



Politechnika Łódzka

Instytut Informatyki

PRACA DYPLOMOWA MAGISTERSKA

Optymalizacja struktury sieci drogowej

Wydział Fizyki Technicznej, Informatyki i Matematyki Stosowanej

Promotor: dr hab. inż. Aneta Poniszecka-Marańda

Współpromotor: mgr inż. Łukasz Chomątek

Dyplomant: inż. Michał Siatkowski

Nr albumu: 186865

Kierunek: Informatyka

Specjalność: Sztuczna Inteligencja i Inżynieria Oprogramowania

Łódź 30.03.2015

Instytut Informatyki

90-924 Łódź, ul. Wólczańska 215, budynek B9

tel. 042 631 27 97, 042 632 97 57, fax 042 630 34 14 email: office@ics.p.lodz.pl



Spis treści

| | |
|--|-----------|
| 1 Wstęp | 5 |
| 1.1 Problematyka optymalizacji ruchu drogowego | 5 |
| 1.2 Cel pracy | 6 |
| 1.3 Zakres pracy | 6 |
| 1.4 Przegląd literatury w dziedzinie | 7 |
| 1.5 Układ pracy | 8 |
| 2 Optymalizacja struktury sieci drogowej | 9 |
| 2.1 Podstawowe definicje | 9 |
| 2.2 Sieć drogowa w postaci grafu | 10 |
| 2.3 Paradoks Braessa | 12 |
| 2.3.1 Przykładowy, wyjściowy układ drogowy | 12 |
| 2.3.2 Przykładowy, uzupełniony układ drogowy | 14 |
| 2.3.3 Wyjaśnienie intuicyjne paradoksu | 15 |
| 2.4 Graf skierowany silnie spójny | 15 |
| 2.5 Klasyczny algorytm genetyczny | 16 |
| 2.5.1 Podstawowe pojęcia algorytmów genetycznych | 16 |
| 2.5.2 Klasyczny algorytm genetyczny | 17 |
| 2.5.3 Operacje klasyczne, selekcja | 18 |
| 2.5.4 Operacje klasyczne, krzyżowanie | 19 |
| 2.5.5 Operacje klasyczne, mutacja | 19 |
| 2.6 Użycie grafów w algorytmie genetycznym | 20 |
| 3 Technologie i metody użyte w projekcie | 21 |
| 3.1 Symulator transportu | 21 |
| 3.1.1 Agenci | 21 |
| 3.1.2 Symulacja | 22 |
| 3.2 Opis badanego miasta Sioux Falls | 23 |
| 3.3 Ustawienia symulatora transportu | 26 |

SPIS TREŚCI

| | | |
|-----------------------|---|-----------|
| 3.4 | Algorytmy genetyczne | 28 |
| 3.5 | Obsługa grafów | 29 |
| 3.6 | Technologie i metodologie programistyczne | 29 |
| 3.7 | Wdrożenie | 29 |
| 3.8 | System obliczeniowy | 29 |
| 4 | Opis projektu | 31 |
| 4.1 | Dane wejściowe | 31 |
| 4.2 | Ustawienia projektu | 33 |
| 4.3 | Wyniki | 34 |
| 4.4 | Analiza symulacji | 45 |
| 5 | Podsumowanie | 49 |
| 5.1 | Dyskusja wyników | 49 |
| 5.2 | Analiza wyników | 50 |
| 5.3 | Perspektywy dalszych badań w dziedzinie | 52 |
| 5.4 | Struktura projektu | 53 |
| Spis rysunków | | 55 |
| Spis tabel | | 57 |
| Spis listingów | | 59 |
| Bibliografia | | 61 |
| Abstract | | 63 |

Rozdział 1

Wstęp

W poniższym rozdziale zostały pokrótko opisane motywy wyboru tematu pracy dyplomowej. Omówiono najważniejsze aspekty problemu optymalizacji ruchu drogowego oraz przedstawiony został cel niniejszej pracy. Na zakończenie przybliżona zostaje struktura pracy wraz z krótkim omówieniem kolejnych rozdziałów.

1.1 Problematyka optymalizacji ruchu drogowego

Problemy komunikacji w dzisiejszych miastach są wszystkim znane. Zatory drogowe i korki w godzinach szczytu są chlebem powszednim. Pomimo wielu prób i sposobów, wciąż nie istnieje metoda jednoznacznie rozwiązująca tę kwestię. Bezspornie, dotyczy to wszystkich miast na świecie. Z teoretycznego punktu widzenia, jedynym rozwiązaniem jest komunikacja publiczna. Oczywiście jest jednak, że nigdy nie doprowadzimy do sytuacji, gdy wszyscy mieszkańcy zrezygnują ze swoich pojazdów. Dodatkowo, wiele osób i usług wymaga oddzielnej formy transportu. W obliczu tych faktów miasta decydują się na rozwój swojej infrastruktury drogowej. Budowa nowych tras oraz poszerzanie starych przynosi nadzieję mniejszych zatorów, a co za tym idzie, szybszego przejazdu do celu. Niestety, historia pokazuje, że takie inwestycje nie zawsze przynoszą oczekiwane korzyści.

Teorii próbujących解释 te zjawiska, jak również dowodów, które je popierają lub obalają jest wiele. Jedną z najpopularniejszych oraz taką, która została wykorzystana w niektórych miastach na świecie jest paradoks Braessa [22]. Jest to twierdzenie matematyczne orzekające, że w pewnym modelu ruchu drogowego, czasy podróży pojazdów mogą ulec wydłużeniu po dodaniu do sieci drogowej nowego połączenia. Ma ono również zastosowanie w przypadku sieci komputerowych oraz istnieją jego analogie dla doświadczeń fizycznych.

1.2 Cel pracy

Tematem pracy jest optymalizacja struktury sieci drogowej. Opierając się o wspomniany paradoks Braessa sformułowany został cel, którym jest stworzenie rozwiązania, dokonujące optymalizacji zadanej sieci drogowej.

Optymalizacja zostanie przeprowadzona wykorzystując algorytmy genetyczne, które zakładają użycie pewnych z góry ustalonych parametrów. Modyfikacja sieci drogowej ma odbyć się poprzez zastosowanie założenia paradoksu, dokładniej, przez zamknięcie wybranych ulic. W efekcie dla danej sieci drogowej, średni czas podróży powinien ulec skróceniu.

1.3 Zakres pracy

Studia literaturowe

Badania rozpoczęto od poszukiwania źródeł traktujących o opisywanym problemie. Paradoks Braessa został sformułowany w 1970 roku i był od tego czasu wykorzystywany przy planowaniu przestrzeni i infrastruktury wielu miast, na przykład [23]:

- Korea, Seul, likwidacja między innymi estakad Cheonggyecheon,
- Niemcy, Stuttgart, likwidacja dróg zbudowanych w latach 60-tych XX wieku,
- USA, Nowy Jork, czasowe zamknięcie ulicy 42,
- Kanada, Winnipeg.

Propozycja rozwiązania problemu

Oczywistym rozwiązaniem problemu komunikacji mogłoby być stworzenie idealnej sieci odpowiadającej potrzebom danego miasta. Rozbudowa lub modyfikacja tej infrastruktury jest jednak kosztowna i czasochłonna. Z tego powodu sprawdzane jest rozwiązanie zaproponowane przez Braessa. Ponieważ istnieją prace negujące lub podważające paradoks [2], zdecydowano, by przy potwierdzaniu wyniku optymalizacji nie kierować się wyłącznie założeniami w nim zawartymi.

Opis zastosowania algorytmów genetycznych

Ponieważ nie zostały znalezione żadne przesłanki wskazujące jednoznaczną ocenę co do słuszności zamknięcia danej ulicy, w pracy zdecydowano się na losowe przeszukiwanie przestrzeni rozwiązań. Jednym z rozwiązań w przypadku takich poszukiwań są algorytmy genetyczne, które zostały wykorzystane w pracy.

Przedstawienie oceny optymalizacji

Paradoks Braessa zakłada dość oczywiste potwierdzenie swojej wiarygodności. Dlatego, w celu sprawdzenia jego słuszności, wybrany został zewnętrzny sposób oceniania. Taką rolę spełnia system symulacji. W efekcie, ocena rozwiązania jest niezależna od metody twierdzenia, poprzez symulację rzeczywistego ruchu w danej sieci drogowej. Wynik symulacji jest jednoznaczną wartością liczbową, przedstawiającą średni czas przejazdów wszystkich pojazdów, biorących udział w danym scenariuszu. Zakładając niezmienne plany docelowe przejazdów, poprzez manipulację strukturą drogową, dążymy oczywiście do minimalizacji średniego czasu przejazdu.

Ocena możliwości wdrożenia proponowanych rozwiązań

Paradoks Braessa nie jest jedynym twierdzeniem traktującym o problemach komunikacyjnych miast. Wiele teorii jest opartych głównie na socjologicznych lub psychologicznych założeniach. Są to na przykład paradoks Downsa Thomsona [24] czy prawo Lewisa Mogridge'a [25]. Niestety, są one niemniej ważne. Biorąc pod uwagę złożoność problemu, wynik otrzymany podczas eksperymentu nie może być dowodem ani decydującym głosem w decyzjach dotyczących ustalania rzeczywistej sieci drogowej miasta.

1.4 Przegląd literatury w dziedzinie

By przybliżyć temat problemów komunikacyjnych i rozwiązania zaproponowanego przez Braessa polecamy pracę magisterską *Leslie Arthur Keith Bloy* [1]. W pracy zostały opisane również inne twierdzenia dotyczące ruchu drogowego. Publikacja *Reducing the Effects of the Braess Paradox with Information Manipulation* [4] jest bardzo dobrym uzupełnieniem tematu o interesującą nas kwestię symulacji wieloagentowych. Prezentuje ona różnice w wynikach dla przypadku losowego wyboru drogi przez agentów oraz tej wybranej przy użyciu inteligencji kolektywnej¹.

Zbiorowa praca napisana na potrzeby międzynarodowej konferencji dotyczącej technologii symulacji [4] jest pozycją, która opisuje podobny do podejmowanego przez nas problem. Mianowicie skupia się na modyfikacjach drogi poprzez zmiany dostępności jej składowych.

W zupełnie oddzielnej tematyce, algorytmów genetycznych, polecamy pozycję, która jest wstępem do tej tematyki [6]. Pozycja zawiera bogaty zestaw problemów przykładowych wraz z opisem ich rozwiązania. Ponadto dogłębnie opisuje ona każdy aspekt działania zastosowanych algorytmów.

¹ang. Collective Intelligence (COIN)

1.5 Układ pracy

Na początku przedstawione zostały zagadnienia dotyczące trudności komunikacyjnych miast oraz znane i wykorzystywane powszechnie rozwiązania. Przybliżony został cel pracy wraz z opisem zaproponowanej metody optymalizacji problemu.

Rozdział 2 zawiera opis teoretyczny zagadnień. Wyjaśnia on wykorzystywane struktury oraz matematyczne prawa wykorzystane podczas ich modyfikacji. Wskazany zostaje sposób reprezentacji struktur sieci drogowych i ich optymalizacja.

W rozdziale 3 opisane zostają technologie, które zostały użyte podczas tworzenia rozwiązania. Zawiera on opis zewnętrznych biblioteki, użytych w pracy oraz wyjaśnienie działania kolejnych elementów systemu. Opisany zostaje również sposób wdrożenia rozwiązania.

Głównym rozdziałem pracy jest rozdział 4, w którym przedstawiono wykonaną aplikację. Omówione zostają również parametry wykorzystane podczas badań wraz z ich wartościami. Na zakończenie przedstawione zostają wyniki optymalizacji przykładowej sieci.

W rozdziale podsumowującym 5, zawarte są wnioski wraz z omówieniem wyników badań. Prezentowane są możliwości rozwoju pracy. Kolejne części pracy to spis rysunków, tabel, listingów i bibliografie, w tej kolejności.

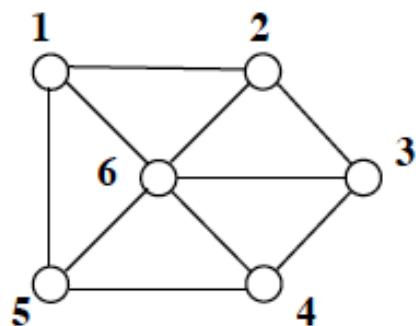
Rozdział 2

Optymalizacja struktury sieci drogowej

Poniższy rozdział zawiera opis teoretyczny zagadnień. Wyjaśnia on wykorzystywane struktury oraz matematyczne prawa wykorzystane podczas ich modyfikacji. Wskazany zostaje sposób reprezentacji struktur sieci drogowych i ich optymalizacja.

