕

Topology tolerance

0,000000
↕

Output Interpolation

[Zapisz w pliku tymczasowym]
...

Wczytaj plik wynikowy po zakończeniu

Output Polygon

[Twórz warstwę tymczasową]
...

Wczytaj plik wynikowy po zakończeniu

Wykonaj jako przetwarzanie wsadowe...
0%
Anuluj

Uruchom
Zamknij

Iso-Area as Polygons (from Layer)

General:
 This algorithm implements iso-area analysis to return the **iso-area polygons for a maximum cost level and interval levels** on a given **network dataset for a layer of points**. It accounts for **points outside of the network** (eg. *non-network-elements*) and increments the iso-areas cost regarding to distance/default speed value. Distances are measured accounting for **ellipsoids**. Please, **only use a projected coordinate system (eg. no WGS84)** for this kind of analysis.

Parameters (required):
 Following Parameters must be set to run the algorithm:

- Network Layer
- Startpoint Layer
- Unique Point ID Field (numerical)
- Maximum cost level for Iso-Area
- Cost Intervals for Iso-Area Bands
- Cellsize in Meters (increase default when analyzing larger networks)
- Cost Strategy

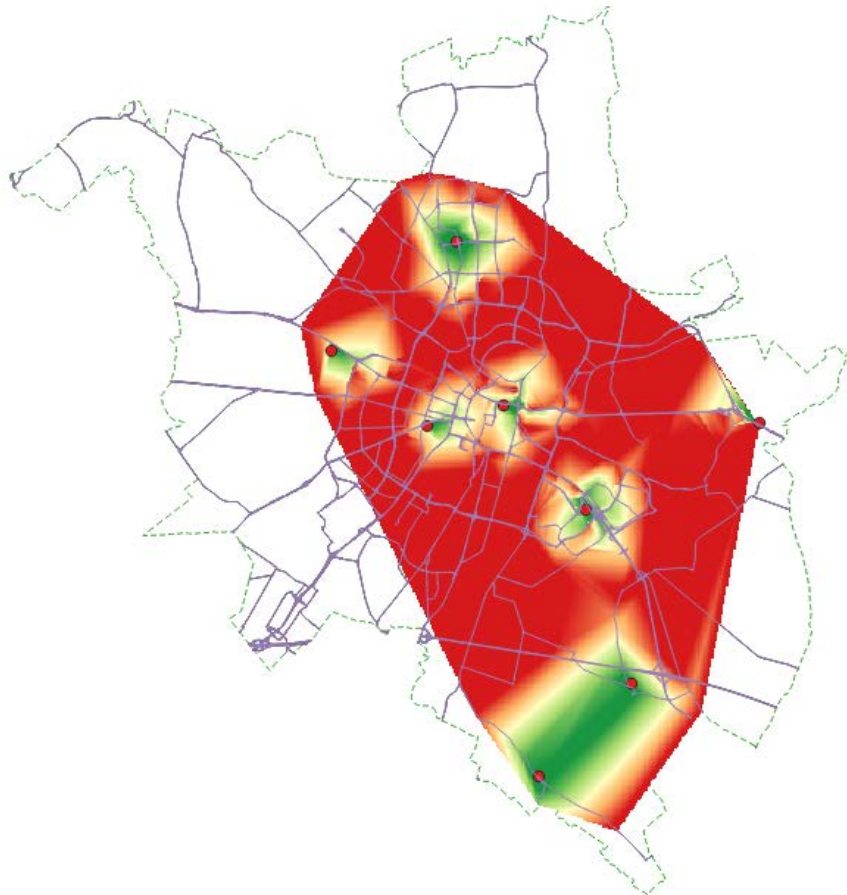
Parameters (optional):
 There are also a number of *optional parameters* to implement **direction dependent** shortest paths and provide information on **speeds** on the networks edges.