2.1 Podstawowe definicje

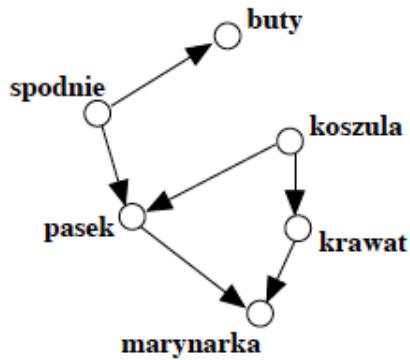
Definicja 1 *Grafem (nieskierowanym) nazywa się parę zbiorów (V, E) . Elementy zbioru V nazywają się wierzchołkami, natomiast elementy zbioru E to krawędzie. Każda krawędź jest parą wierzchołków, tzn. $E \subseteq u, v : u, v \in V$ [9].*



Rys. 2.1: Przykładowy graf nieskierowany

Przykładowy graf nieskierowany (Rys. 2.1) może zostać opisany przez zbiory: $V = \{1, 2, 3, 4, 5, 6\}$ $E = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 5\}, \{6, 5\}, \{6, 1\}, \{2, 6\}, \{3, 6\}, \{4, 6\}\}$

Definicja 2 *Grafem skierowanym nazywa się taki graf, w którym każda krawędź ma zdefiniowany początek i koniec, tzn. pary grafu muszą być uporządkowane. Wtedy $E \subseteq V \times V = u, v : u, v \in V$. Krawędź (u, v) najłatwiej wyobrazić sobie jako strzałkę od u do v , dlatego często oznacza się ją przez $u \rightarrow v$ [9].*



Rys. 2.2: Przykładowy graf skierowany

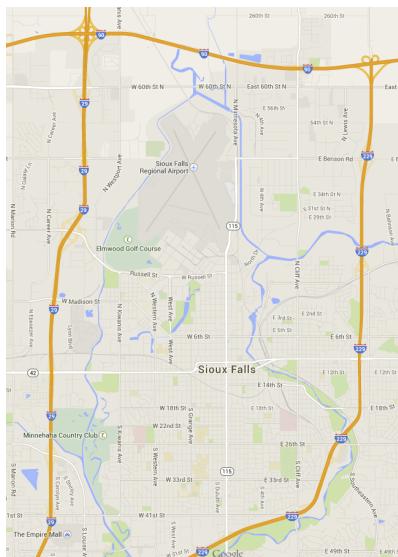
Przykładowy graf skierowany (Rys. 2.2) może zostać opisany przez zbiory:

$$V = \{spodnie, buty, pasek, koszula, krawat, marynarka\}$$

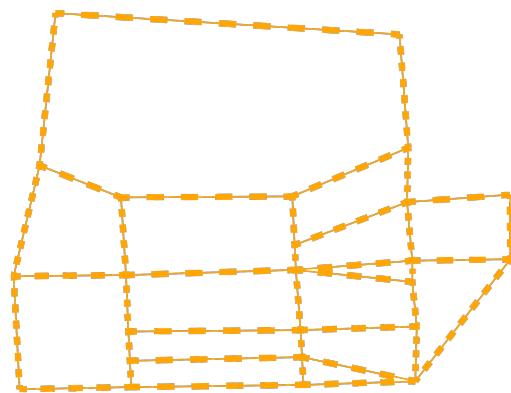
$$E = \{\{spodnie \rightarrow buty\}, \{spodnie \rightarrow pasek\}, \{pasek \rightarrow koszula\}, \{koszula \rightarrow krawat\}, \{pasek \rightarrow marynarka\}, \{krawat \rightarrow marynarka\}\}$$

2.2 Sieć drogowa w postaci grafu

W przypadku sieci drogowej mamy oczywiście do czynienia z abstrakcyjną strukturą sieci. Przez sieć drogową rozumie się bowiem układ dróg lub ulic na przykład w mieście. W ramach pracy wykorzystano pewien fragment większej sieci. Przykładowo, było to miasto Sioux Falls w Południowej Dakocie, prezentowane na rysunku 2.3.

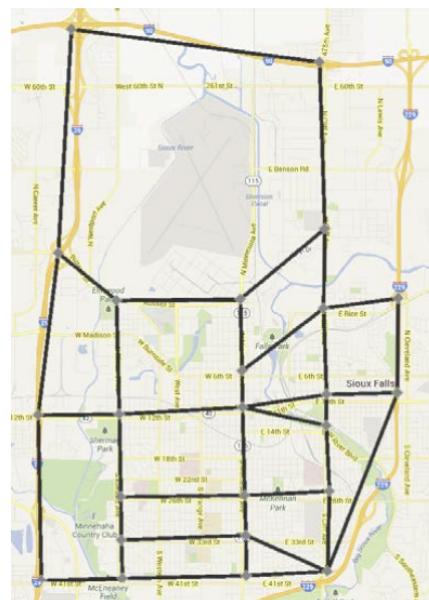


- E odnosi się do ulic pomiędzy tymi skrzyżowaniami. W pracy posługiwano się zawsze grafem skierowanym. W związku z tym, w przypadku ulic dwukierunkowych tworzone są pary krawędzi z odpowiednimi kierunkami, nawet jeśli ulice nie są rozłączne w rzeczywistości. Efekt transformacji fragmentu sieci drogowej zaprezentowanej w postaci grafu skierowanego zaprezentowano na rysunku 2.4.



Rys. 2.4: Sieć drogowa miasta Sioux Falls w postaci grafu

Na pierwszy rzut oka nie jest widoczne podobieństwo pomiędzy mapą a jej odzwierciedleniem w postaci grafu. Wynika to z braku dopasowania geometrycznego po przekształceniu. Poniżej (Rys. 2.5) prezentowana jest sieć drogowa miasta w postaci grafu, po przekształceniu geometrycznym, tak by odzwierciedlała on mniej więcej ulice widoczne na mapie.



Rys. 2.5: Graf sieci drogowej miasta z dopasowaną geometrią

2.3 Paradoks Braessa

Jak już wcześniej wspomniano, paradoks Braessa to twierdzenie matematyczne orzekające, że w pewnym modelu ruchu drogowego czasy podróży pojazdów mogą ulec wydłużeniu po dodaniu do sieci drogowej nowego połączenia. Autorem twierdzenia jest niemiecki matematyk Dietrich Braess [22]. Paradoks działa w oparciu o model ruchu drogowego, który ma następujące cechy [5]:

1. Sieć drogowa składa się ze skończenie wielu węzłów i łączących je odcinków dróg.
2. Po sieci porusza się skończenie wiele pojazdów, a każdy z nich ma wyznaczony węzeł startowy i węzeł docelowy.
3. Odcinki dróg mają przypisane sobie czasy przejazdu, przy czym czasy te mogą zależeć od liczby pokonujących dany odcinek pojazdów.
4. Układ sieci drogowej i czasy przejazdu poszczególnych odcinków są znane pojazdom.
5. Celem pojazdów jest przejazd przez sieć z węzłów początkowych do docelowych po trasie złożonej z odcinków drogowych tak, by zminimalizować łączny czas ich pokonania.
6. Decyzję o wyborze tras pojazdy podejmują indywidualnie i niezależnie od siebie.

2.3.1 Przykładowy, wyjściowy układ drogowy

Paradoks w oryginalnym artykule wytlumaczony jest na prostym przykładzie, który jest zaprezentowany poniżej. Za punkt wyjściowy przyjmujemy pewny układ dróg o znanych czasach przejazdów.

Sieć drogowa i auta

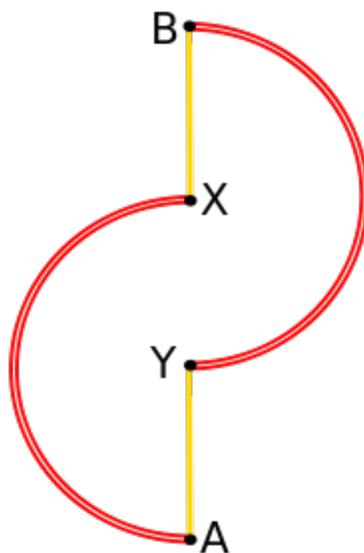
Sytuacja, w której ujawnia się paradoks Braessa jest skonstruowany z czterech miast A , B , X i Y . Są one połączone odcinkami drogowymi, jak na rysunku 2.6 i z odpowiednimi czasami przejazdu, przy czym p oznacza gęstość ruchu w tysiącach aut.

Poniżej znajduje się opis dostępnej sieci drogowej przedstawionej na rysunku 2.6.

Autostrady:

$$AX, t_{AX}(p) = 50 + p \text{ min}$$

$$YB, t_{YB}(p) = 50 + p \text{ min}$$



Rys. 2.6: Wyjściowy układ drogowy

Drogi lokalne:

$$AY, t_{AY}(p) = 10p \text{ min}$$

$$XB, t_{XB}(p) = 10p \text{ min}$$

Aut jest 6000 i wszystkie mają za zadanie przejechać trasę z A do B.

Analiza równowagi Nasha

Każde auto musi zdecydować się na wybór trasy: albo AXB , albo AYB . **Równowaga Nasha** to taka sytuacja, w której każdy z samochodów spowoduje wydłużenie swojego czasu jazdy, zmieniając decyzję dotyczącą wyboru trasy przy niezmienionych decyzjach pozostałych aut. Jeśli p i q oznaczają liczby aut w tysiącach, pokonujących odpowiednio trasę AXB i AYB , to na podstawie danych do rysunku 2.6 otrzymuje się równania:

$$\begin{aligned} p + q &= 6 \\ t_{AX}(p) + t_{XB}(p) &= t_{AY}(q) + t_{YB}(q) \\ 50 + p + 10p &= 10q + 50 + q \end{aligned}$$

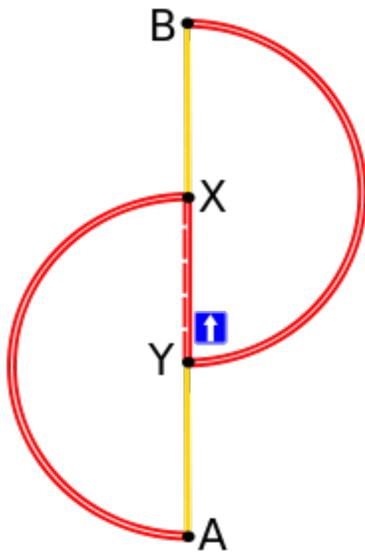
Rozwiązaniem równania jest $p = q = 3$. Wynik może zostać wykorzystany do obliczenia średniego czasu przejazdu aut po dostępnej sieci drogowej. Przy tej gęstości ruchu pokonanie obu dostępnych tras zabiera $50 + 3 + 30 = 83$ minuty.

2.3.2 Przykładowy, uzupełniony układ drogowy

Pamiętając o początkowym stanie dróg i średnim czasie przejazdu, układ drogowy zostaje uzupełniony o autostradę, mającej na celu skrócenie czasu potrzebnego na podróż pomiędzy punktami w grafie.

Sieć drogowa i auta

Do wyjściowego układu drogowego dodana zostaje autostrada YX . Uzupełniony układ drogowy zaprezentowany jest na rysunku 2.7.



Rys. 2.7: Uzupełniony układ drogowy

Poniżej znajduje się opis dotyczący uzupełnionego fragmentu sieci drogowej przedstawionej na rysunku 2.7.

Autostrady:

$$YX, t_{YX}(p) = 10 + p \text{ min}$$

Aut jest nadal 6000 i wszystkie mają za zadanie przejechać trasę z A do B .

Analiza równowagi Nasha

Jeśli p, q i r oznaczają liczby aut w tysiącach pokonujących odpowiednio trasy AXB , AYB i $AYXB$, to na podstawie danych do rysunku 2.7 otrzymuje się równania:

$$p + q + r = 6$$

$$t_{AX}(p) + t_{XB}(p + r) = t_{AY}(q + r) + t_{YB}(q) = t_{AY}(q + r) + t_{YX}(r) + t_{XB}(p + r)$$

$$50 + p + 10(p + r) = 10(q + r) + 50 + q = 10(q + r) + 10 + r + 10(p + r)$$

Rozwiązaniem równania jest $p = q = r = 2$. Wynik może zostać wykorzystany do obliczenia średniego czasu przejazdu aut po dostępnej sieci drogowej. Czas przejazdu dla każdej z tych dróg wynosi wówczas $50 + 2 + 10(2 + 2) = 92$ minuty.

2.3.3 Wyjaśnienie intuicyjne paradoksu

Wąskim gardłem systemu są drogi lokalne, na których czas przejazdu bardzo szybko wzrasta wraz z intensywnością ruchu. Po pojawienniu się dodatkowej drogi dostępna staje się nowa trasa, prowadząca oprócz nowego skrótu YX tylko drogami lokalnymi.

Z perspektywy całości systemu nowy odcinek drogowy odciąża ruch na autostradach, gdzie jest to mało odczuwalne, a w zamian jeszcze bardziej zagęszcza ruch na drogach lokalnych, powodując wydłużenie czasu podróży.

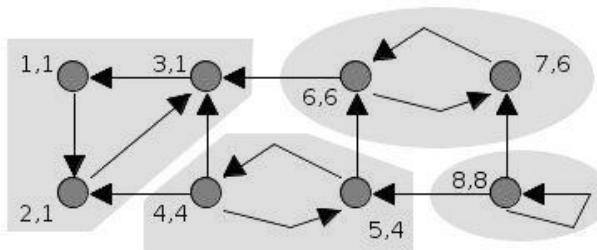
2.4 Graf skierowany silnie spójny

Na potrzeby wybranego w projekcie systemu symulacji, sieć drogowa przedstawiona w postaci grafu skierowanego musi spełniać warunek silnej spójności.

Definicja 3 *Grafem skierowanym silnie spójnym nazywamy graf skierowany, w którym możliwe jest dotarcie do każdego wierzchołka, zaczynając z dowolnego innego poprzez dowolną ilość krawędzi. Wszystkie wierzchołki w grafie skierowanym silnie spójnym muszą zatem posiadać przynajmniej jedną krawędź wchodząjącą i jedną wychodzącą[26].*

Definicja 4 *Składowa silnie spójna (ang. strongly connected component) jest maksymalnym podgrafem, w którym istnieją ścieżki pomiędzy każdym dwoma wierzchołkami. Jeśli podgraf ten obejmuje wszystkie wierzchołki grafu, to mówimy, że dany graf skierowany jest silnie spójny (ang. strongly connected digraph). W grafach nieskierowanych każdy graf spójny jest również silnie spójny. Poglądowo sytuacja ta jest przedstawiona na rysunku 2.8.*

W sprawdzaniu grafu pod względem spójności, wykorzystujemy **algorytm Tarjana** do znajdowania składowych silnie spójnych. Podstawowym założeniem algorytmu Tarjana jest przeszukiwanie grafu w głąb zaczynając od dowolnego wierzchołka wybranego w sposób arbitralny. Tak, jak w przypadku klasycznego przeszukiwania w głąb, każdy sąsiadujący wierzchołek po odwiedzeniu zostaje oznaczony i algorytm nigdy ponownie go nie odwiedza. Dzięki temu tworzy się kolekcję przeszukanych drzew, która jest drzewem rozpinającym grafu. Składowe silnie spójne są następnie odnajdowane jako poddrzewa, a korzenie tych poddrzew są nazywane korzeniami składowych silnie spójnych.



Rys. 2.8: Przykładowy graf z zaznaczonymi składowymi silnie spójnymi

Każdy wierzchołek grafu może być wybrany na korzeń składowej silnie spójnej, jeśli zostanie wybrany jako pierwszy wierzchołek podczas przeszukiwania w głąb.

W efekcie, zawsze przed rozpoczęciem symulacji sprawdza się, czy graf jest grafem skierowanym silnie spójnym lub dokładniej mówiąc, czy posiada tylko jedną składową silnie spójną.

2.5 Klasyczny algorytm genetyczny

Idea algorytmu genetycznego została zaczerpnięta z nauk przyrodniczych, opisujących zjawiska doboru naturalnego i dziedziczenia. Mechanizmy te polegają na przetrwaniu osobników najlepiej dostosowanych w danym środowisku, podczas gdy osobniki gorzej przystosowane są eliminowane. Z kolei te osobniki, które przetrwają, przekazują informację genetyczną swoim potomkom. Krzyżowanie informacji genetycznej otrzymanej od „rodziców” prowadzi do sytuacji, w której kolejne pokolenia są przeciętnie coraz lepiej dostosowane do warunków środowiska. Mamy tu więc do czynienia ze swoistym procesem optymalizacji.

2.5.1 Podstawowe pojęcia algorytmów genetycznych

Definicja 5 *Populacją* nazywamy zbiór osobników o określonej liczbie.

Definicja 6 *Osobnikami* populacji w algorytmach genetycznych są zakodowane w postaci chromosomów zbiory parametrów zadania, czyli rozwiązania, określone też jako punkty przestrzeni poszukiwań. Osobniki czasami nazywa się organizmami.

Definicja 7 *Chromosomy*, inaczej łańcuchy lub ciągi kodowe, to uporządkowane ciągi genów.

Definicja 8 *Gen, nazywany też cechą, znakiem, detektorem, stanowi pojedynczy element genotypu, w szczególności chromosomu. Genotyp, czyli struktura, to zespół chromosomów danego osobnika. Zatem osobnikami populacji mogą być genotypy albo pojedyncze chromosomy.*

Definicja 9 *Fenotyp jest zestawem wartości, odpowiadających danemu genotypowi, czyli zdekodowaną strukturą, a więc jest zbiorem parametrów zadania (rozwiązaniem, punktem przestrzeni poszukiwań).*

Definicja 10 *Allel to wartość danego genu, określona jako wartość cechy lub wariant cechy.*

Definicja 11 *Locus to pozycja, która wskazuje miejsce położenia danego genu w łańcuchu, czyli chromosomie.*

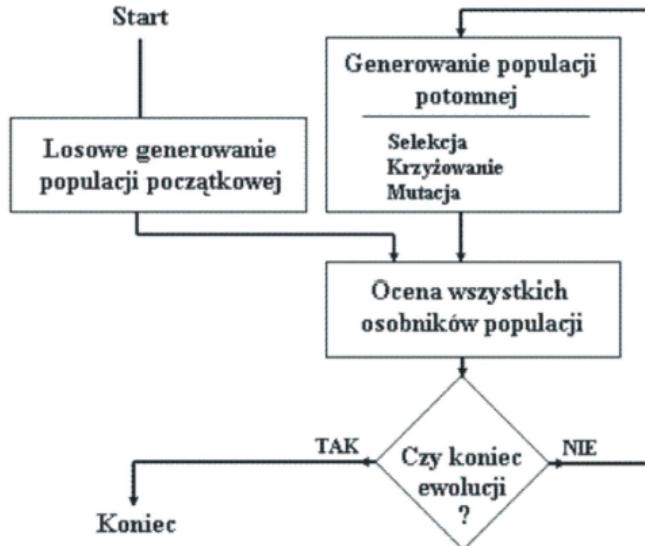
Definicja 12 *Funkcja przystosowania nazywana też funkcją dopasowania lub funkcją oceny Stanowi ona miarę przystosowania (dopasowania) danego osobnika w populacji.*

2.5.2 Klasyczny algorytm genetyczny

Na podstawowy (klasyczny) algorytm genetyczny, nazywany także elementarnym lub prostym, składają się następujące kroki:

1. inicjacja, czyli wybór początkowej populacji chromosomów,
2. ocena przystosowania chromosomów w populacji,
3. sprawdzenie warunku zatrzymania,
4. selekcja chromosomów,
5. zastosowanie operatorów genetycznych,
6. utworzenie nowej populacji,
7. wyprowadzenie najlepszego chromosomu.

Najłatwiej wyobrazić sobie powyższe kroki analizując je na schemacie przedstawionym na rysunku 2.9.



Rys. 2.9: Ogólny schemat algorytmu genetycznego

2.5.3 Operacje klasyczne, selekcja

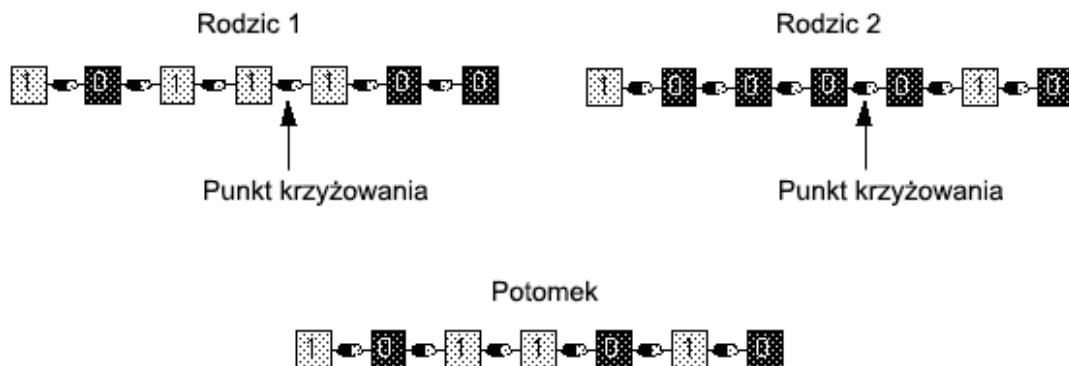
Selekcja chromosomów polega na wybraniu na podstawie obliczonych wartości funkcji przystosowania, tych chromosomów, które będą brały udział w tworzeniu potomków do następnego pokolenia, czyli następnej generacji. Wybór ten zwykle odbywa się zgodnie z zasadą naturalnej selekcji, to znaczy, największe szanse na udział w tworzeniu nowych osobników mają chromosomy o największej wartości funkcji przystosowania. Istnieje wiele metod selekcji populacji.

W projekcie wykorzystana została **selekcję turniejową**, która polega na podzieleniu populacji na podgrupy k-elementowe (k to rozmiar turnieju – zwykle 2 lub 3) i wyborze z każdej podgrupy osobnika o najlepszym przystosowaniu. Można to zrobić poprzez wybór losowy lub wybór deterministyczny. Wtedy wyboru dokonuje się z prawdopodobieństwem równym 1. Metoda turniejowa nadaje się zarówno do problemów maksymalizacji jak i minimalizacji.

W klasycznym algorytmie genetycznym możliwa jest sytuacja, w której do nowej populacji nie zostaną wybrane najlepsze osobniki poprzedniej populacji. Chcąc zapobiec takiej sytuacji zdecydowano się na zastosowanie **strategii elitarnej**. W jej przypadku nacisk położony jest na zachowanie w kolejnych iteracjach najlepiej przystosowanych osobników. Dzięki temu zachowywane jest najlepsze rozwiązanie (lub N najlepszych rozwiązań) pomiędzy iteracjami [11].

2.5.4 Operacje klasyczne, krzyżowanie

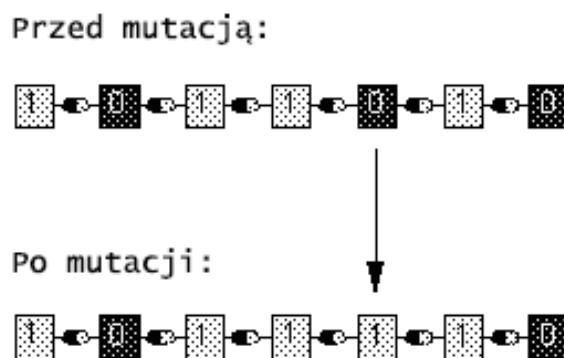
Operacja krzyżowania jest podstawową operacją algorytmów genetycznych, służącą do rekombinacji materiału genetycznego. Operacja opiera się na dwóch chromosomach, których części materiału genetycznego zostają wymieszane w celu utworzenia nowego chromosomu. Podstawowa operacja krzyżowania opiera się na jednym punkcie krzyżowania i została przedstawiona na poglądowym rysunku 2.10 [12].



Rys. 2.10: Ogólny schemat operacji krzyżowania

2.5.5 Operacje klasyczne, mutacja

Proces rekombinacji przez krzyżowanie nie byłby w stanie odkryć całej przestrzeni poszukiwań, jeżeli dana kombinacja nie byłaby obecna w sekcjach populacji. To mogłoby prowadzić do błędного wyniku. **Operacja mutacji** pozwala na wprowadzenie nowych struktur genetycznych w obecnej populacji. Dokonuje się tego poprzez losową zmianę dowolnego genu w chromosomie. Sytuacja jest przedstawiona na poglądowym rysunku 2.11 [12].



Rys. 2.11: Ogólny schemat operacji mutacji

2.6 Użycie grafów w algorytmie genetycznym

W niniejszej pracy posłużono się przede wszystkim grafami. W tym wypadku klasyczna odmiana algorytmu genetycznego musiałaby zostać zmodyfikowana na potrzeby przedstawienia grafów jako chromosomów. Dodatkowo, wymagałoby to zastosowania nowych sposobów krzyżowania i mutacji jednostek.

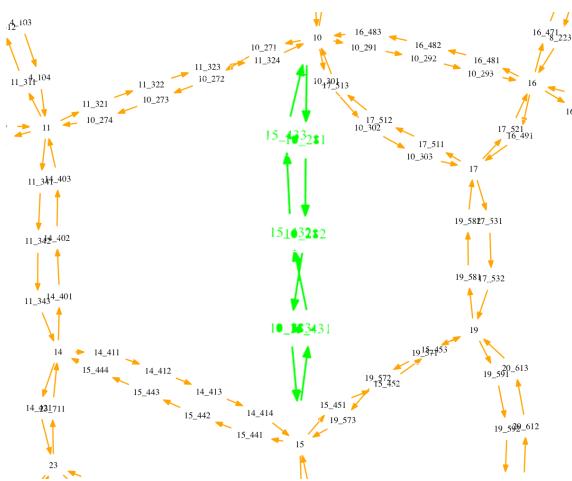
Zdecydowano, że zamiast przystosowywać algorytm genetyczny do nowej struktury, struktura zostanie przystosowana do algorytmu. Ponieważ postawionym zadaniem jest zdecydowanie o zamknięciu lub nie danej ulicy (krawędzi grafu), postanowiono przedstawić tę strukturę jako listę opisującą taki stan. Zgodnie z tą myślą, sieć drogowa (graf) jest tłumaczona na listę elementów przyjmujących wartości:

- 1 (prawda) — dla ulicy (węzła), który jest przejezdny,
 - 0 (fałsz) — dla ulicy (węzła) zamkniętego dla ruchu.

Tworzona w ten sposób lista stanów jest tak naprawdę tablicą booleanowską. Bez problemu może być zatem modyfikowana przez algorytm genetyczny. Bardzo łatwym jest również odtworzenie obecnego stanu sieci drogowej. Przykładową sytuację przedstawiają rysunki 2.12 i 2.13.

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|

Rys. 2.12: Fragment sieci w postaci tablicy binarnej



Rys. 2.13: Fragment sieci w postaci grafu

Przedstawiony na rysunku 2.12 fragment sieci zawiera kolejne elementy zerowe, które w tłumaczeniu na przykładowy graf, są zaznaczone kolorem zielonym (Rys. 2.13). W tym przypadku krawędzie te zostaną wyłączone z ruchu.

Rozdział 3

Technologie i metody użyte w projekcie

W niniejszym rozdziale opisane zostają technologie, które zostały użyte podczas tworzenia rozwiązania. Zawiera on opis zewnętrznych biblioteki, użytych w pracy oraz wyjaśnienie działania kolejnych elementów systemu. Opisany zostaje również sposób wdrożenia rozwiązania.

3.1 Symulator transportu

MATSim jest środowiskiem do implementacji szerokiej skali symulacji transportu, opartych na agentach. Składa się z wielu modułów, które mogą zostać połączone lub używane oddzielnie. Moduły można zastępować własnymi implementacjami w celu przetestowania pojedynczych aspektów pracy [13].

Oczywiście *MATSim* nie jest jedynym dostępnym symulatorem transportu. Podobnych rozwiązań jest wiele. Wybór motywowany był jednak przede wszystkim dostępnością materiałów szkoleniowych i przykładowymi scenariuszami. *MATSim* jest żywym projektem, opartym o licencję otwartego oprogramowania (*ang. open source*). Dysponuje szerokim zasobem przykładów i gotowych scenariuszy.

3.1.1 Agenci

Przy pomocy *MATSim* możliwa jest symulacja ruchu dla codziennych zachowań komunikacyjnych. W tym celu wykorzystywani są agenci, reprezentujący pojedyncze jednostki ludzkie. Każdy agent posiada własny plan dnia, pracę i cele dodatkowe, które realizuje według swojego uznania. Symulacja zwykle pokrywa cały dzień czynności wykonywanych przez wszystkich agentów jednocześnie. Pozwala to na śledzenie pojedynczych jednostek od momentu wyjścia z domu do pracy, robienie zakupów i wieczornego powrót. Celem każdego agenta jest uzyskanie jak najwyższego „wyniku” w ciągu dnia.

Agent zdobywa punkty poprzez realizowanie swoich planów (na przykład: praca, zakupy), natomiast traci je podczas komunikacji lub za spóźnienia.

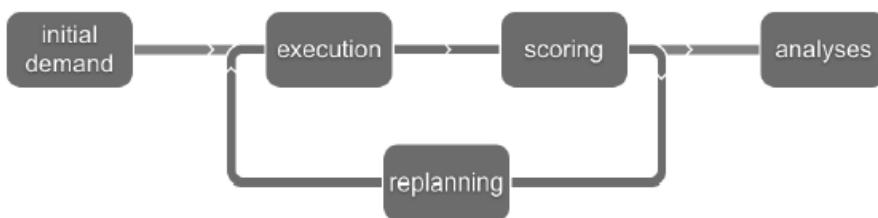
Agenci potrafią modyfikować swoje początkowe plany, tak by nowe ich warianty pozwoliły im na zdobycie wyższego wyniku. Główne założenie procesu optymalizacji wywodzi się z algorytmów (co-)ewolucyjnych. Częstka „co” została dodana ze względu na oddzielność algorytmu genetycznego dla każdego agenta z osobna, jednak wyniki uzyskanych planów dnia, są zależne od całego systemu agentów.

3.1.2 Symulacja

Podczas podstawowej symulacji w środowisku *MATSim*, potrzeby przemieszczania się agentów są symulowane w zadanym środowisku, na przykład sieci drogowej. Podczas typowej symulacji można wyróżnić pięć podstawowych etapów:

- initial demand,
- execution,
- scoring,
- replanning,
- analysis.

Powyzsze kroki uruchamiane są w kolejności, tak jak przedstawiono na rysunku 3.1. Dokładny opis każdego etapu znajduje się w dokumentacji użytkownika [7] oraz zostają dokładniej opisane w rozdziale 3.3.