- Direction Field
- Value for forward direction
- Value for backward direction
- Value for both directions
- Default direction
- Speed Field
- Default Speed (affects entry/exit costs)
- Topology tolerance

Output:
 The output of the algorithm are two layers:

- TIN-Interpolation Distance Raster
- Iso-Area Polygons with cost levels as attributes

13





















3. ALGORYTMY SIECIOWE GRASS – V.NET (QGIS 2.18 ORAZ 3.X)

W wyniku błędu QGISa algorytmy GRASS NIE działają w wersji 3.4.0 i 3.4.1 pod Windowsem zlokalizowanym dla polskich ustawień językowych. Działają za to w wersji 3.4.2, 3.2.3 oraz 2.18.25.

QGIS dysponuje też częścią algorytmów sieciowych GRASS. Algorytmy te znajdują się w panelu processingu w grupie *GRASS* → *wektor [v.]* i zaczynają od „v.net.”. Przykładowo są to:

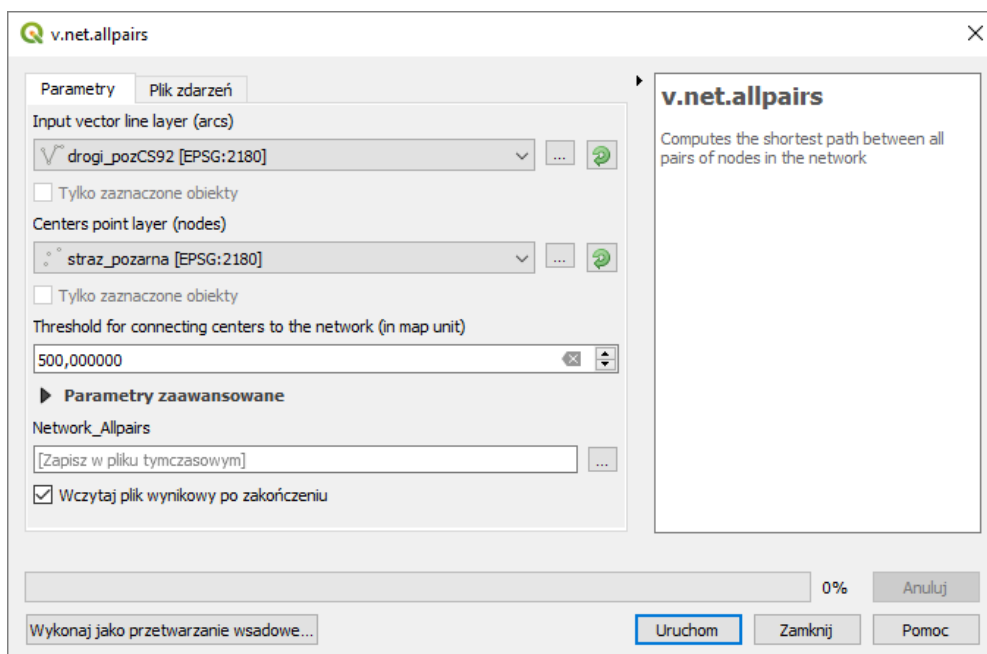
- *v.net.allpairs* – generuje macierz odległości między wszystkimi parami punktów danej warstwy wektorowej
- *v.net.distance* – jak wyżej, tylko dla dwóch warstw wektorowych – początkowej i końcowej
- *v.net.path* – wyznacza najkrótsze ścieżki pomiędzy wybranymi punktami
- *v.net.iso* – wyznacza izoliny dla danej sieci drogowej
- *v.net.timetable* – wyznacza najszybszą ścieżkę na podstawie rozkładów jazdy
- *v.net.salesman* – rozwiązuje problem komiwojażera dla wybranej warstwy punktowej, to jest problem znalezienia najkrótszej ścieżki, która połączy w cykl wszystkie punkty na wybranej warstwie punktowej
- *v.net.steiner* – stara się odnaleźć drzewo Steinera, to jest najkrótszą ścieżkę, która łączy wszystkie punkty ze sobą.