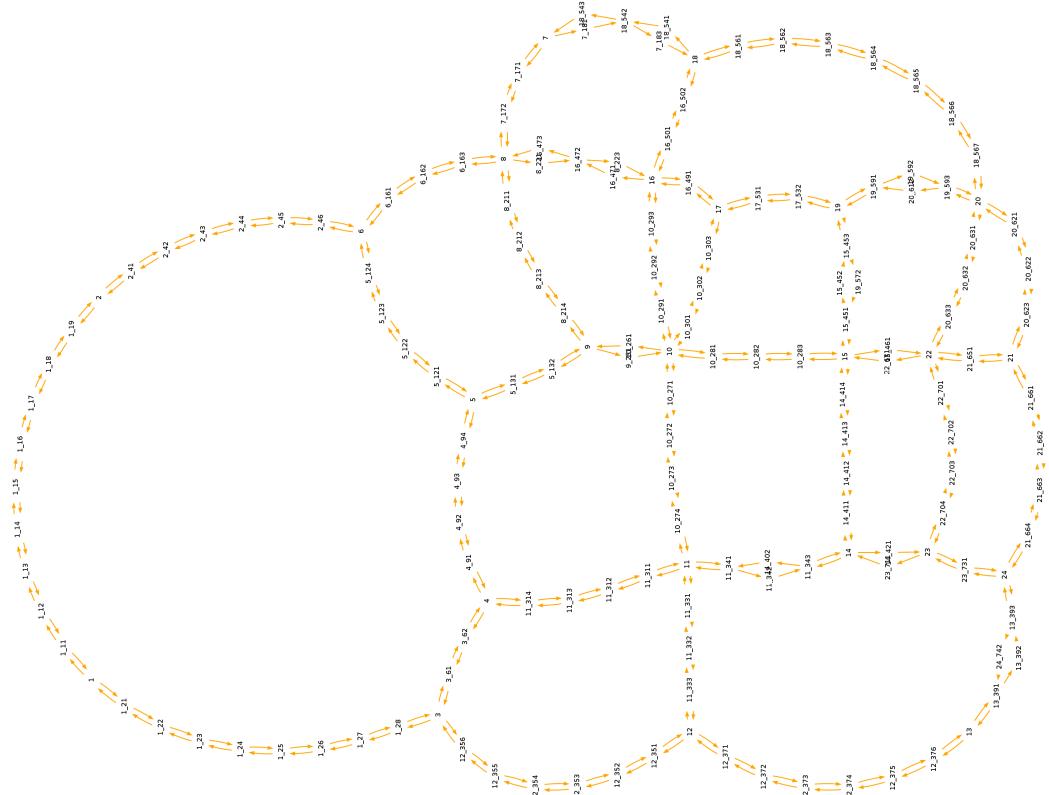
Rys. 3.1: Schemat symulacji

W skrócie, dla zadanej liczby uruchomień symulacji powtarzany jest scenariusz tego samego dnia, dla tego samego zestawu agentów. Ci ostatni, jak zostało wspomniane, po każdym uruchomieniu są oceniani i na podstawie poprzednio uzyskanych wyników, w kolejnej iteracji próbują swoje wyniki polepszyć. Ta optymalizacja, o której wspomniano w rozdziale 3.1.1, jest najważniejszą funkcją symulatora. Jej działanie jest oczywiście nierealne, ponieważ agenci codziennie „przeżywają” ten sam dzień, próbując jak najlepiej dopasować swój plan transportu do panującej sytuacji na drodze.

3.2 Opis badanego miasta Sioux Falls

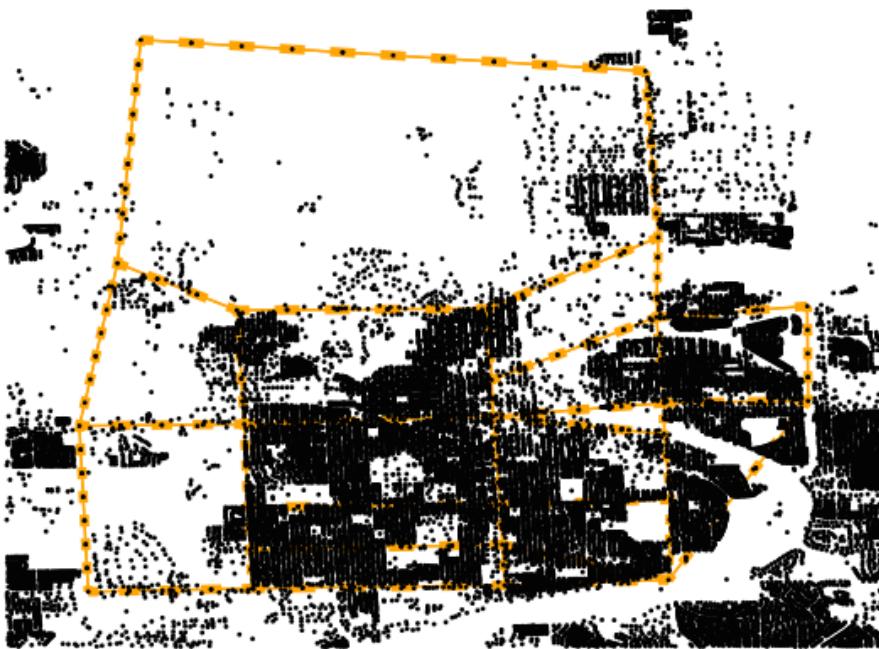
Główna zaletą korzystania z MATSim, jest dość pokaźny zbiór danych przykładowych. Jednym z nich jest materiał zaprezentowany przez twórców aplikacji, który udostępnili w 2013 roku na zgromadzeniu użytkowników platformy [8]. Przykład ten dotyczy miasta Sioux Falls w Południowej Dakocie. Domyślnie scenariusz składa się z:

- dwóch grup zapotrzebowania bez charakterystyk socjodemograficznych:
 - 68094 agentów z samochodem oraz korzystających z transportu publicznego,
 - 40877 agentów posiadających samochód.
- dostosowanej sieci drogowej miasta Sioux Falls,
- transportu publicznego razem z rozkładem jazdy,
- przykładowych miejsc zamieszkania, pracy i rozrywki.



Rys. 3.2: Graf sieci miasta Sioux Falls wykorzystany w badaniach

Dostarczony materiał danych jest zbyt obszerny na potrzeby eksperymentu. Dostosowano go zatem poprzez usunięcie transportu publicznego oraz wyposażenie każdego

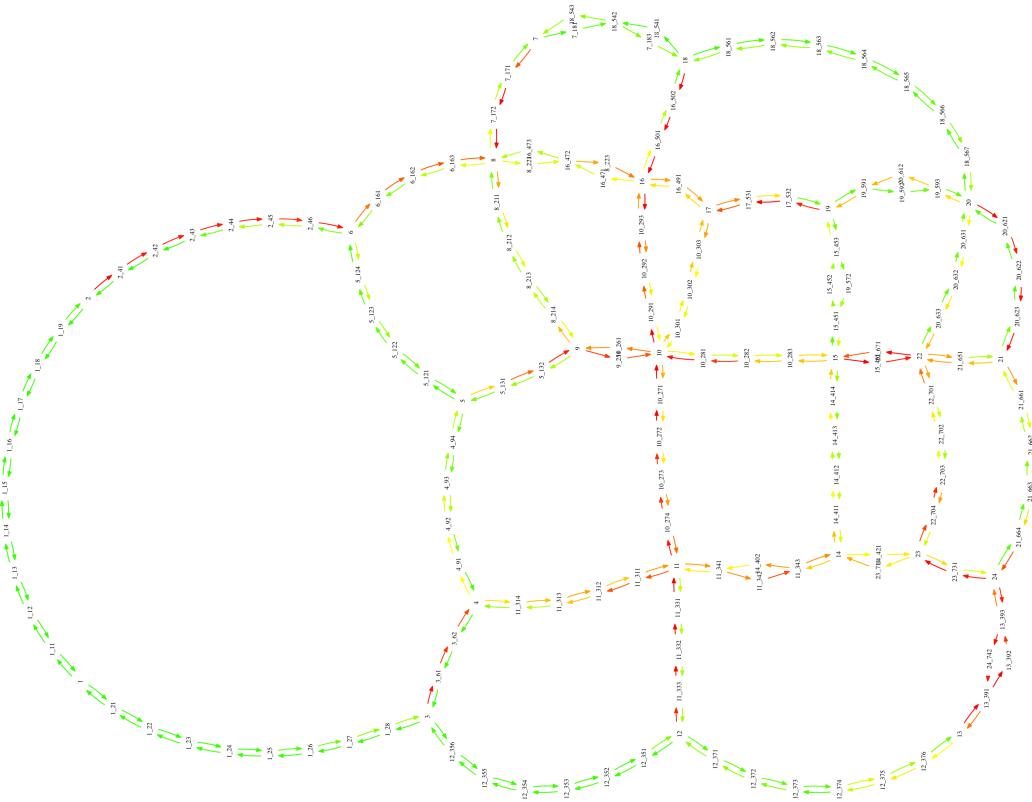


Rys. 3.3: Rozkład budynków na grafie miasta Sioux Falls

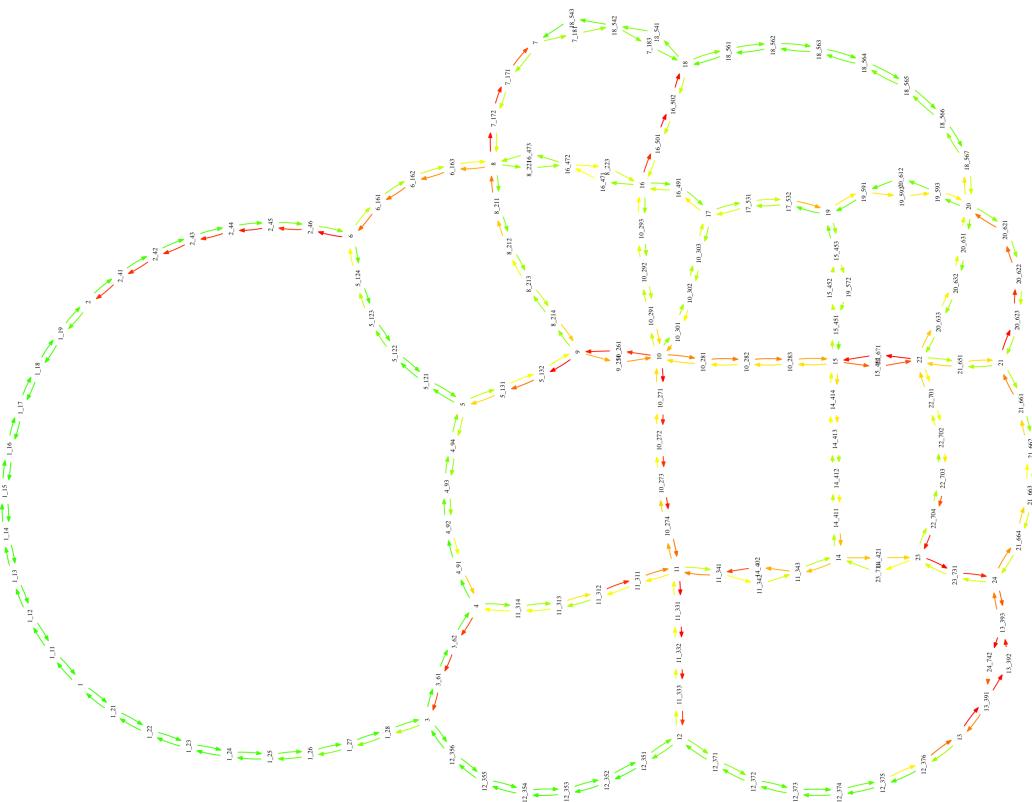
agenta we własny samochód. Powyższe modyfikacje znacznie wzmogły ruch w mieście (co było pozytywnym efektem) i skróciły obliczenia związane z symulacjami. Na rysunku 3.2 przedstawiono sieć miasta odwzorowaną w postaci grafu. Rysunek został obrócony o 90° w lewo, dla zwiększenia jego czytelności. Natomiast na rysunku 3.3 zaprezentowano rozkład miejsc pracy, domostw i innych zakładów nałożonych na graf sieci drogowej miasta.

Na rysunkach 3.4 i 3.5 zaprezentowano ruch drogowy w najbardziej intensywnych godzinach dnia, tj. godzinach szczytu. Ruch jest przedstawiony jako natężenie ruchu na każdej krawędzi (ulicy), poprzez odniesienie go do możliwości przepustowości każdego węzła. Kolor zielony oznacza małe obciążenie, żółty średnie i czerwony - wysokie natężenie ruchu w tym miejscu. Rysunki zostały obrócone o 90° w lewo, dla zwiększenia ich czytelności.

3.2. OPIS BADANEGO MIASTA SIOUX FALLS



Rys. 3.4: Natężenie ruchu w mieście Sioux Falls w godzinach 6.00-7.00

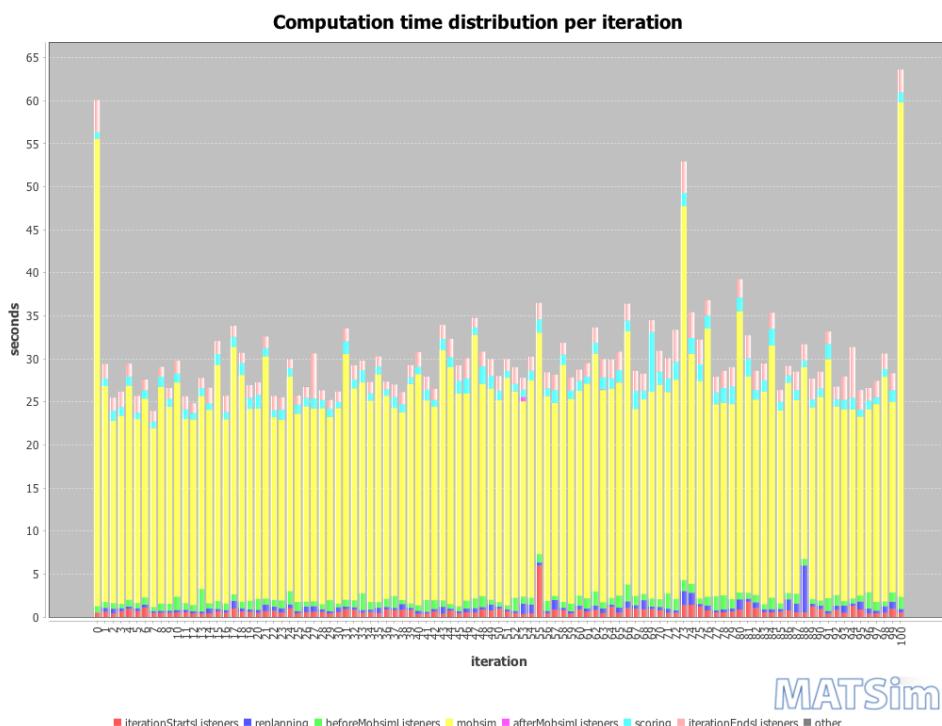


Rys. 3.5: Natężenie ruchu w mieście Sioux Falls w godzinach 16.00-17.00

3.3 Ustawienia symulatora transportu

Ważną kwestią przed realizacją samego zadania optymalizacji jest dobór jego ustawień. Część z nich jest oczywiście stała lub nie ma wpływu bezpośrednio na obliczenia. Kluczową rolę dla ostatecznego wyniku odgrywają jednak parametry samej symulacji.