-  v.net
-  v.net.alloc
-  v.net.allpairs
-  v.net.bridge
-  v.net.centrality
-  v.net.components
-  v.net.connectivity
-  v.net.distance
-  v.net.flow
-  v.net.iso
-  v.net.nreport
-  v.net.path
-  v.net.report
-  v.net.salesman
-  v.net.spanningtree
-  v.net.steiner
-  v.net.timetable
-  v.net.visibility

Algorytmy GRASS wymagają innego sposobu informowania o kierunkowości i maksymalnej dopuszczalnej prędkości na ulicach niż algorytmy opisane wcześniej. Powinny być to atrybuty w formie liczbowej, która reprezentuje koszt przebycia danego odcinka drogi. Jako, że nasze dane nie uwzględniają takich informacji, to algorytmy nie będą ich uwzględniać w rozwiązywanych poniżej przykładach.

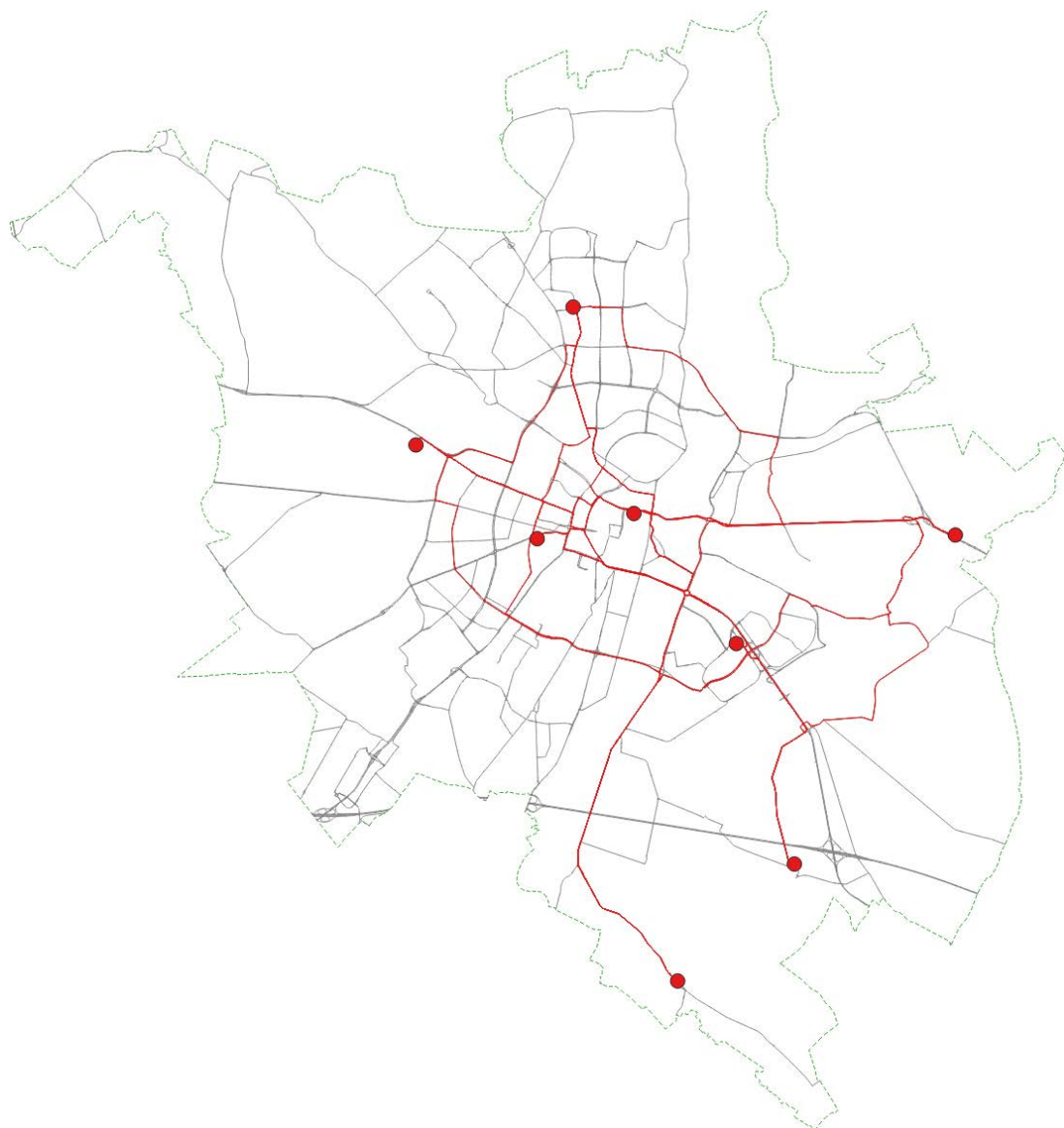
3.1. v.net.allpairs