Większość ustawień została niezmieniona, pochodzi więc od twórców przykładu Sioux Falls, przybliżonego w rozdziale 3.2. Symulator zawiera jednak ważny parametr ilości iteracji. Zgodnie z założeniami symulatora, podczas każdej kolejnej iteracji, celem agenta jest zwiększenie jego średniego wyniku. Dąży on zatem do optymalizacji swoich planów dnia względem warunków na drodze. Krok ten został już opisany w rozdziale 3.1.1. Przekłada się to bezpośrednio również na średni czas podróży każdego agenta. Twórcy zalecają stosowanie wysokich liczb iteracji, dla uzyskania jak najlepszych wyników. Operacje te są jednak czasochłonne. Na rysunku 3.6 przedstawiony został graf czasu uruchomienia symulacji jednej sieci dla stu iteracji. Podczas symulacji wykorzystywana była domyślna sieć miasta.



Rys. 3.6: Graf czasu trwania symulacji MATSim dla stu iteracji

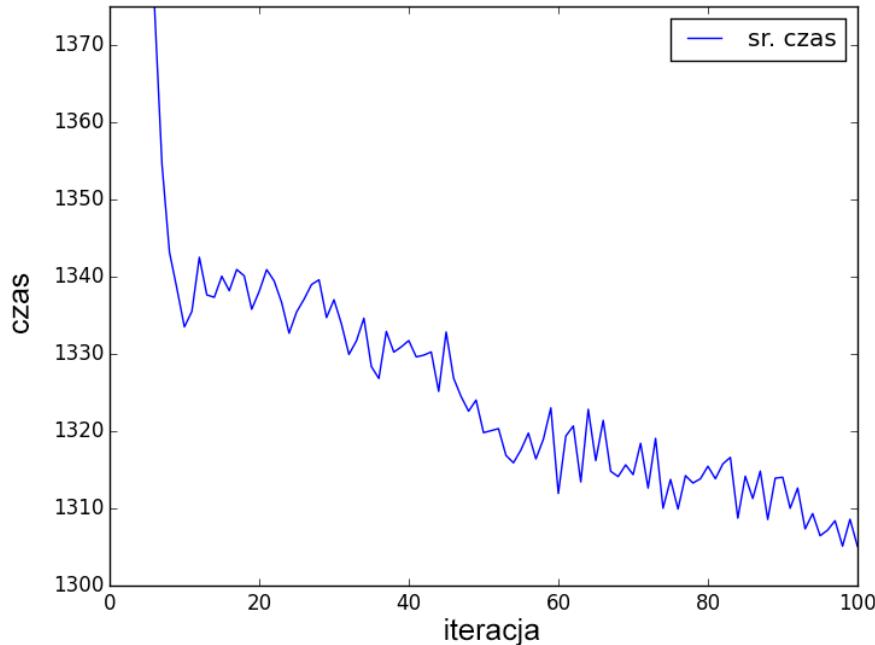
Warto również omówić kolejne kroki każdej iteracji, w zostały podane nazwy procesów z wykresu:

1. initial demand (iterationStartListeners),
2. execution (beforeMobSimListeners, mobSim, afterMobSimListeners),

3. scoring (scoring),
4. replanning (replanning),
5. analyses (iterationEndsListeners).

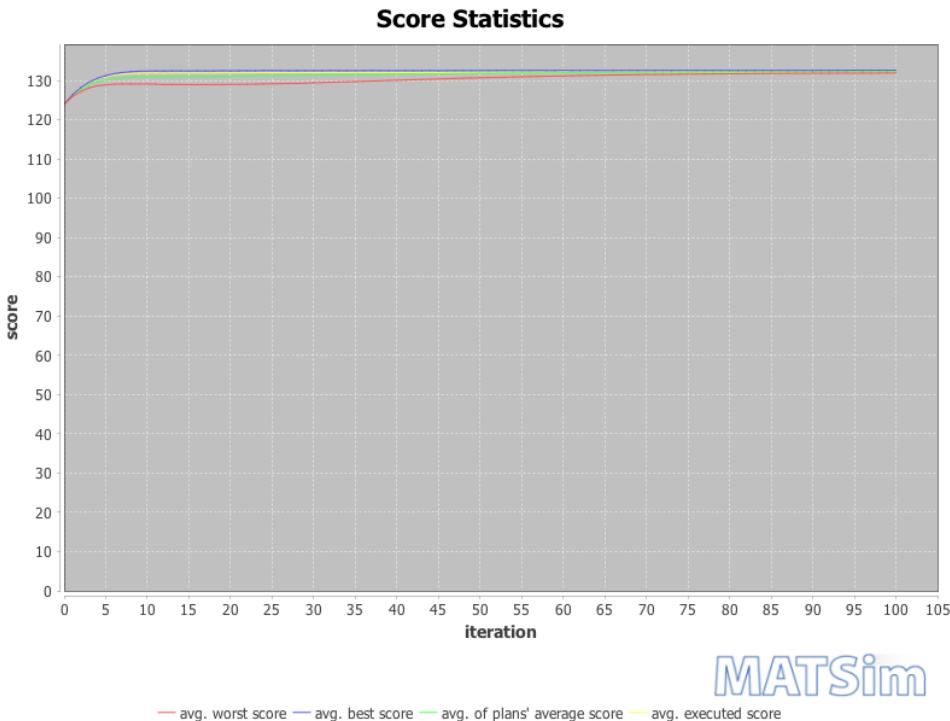
W pierwszym punkcie iteracji, agenci przygotowują się do swoich dni. Znając swoje zajęcia, planują oni drogę oraz dokładne godziny swoich akcji. Podczas fazy *execution* następuje sprawdzenie tych planów w akcji, czyli zachodzi właściwa symulacja. Podczas fazy *scoring* agenci zostają ocenieni na podstawie swojego dnia. Generalnie rzecz biorąc wyższy wynik uzyskują agenci, którzy jak najmniej czasu spędzili w podróży a najczęściej podczas zaplanowanych zajęć. Najważniejszym punktem symulacji jest jednak *replanning*. W tym punkcie agenci „wyciągają wnioski” ze swojego dnia i starają się nie popełnić tych samych błędów, wciąż dążąc do lepszego wyniku. Ostatni punkt iteracji dotyczy stworzenia analizy iteracji dostępnej dla użytkownika.

Analizując wykres czasu trwania kolejnych iteracji, na rysunku 3.6, możemy również odczytać czas całej symulacji, który wyniósł dokładnie 52min. 43sek. W przypadku analizy wielu (setek) sieci, czas więc jest wyraźnie zbyt długi. Przeanalizowaliśmy wpływ ilości iteracji na średni czas przejazdu (rysunek 3.7). Średni wynik¹ uzyskany przez agentów został natomiast przedstawiony na rysunku 3.8.



Rys. 3.7: Graf średniego czasu przejazdów dla stu iteracji

¹ang. score



Rys. 3.8: Graf wyników agentów dla stu iteracji

Na wykresach 3.7 i 3.8, widać, że wpływ iteracji, ma największe znaczenie w początkowych stadiach. W późniejszych iteracjach zmiana następuje dużo wolniej. Wiąże się to z wykorzystanym algorytmem występującym w fazie *replanning*.

Po analizie wykresów prezentowanych w sekcji ??, zdecydowaliśmy w projekcie na użycie 10 iteracji podczas analizy każdej sieci. Wartość ta optymalnie łączy zalety szybkiego spadku wartości średnich czasów podróży i czasu potrzebnego na obliczenia.

3.4 Algorytmy genetyczne

Projekt **The Apache Commons** jest tworzony przez *Apache Software Foundation*. Głównym celem projektu jest stworzenie wolnego oprogramowania do wielokrotnego użytku w języku *Java*. Projekt podzielony jest na trzy główne części: *proper*, *sandbox*, i *dormant* [14].

W przypadku głównej części **Apache Commons Proper** wyróżnia się wiele modułów różniących się funkcjonalnością i celem. **Apache Commons Math** jest modułem zapewniającym rozwiązywanie problemów głównie matematycznych i statystycznych. Znajduje się w nim implementacja podstawowej formy algorytmu genetycznego, którą rozszerzono w pracy.

3.5 Obsługa grafów

Ponieważ moduł do wizualizacji rozwiązań *MATSim* nie spełniał oczekiwania zarówno pod względem wydajności, jak i stabilności zdecydowano się na stworzenie autorskiej implementacji. W tym celu wykorzystano projekt **NetworkX**. Jest to biblioteka wykonana w języku *Python*, stworzona z myślą o grafach i sieciach. Jest ona dostarczana jako darmowe oprogramowanie na licencji *BSD-new* [15].

3.6 Technologie i metodologie programistyczne

Użyte w pracy technologie programistyczne są poniekąd wymuszone przez języki, w jakich zostały stworzone wykorzystywane rozwiązania pomocnicze. W przypadku simulatora *MATSim* jest to **Java**. Zdecydowano się natomiast na wykorzystanie języka **Python** głównie ze względu na przeważającą przewagę biblioteki *NetworkX* nad bibliotekami obsługującymi grafy w języku *Java*. Nie jest to wyjątek, gdyż ogólnie rozwiązania dostarczane dla *Python*'a, szczególnie w przypadku problemów akademickich, są dużo lepsze.

3.7 Wdrożenie

Ze względu na duże wymagania sprzętowe obliczeń, a zarazem ograniczoną przenośność rozwiązania z powodu skorzystania z języka *Python*, zdecydowano się na usprawnienie rozwiązania. Wykorzystując system *Linux Ubuntu* stworzono maszynę wirtualną, spełniającą wszystkie wymagania do uruchomienia aplikacji. Dzięki temu stało się możliwe wykorzystanie innych komputerów oprócz tego, na którym zostało stworzone rozwiązanie. Podczas badań wykorzystywany był system **Linux Ubuntu 12.04 LTS**[20].

3.8 System obliczeniowy

Korzystając z rozwiązania opisanego w 3.7, dzięki któremu projekt jest możliwy do zainstalowania na innych maszynach, do obliczeń wykorzystano zewnętrznego dostawcę mocy obliczeniowej. Przy wyborze decydującym czynnikiem była cena rozwiązania, co w przypadku chmury **Microsoft Windows Azure** [21], pozwoliło na darmowe rozwiązanie.

3.8. SYSTEM OBLICZENIOWY

Rozdział 4

Opis projektu

W niniejszym rozdziale opisujemy strukturę projektu oraz sposób jego uruchomienia. Ponieważ przeprowadzane przez projekt obliczenia wymagają wiele czasu, zdecydowaliśmy się na pominięcie interfejsu użytkownika przy projektowaniu aplikacji. Całość jest obsługiwana przez plik konfiguracyjny, który zostaje wczytany na początku działania programu i za jego pomocą kontrolujemy przebieg obliczeń. Pomimo dwóch osobnych technologii, w których został wykonany projekt, są one od siebie zależne. Całość jest obsługiwana przez projekt w Java, skrypty w Pythonie jedynie wspierają niektóre procesy.

4.1 Dane wejściowe

Przykładowy plik konfiguracyjny dla projektu z opisem jego funkcji przedstawiono na listingu 4.1.

```
<?xml version="1.0" encoding="UTF-8"?>
<config>
    <project>
        <name>siouxfalls</name>
        <output-dir>../output/</output-dir>
        <threads>4</threads>
        <log-level>INFO</log-level>
        <python-path>/usr/local/bin/python</python-path>
        <python-main>../python/ms/call_center.py</python-main>
        <java-path>/usr/bin/java</java-path>
        <matsim-jar>../matsim/matsim-r31927mod.jar</matsim-jar>
        <matsim-xmx>2g</matsim-xmx>
    </project>
    <scenario>
        <config>../scenarios/siouxfalls/config.xml</config>
        <network>../scenarios/siouxfalls/network.xml</network>
        <population>../scenarios/siouxfalls/population.xml</population>
        <facilities>../scenarios/siouxfalls/facilities.xml</facilities>
        <iterations>9</iterations>
    </scenario>
    <genetics>
```

4.1. DANE WEJŚCIOWE

```
<population-size>24</population-size>
<max-generations>200</max-generations>
<elitism-rate>2</elitism-rate>
<crossover-rate>1.0</crossover-rate>
<mutation-rate>0.8</mutation-rate>
<tournament-arity>4</tournament-arity>
</genetics>
</config>
```

Listing 4.1: Plik konfiguracyjny projektu

Jak widać, konfiguracja dzieli się na trzy podstawowe grupy:

- project,
- scenario,
- genetics.

Grupa *project* odpowiada za główne ustawienia całego projektu, w *scenario* znajdziemy ustawienia dotyczące symulacji przeprowadzanej przez MATSim, natomiast sekcja *genetics* zawiera ustawienia dotyczące algorytmu genetycznego.

Wyjaśnienie opcji *project*:

- name — nazwa projektu, używana jako katalog wyjściowy,
- output dir — katalog, gdzie zapisujemy wyniki,
- threads - ilość wątków, które mają być użyte podczas obliczeń,
- log level - poziom logowania Log4J¹,
- python path - ścieżka instalacji Pythona,
- python main - folder ze skryptami pomocniczymi,
- java-path - ścieżka instalacji Javy,
- matsim-jar - ścieżka do biblioteki MATSim,
- matsim-xmx - maksymalna pamięć RAM dostępna dla symulatora MATSim.

Wyjaśnienie opcji *scenario*:

- config - ścieżka dostępu pliku konfiguracyjnego scenariusz MATSim,
- network - ścieżka dostępu pliku z siecią wejściową scenariusza,

¹zewnętrzna biblioteka do logowania, dostępne poziomy: DEBUG, INFO, WARN, ERROR i FATAL

- population - ścieżka dostępu pliku z populacją scenariusza,
- facilities - ścieżka dostępu pliku z budynkami scenariusza,
- iterations - ilość iteracji symulacji MATSima.

Wyjaśnienie opcji *genetics*:

- population size - rozmiar populacji,
- max generations - ilość testowanych generacji (warunek stopu),
- elitism rate - ilość najlepszych chromosomów biorących udział w kolejnej iteracji,
- crossover rate - szansa na krzyżowanie osobników z poprzedniej populacji przed dodaniem ich do kolejnej generacji,
- mutation rate - szansa na mutację pojedynczego genu wybranych osobników,
- tournament rate - ilość osobników biorących udział w turnieju.

4.2 Ustawienia projektu

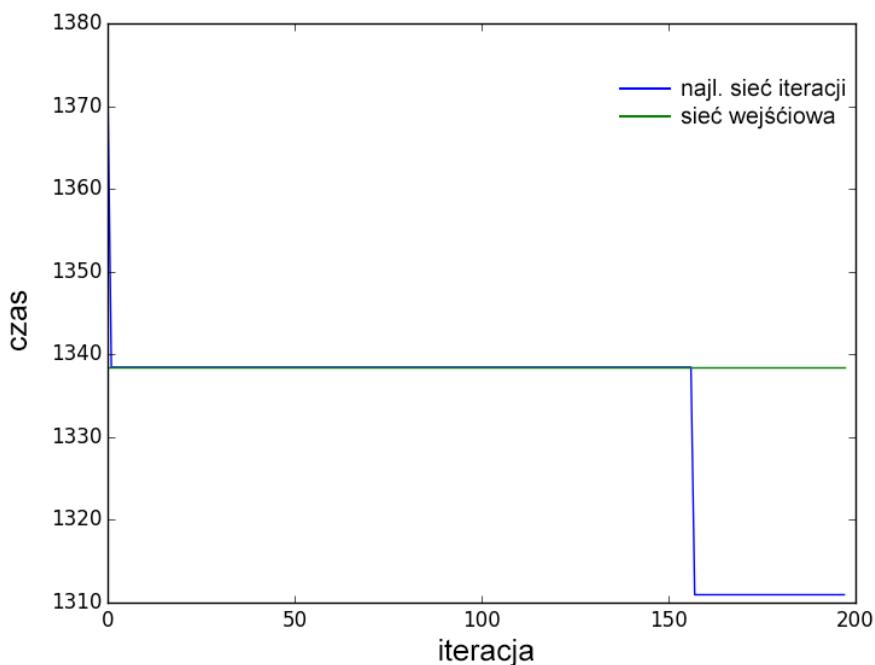
Podczas badań wykorzystaliśmy ustawienia algorytmu genetycznego zgodne z listiniem ???. Wartości zostały dobrane na podstawie doświadczenia nabytego w trakcie studiów literaturowych.

```
<genetics>
    <population-size>24</population-size>
    <max-generations>200</max-generations>
    <elitism-rate>2</elitism-rate>
    <crossover-rate>1.0</crossover-rate>
    <mutation-rate>0.8</mutation-rate>
    <tournament-arity>4</tournament-arity>
</genetics>
```

Listing 4.2: Ustawienia algorytmu genetycznego podczas badań

4.3 Wyniki

Wykres przedstawiający zestawienie wyniku najlepszej sieci wyłonionej w danej iteracji, do wyniku sieci wejściowej przedstawiamy na rysunku 4.1.



Rys. 4.1: Graf wyników algorytmu genetycznego

Ponieważ w projekcie wykorzystany został elitarny model populacji algorytmu genetycznego, powyższy wykres nie przedstawia praktycznie żadnych zmian w populacji, która została ulepszona. Wyniki wszystkich chromosomów były jednak zachowywane w bazie. Dzięki temu możemy przedstawić więcej sieci spełniających założenia projektowe.

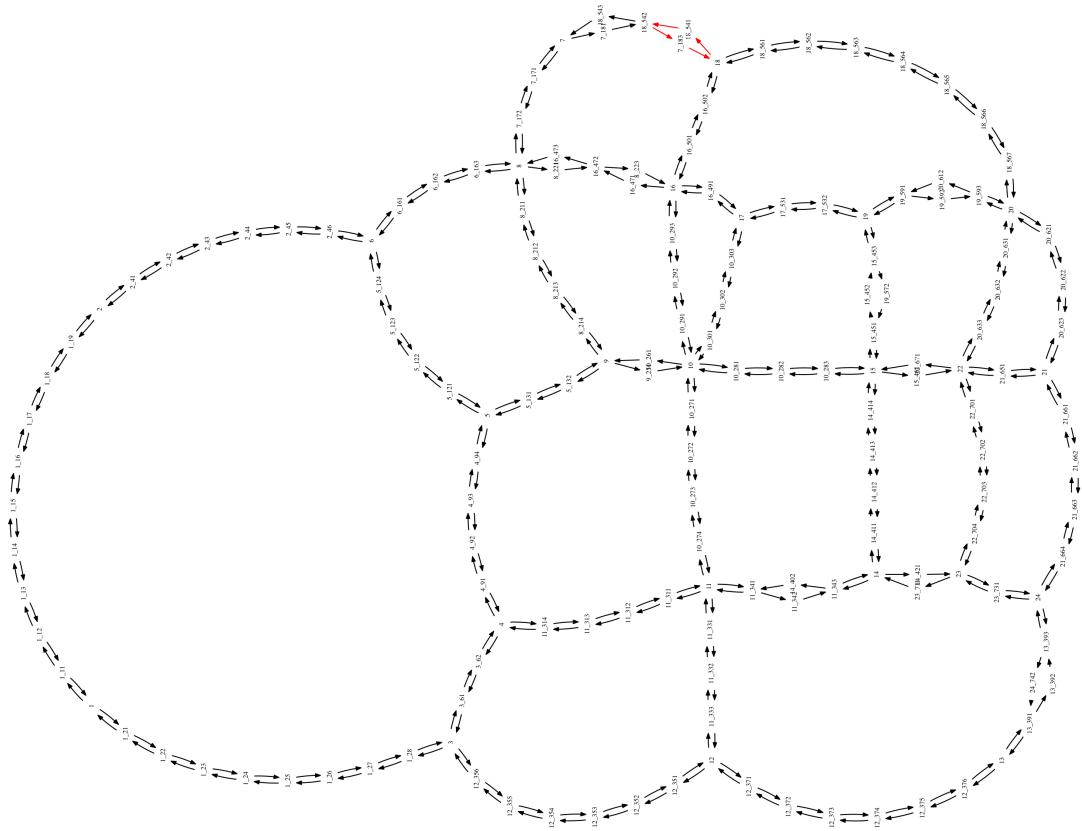
Najlepsze wyniki symulacji przedstawiamy w tabeli 4.1. Przedstawione są tutaj wszystkie sieci, które uzyskały wynik lepszy od sieci wejściowej, tj. 1338.4505528474617. Od teraz, będziemy odnosić się do tych sieci używając ich numeru identyfikacyjnego (pierwsza kolumna).

Tablica 4.1: Zbiór najlepszych otrzymanych sieci

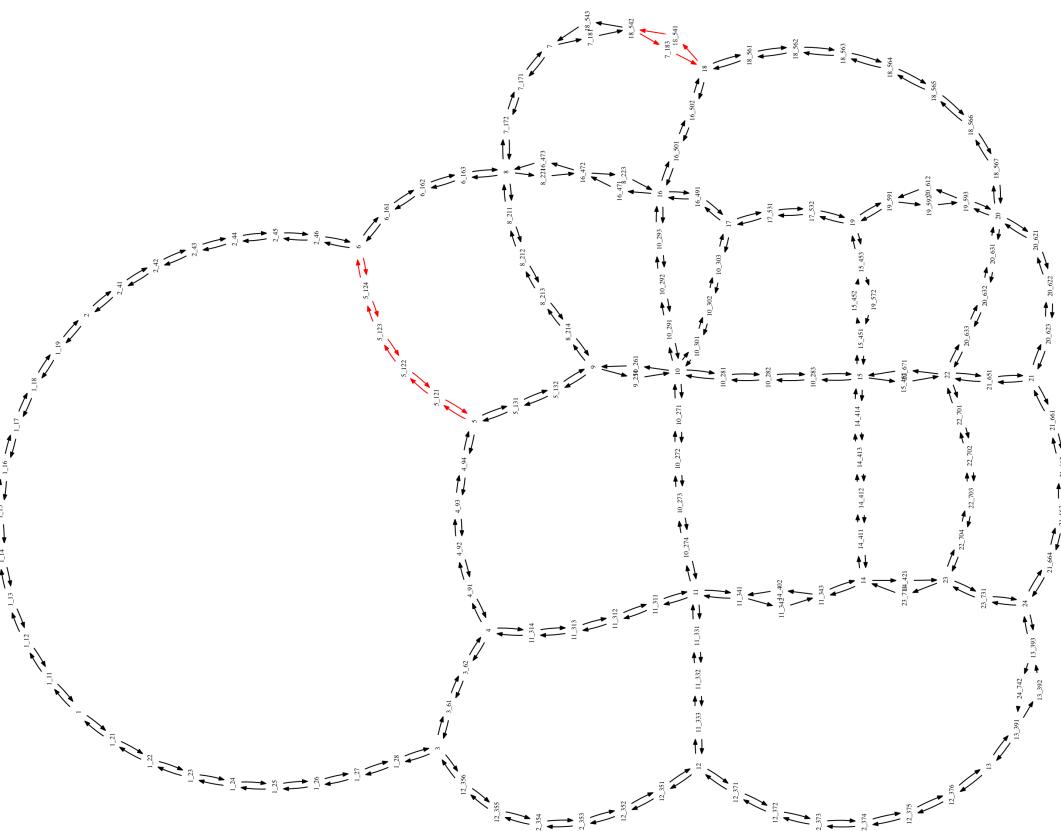
| ID | Reprezentacja binarna | Wynik sieci |
|----|--|------------------|
| 1 | 011 111 | 1310.95255617644 |
| 2 | 011 111 | 1312.64143383664 |
| 3 | 011 111 | 1312.73299250981 |
| 4 | 011 111 | 1313.75345975508 |
| 5 | 011 111 | 1318.77194744977 |
| 6 | 011 111 | 1320.54353227916 |
| 7 | 011 111 | 1322.64708120319 |
| 8 | 011 111 | 1323.53190464867 |
| 9 | 011 111 | 1325.07095470218 |
| 10 | 011 111 | 1327.94727737487 |
| 11 | 011 111 | 1329.46298894305 |
| 12 | 011 111 | 1332.87335631911 |
| 13 | 011 111 | 1334.28138152419 |
| 14 | 011 111 | 1334.48972773749 |
| 15 | 011 111 | 1335.74516109856 |
| 16 | 011 111 | 1335.96306027821 |
| 17 | 011 111 | 1336.65051123529 |

4.3. WYNIKI

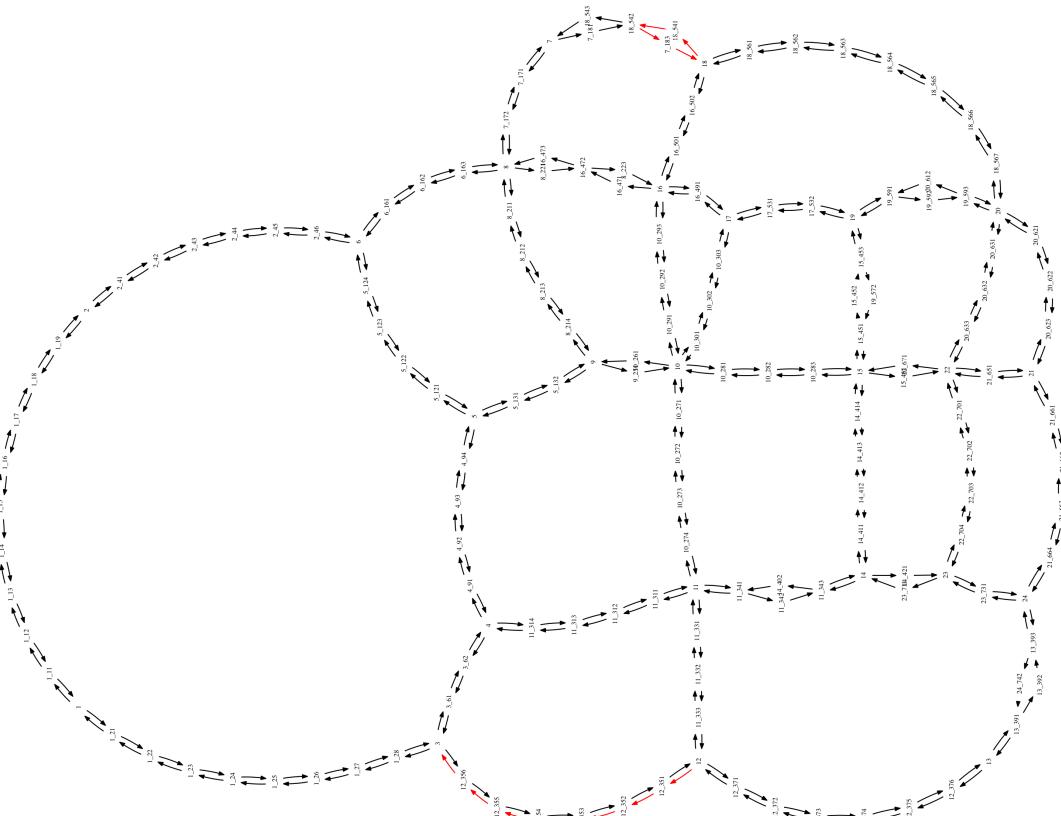
Na rysunkach 4.2 - 4.18 prezentujemy grafy uzyskanych sieci, dla wizualizacji rozwiązań. **W celu uproszczenia odnajdowania usuniętych węzłów, są one zaznaczone kolorem czerwonym** Tak, jak w poprzednim przypadku, sieci zostały obrócone o 90° w lewo. W niektórych przypadkach wycięte węzły są trudno widoczne, z powodu nikiej długości strzałki. Dotyczy to przede wszystkim przykładów: 4.6, 4.8, 4.10, 4.11, 4.12, 4.13, 4.14, 4.15, 4.17. W tych sieciach zostały wycięte drogi w pobliżu węzła o numerze 15.