Wyszukajmy najkrótsze ścieżki pomiędzy wszystkimi parami jednostek straży pożarnej w Poznaniu. Algorytm *v.net.allpairs* od narzędzi QNEAT3 odróżnia go to, że w liniowa warstwa wynikowa łączy punkty wykorzystując odcinki sieci. Wartość kosztu jest dopisywana do tabeli atrybutów za ostatnią kolumną.



Algorytm do działania wymaga zdefiniowania czterech parametrów:

- *Input vector line layer* – warstwa dróg (drogi_pozCS92.shp)
- *Centers point layer* – warstwa punktowa (straz_pozarna.shp)
- *Threshold for connecting centers to the network* – maksymalna odległość, na jaką można podłączyć punkty do ulic (500 jednostek mapy, tj. metrów w EPSG:2180)
- *Network_Allpairs* – warstwa wynikowa (Zapisz w pliku tymczasowym)

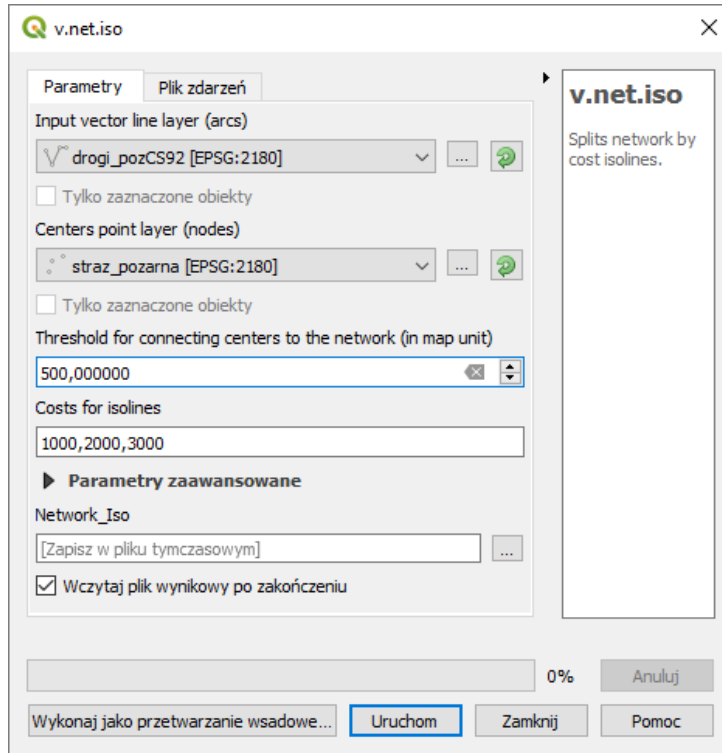


Informacja o koszcie (odległości) podróży została zapisana w tabeli atrybutów (*cost*).