Rys. 4.2: Sieć miasta Sioux Falls, rozwiązanie nr. 1

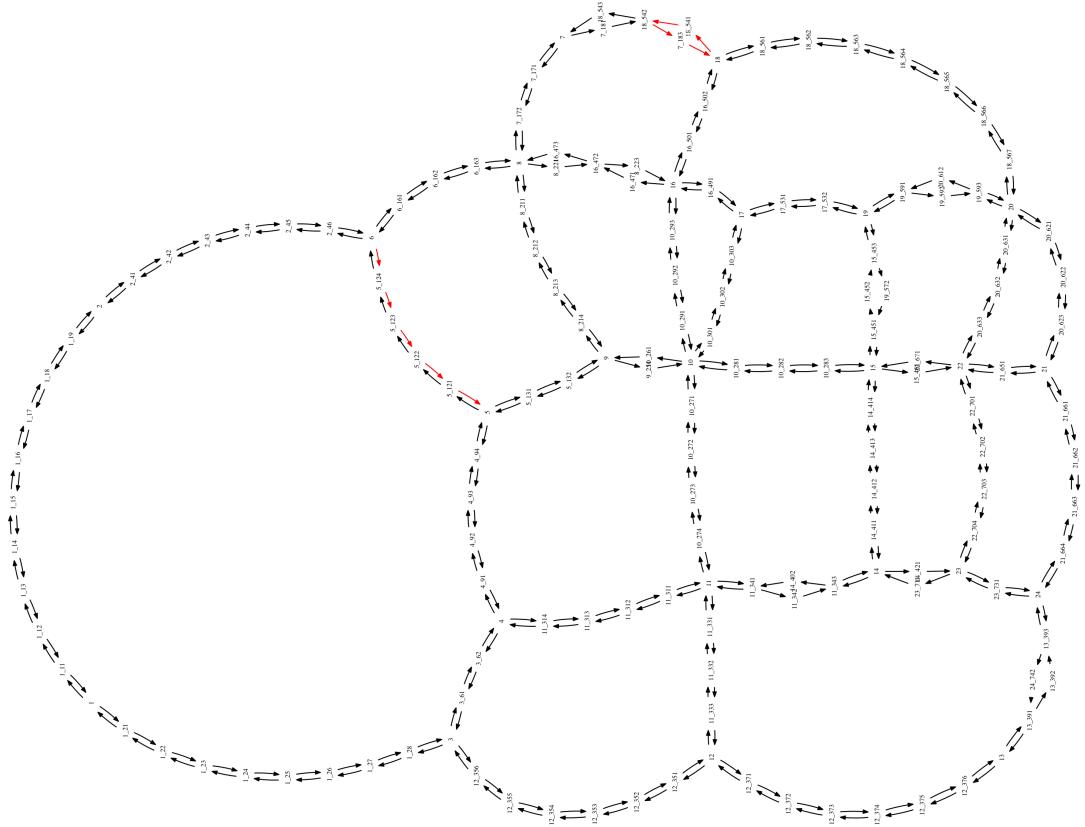


Rys. 4.3: Sieć miasta Sioux Falls, rozwiązanie nr. 2

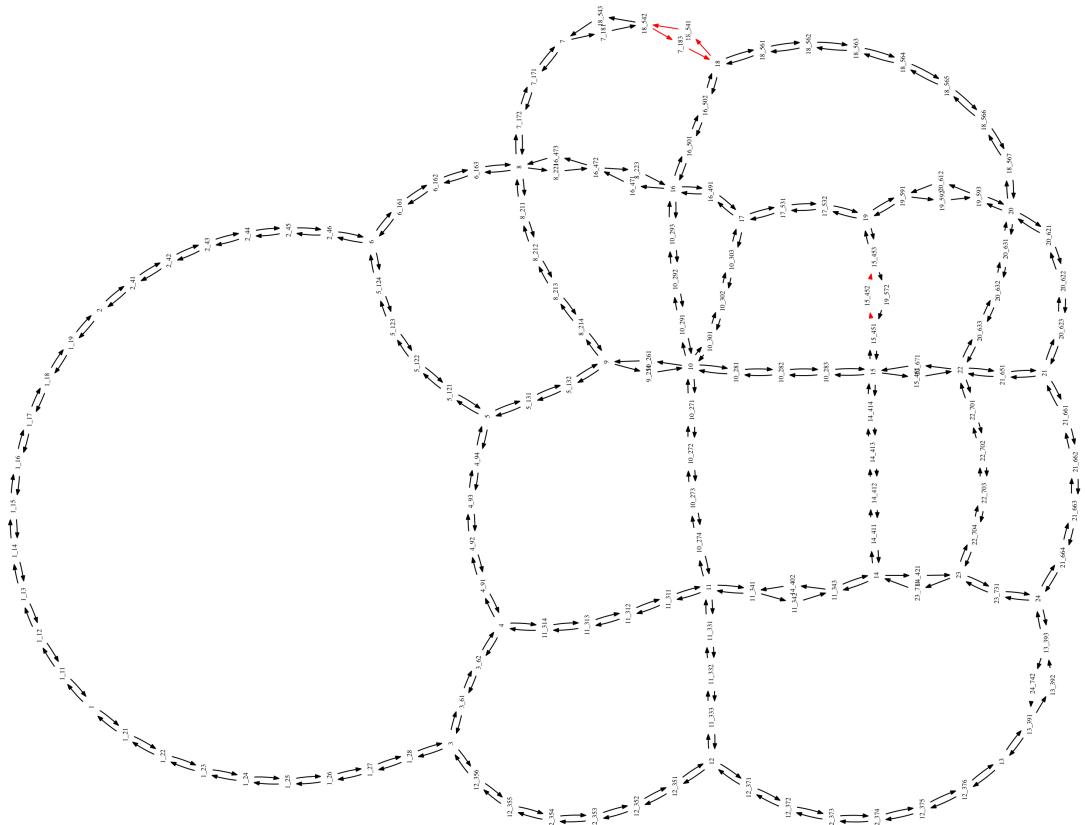


Rys. 4.4: Sieć miasta Sioux Falls, rozwiązanie nr. 3

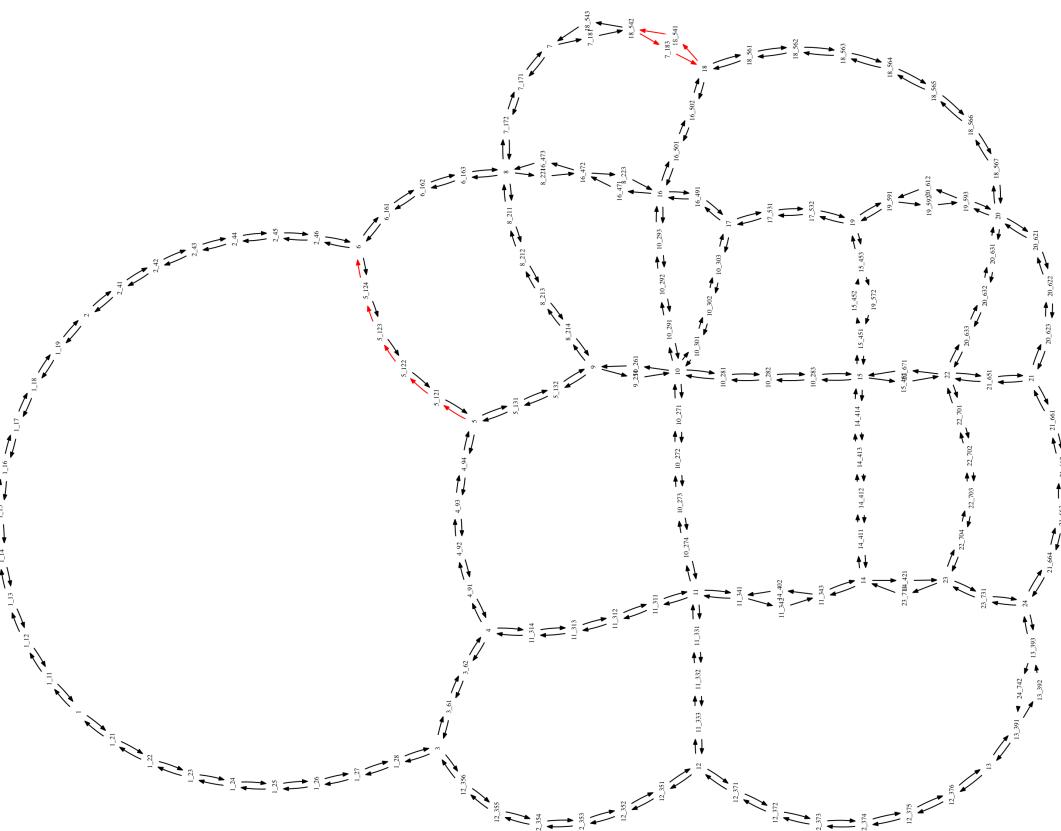
4.3. WYNIKI



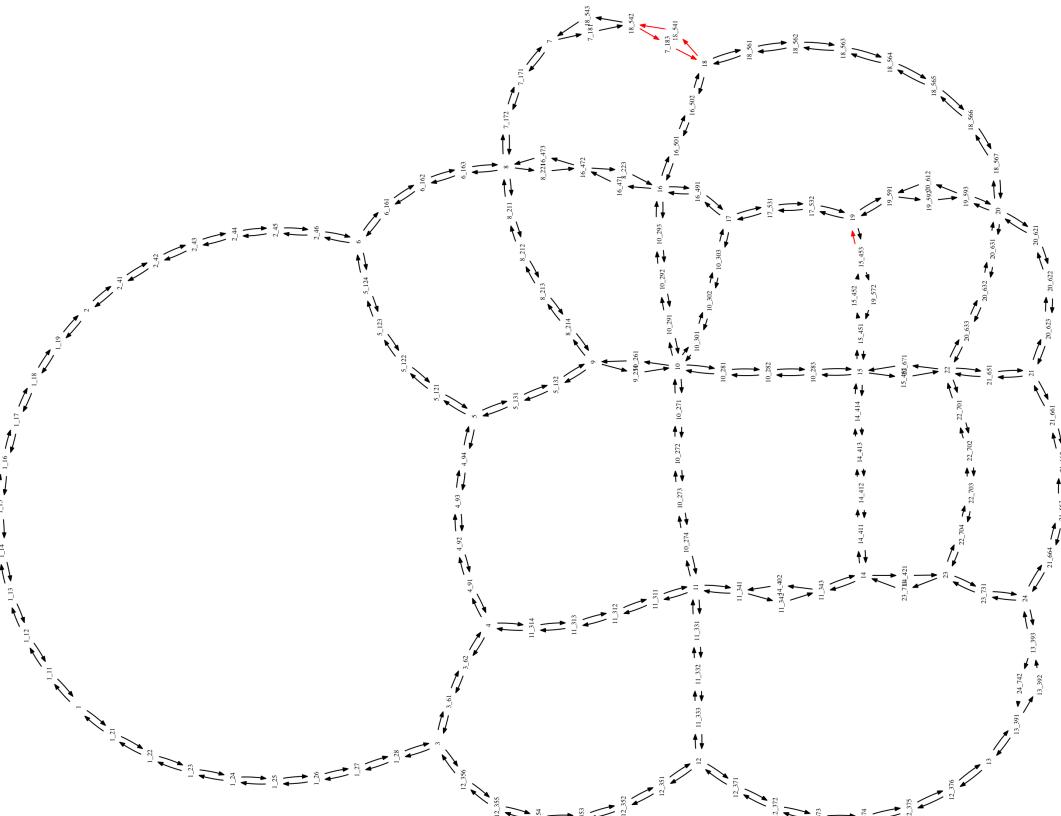
Rys. 4.5: Sieć miasta Sioux Falls, rozwiązanie nr. 4