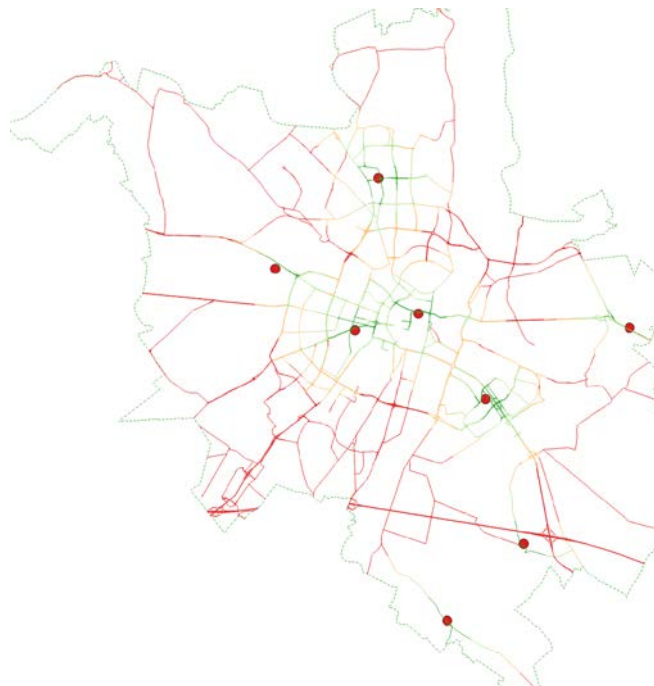
Network_Allpairs :: Liczba obiektów: 4900, odfiltrowanych: 4900, zaznaczonych: 0					
	fid	cat	from_cat	to_cat	cost
1	3165	1	1	2	5656,466
2	3254	1	1	2	5656,466

3.2. v.net.iso

Algorytm v.net.iso wymaga podania jednego dodatkowego parametru – *Cost for isolines*. W tym miejscu należy wskazać kolejne wartości izolinii, które ma się stworzyć, oddzielając je od siebie przecinkami. Pozostaw domyślną wartość „1000,2000,3000”, a pozostałe parametry ustaw tak jak w przypadku algorytmu v.net.allpairs.

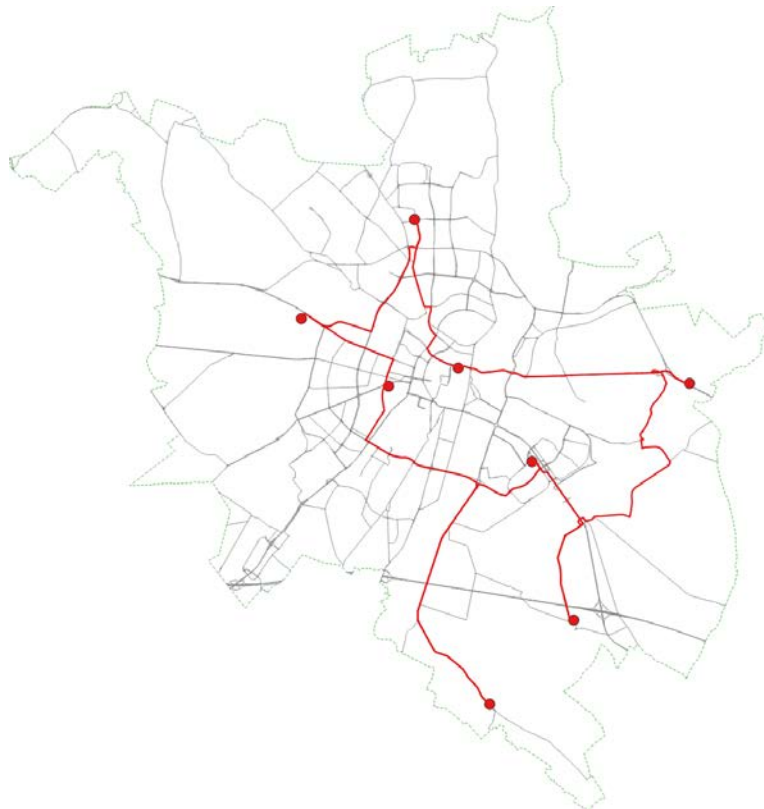
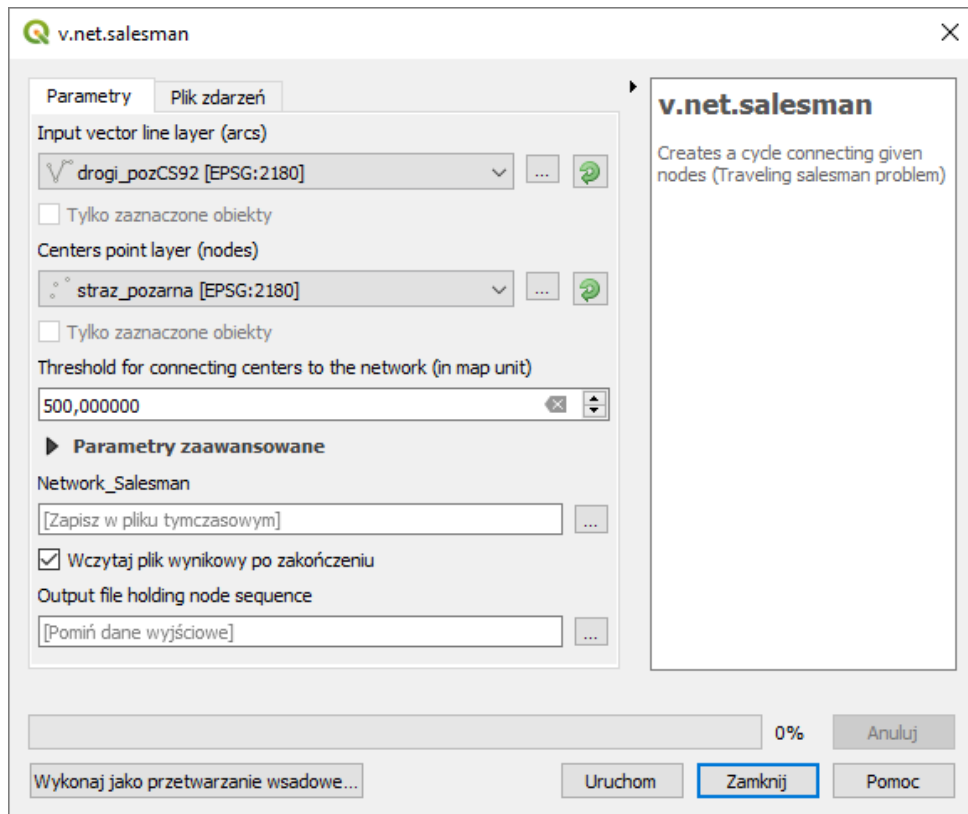


Po uruchomieniu algorytmu utworzona zostanie nowa warstwa. W jej tabeli atrybutów znajduje się *cat*, który określa przydział danego obiektu (łuku drogi) do określonej kategorii: 1 – od 0 do 1000 (metrów), 2 – od 1000 do 2000, 3 – od 2000 do 3000, 4 – od 3000 w górę. Możesz wykorzystać ten atrybut do wystylizowania tej warstwy wektorowej przez wartość unikalną.



3.3. v.net.salesman

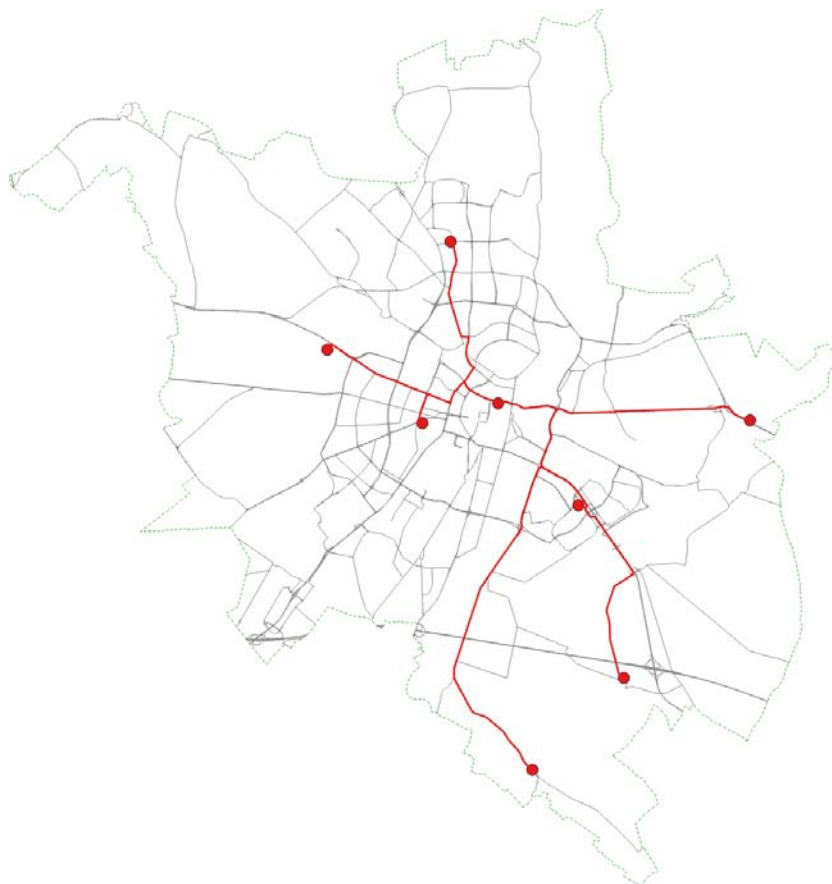
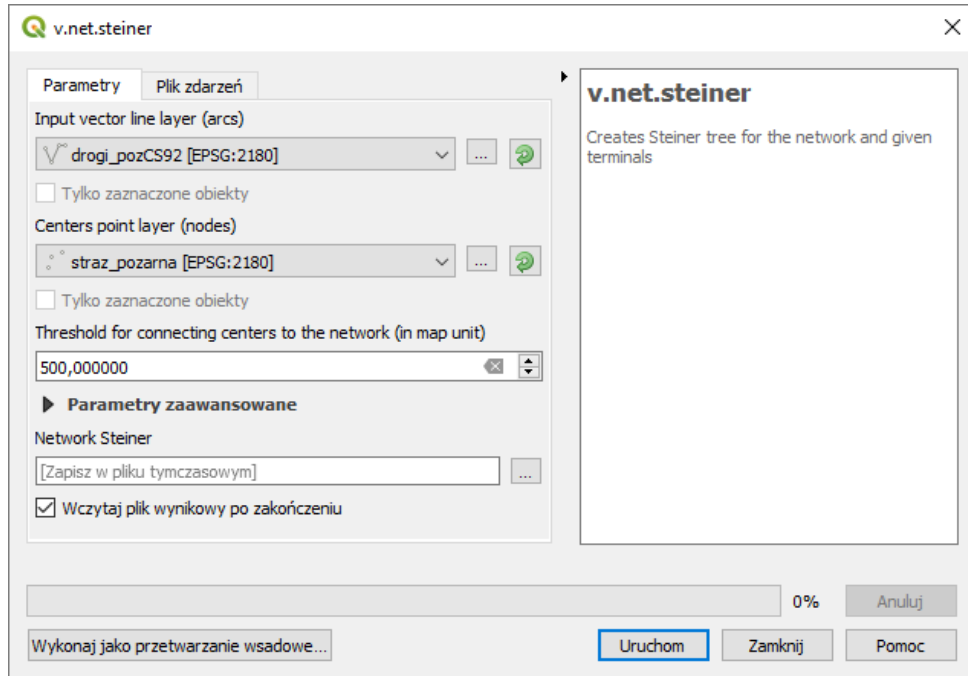
Algorytm rozwiązujący problem komiwojażera wymaga takich samych parametrów, co v.net.allpairs.



Uwaga: Przez wzgląd na charakter problemów, zaproponowane przez QGISa rozwiązania problemów komiwojażera oraz drzewa Steinera nie muszą być rozwiązaniami optymalnymi.

3.4. v.net.steiner

Algorytm generujący drzewo Steinera wymaga takich samych parametrów, co v.net.allpairs. W parametrach zaawansowanych można dodatkowo określić maksymalną dopuszczalną liczbę rozgałęzień drzewa (*Number of Steiner points*). Pozostaw tę wartość na -1, co oznacza brak ograniczeń w tym zakresie. QGIS utworzy tyle rozgałęzień, ile będzie mu potrzebne do zminimalizowania długości linii.



4. INSPEKcja SIECI Z WTYCZKĄ CHINESE POSTMAN SOLVER

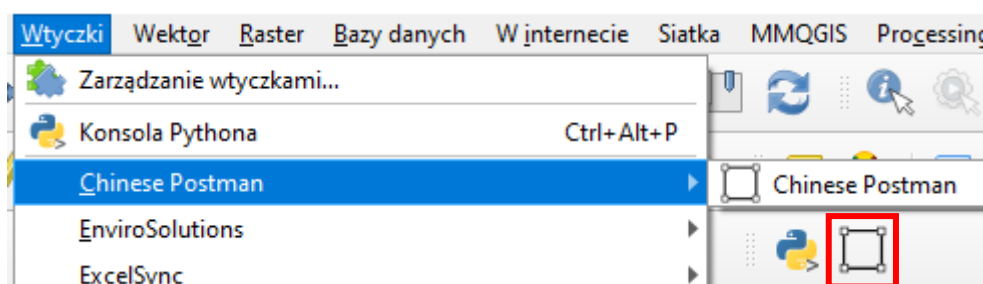
Zainstaluj wtyczkę *Chinese Postman Solver*. Pozwala ona na wyznaczenie najkrótszej ścieżki w sieci, która przechodzi przez wszystkie łuki (ulice) w obrębie tej sieci, czyli rozwiązuje tzw. problem chińskiego listonosza lub problem inspekcji sieci drogowej.

Wtyczka ta oblicza optymalną drogę dla zaznaczonego fragmentu sieci. Zaznacz znajdującą się w centrum część sieci drogowej warstwy *drogi_pozCS92*.



Uwaga: Obliczenia optymalnej ścieżki są czasochłonne – mogą trwać nawet godziny! Stąd pierwsze próby rozwiązania zadania powinny się podjąć przy uwzględnieniu jedynie małych fragmentów sieci.

Uruchom wtyczkę. Można to zrobić wybierając menu [*Wtyczki*→*Chinese Postman*→*Chinese Postman*] lub wybierając symbol solwera z paska narzędzi (rys. niżej w ramce).

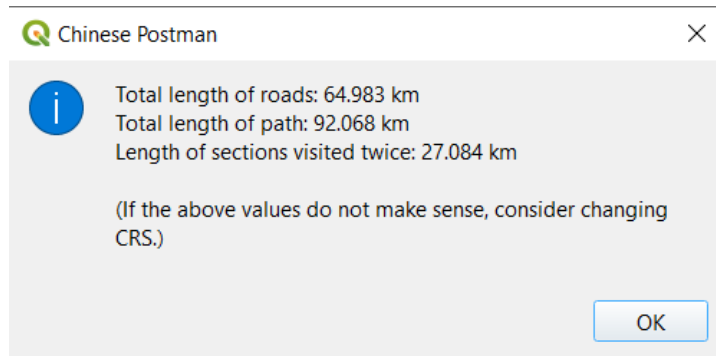


Włączenie wtyczki spowoduje uruchomienie procedury obliczeniowej. W przypadku, gdy sieć drogowa składa się z wielu rozłącznych „wysp”, wtyczka poinformuje o tym i wybierze największą z nich.

Po zakończeniu obliczeń wyświetli się okno z wynikami, w którym wskazane będą trzy parametry:

- Total length of roads – całkowita długość dróg, na których rozwiązywany był problem.
- Total length of path – całkowita długość najkrótszej ścieżki, która pozwala na przejście po każdym odcinku drogi.
- Length of sections visited twice – długość odcinków dróg, które zostały wykorzystane dwukrotnie.

Ponadto w oknie zawarta jest uwaga, że jeśli uzyskane wyniki nie mają sensu, to powinno się ustawić właściwy układ współrzędnych (dla Polski PL-92 lub PL-2000).



Wtyczka utworzy też tymczasową warstwę *chinese_postman*, która będzie przedstawiać wyznaczoną ścieżkę wraz z kierunkiem ruchu.



Należy zauważyć, że wtyczka zawiera uproszczoną implementację problemu, w której nie jest rozróżniania kierunkowość dróg (zapropionowana ścieżka może prowadzić „pod prąd”).