Rys. 4.6: Sieć miasta Sioux Falls, rozwiązańe nr. 5

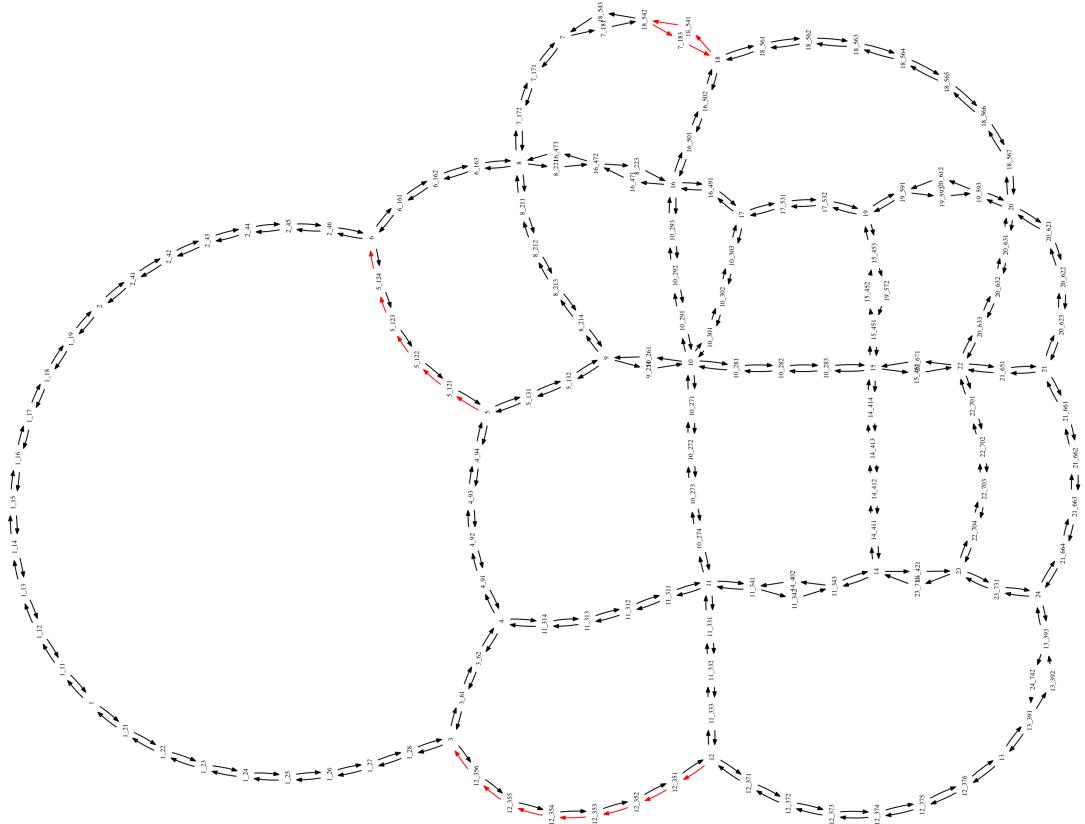


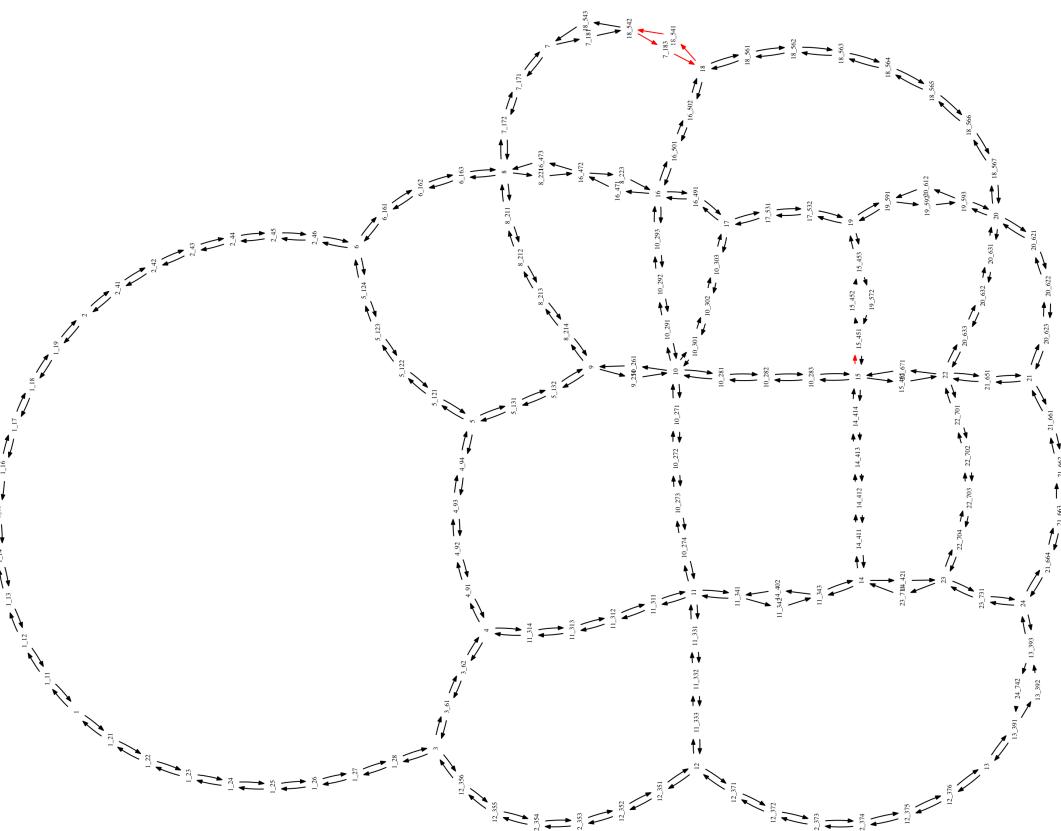
Rys. 4.7: Sieć miasta Sioux Falls, rozwiązanie nr. 6



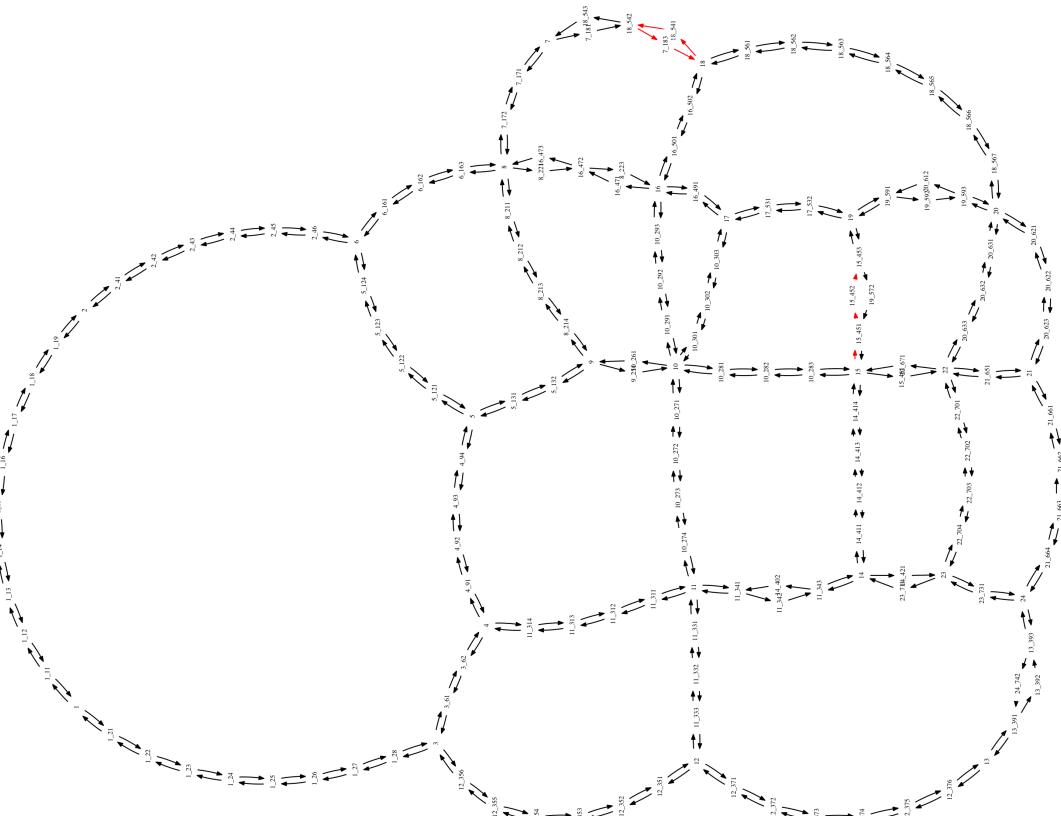
Rys. 4.8: Sieć miasta Sioux Falls, rozwiązanie nr. 7

4.3. WYNIKI



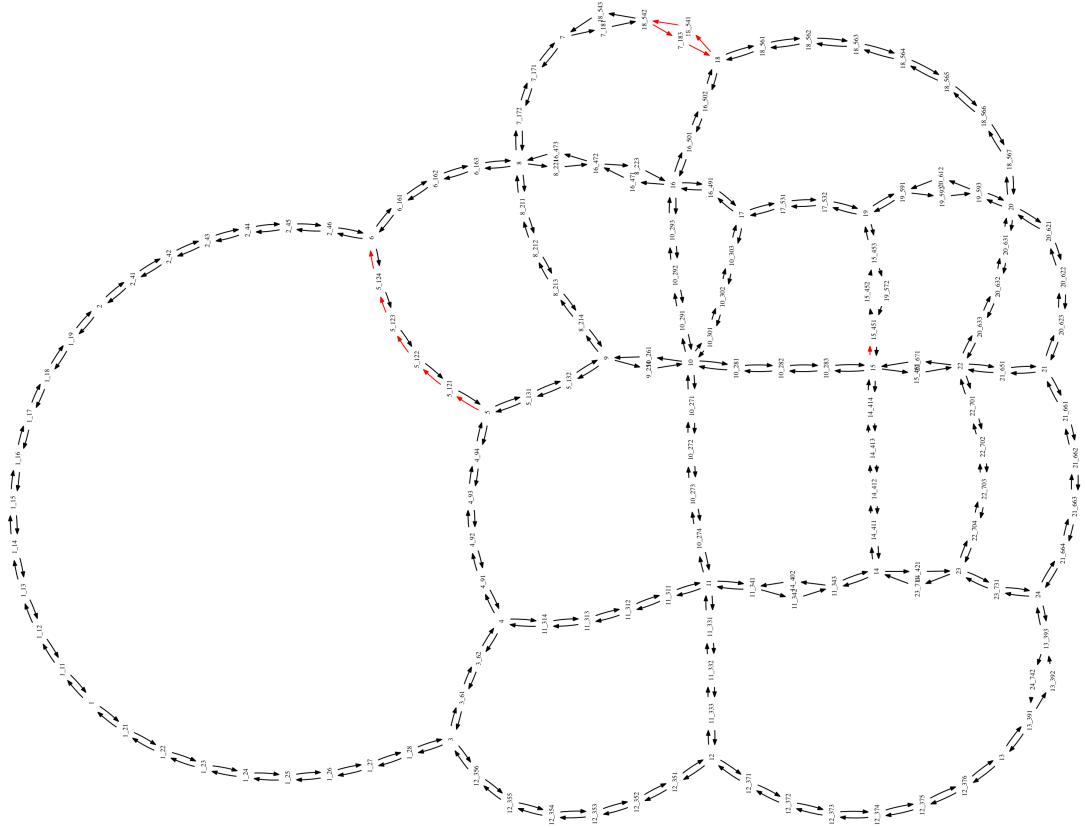


Rys. 4.11: Sieć miasta Sioux Falls, rozwiązanie nr. 10

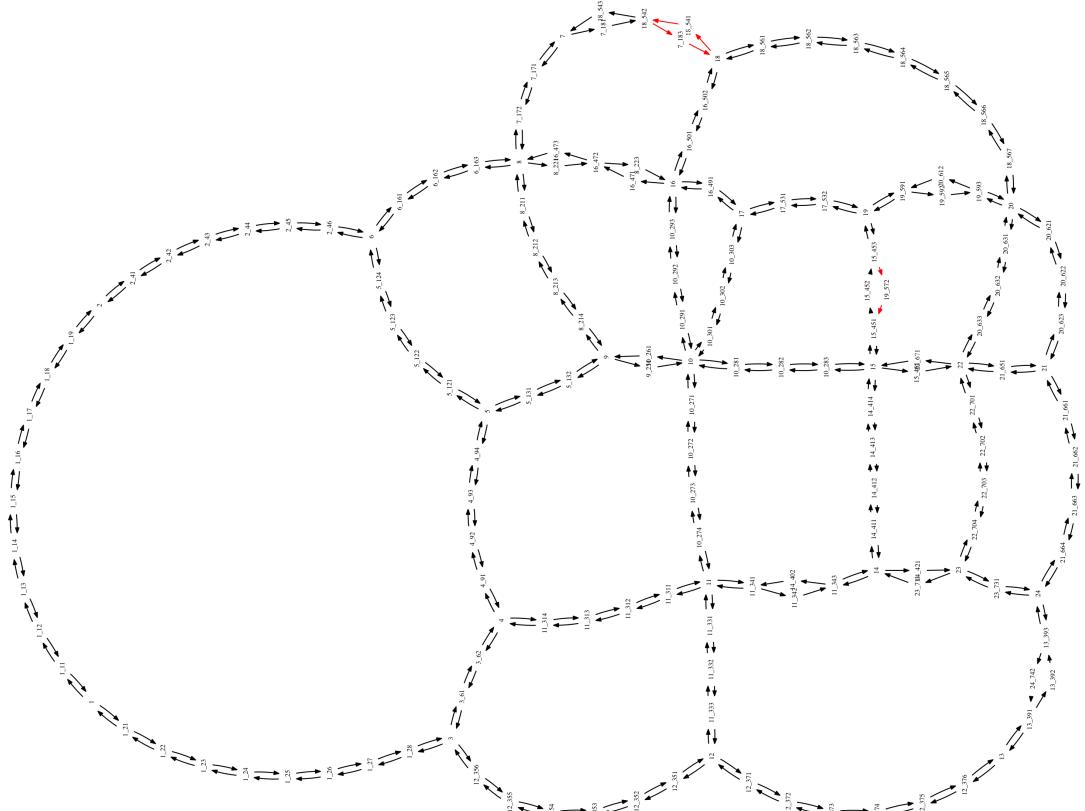


Rys. 4.12: Sieć miasta Sioux Falls, rozwiązanie nr. 11

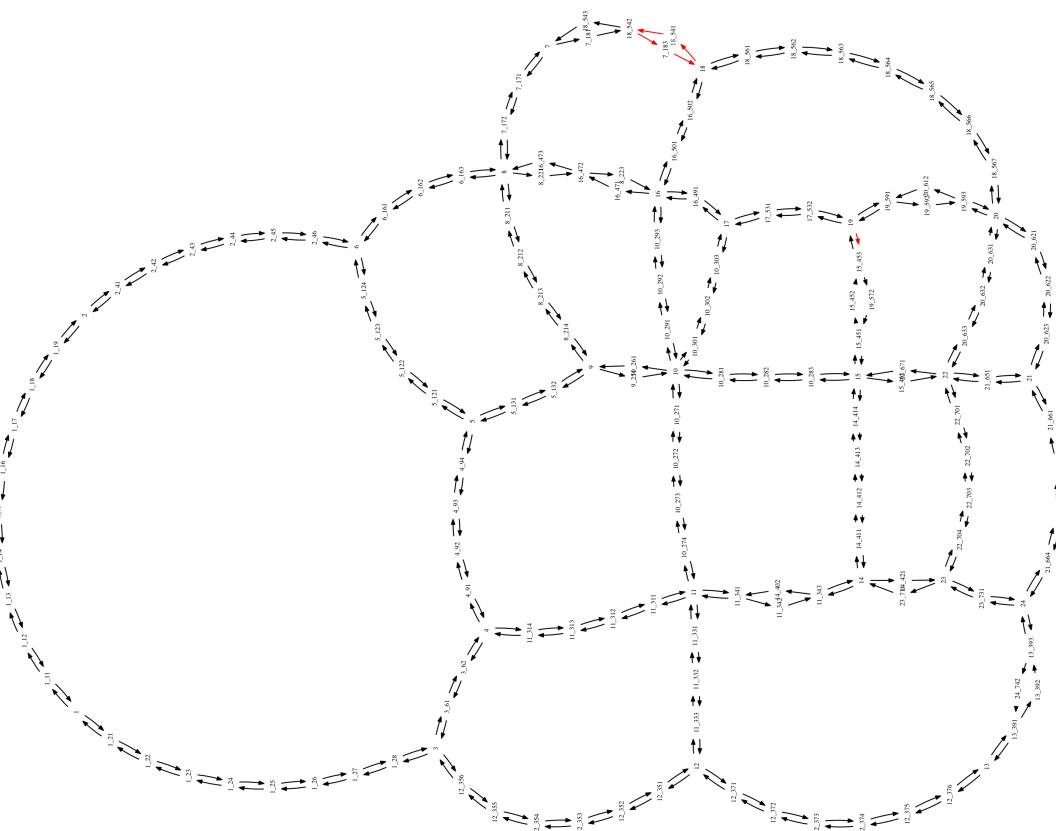
4.3. WYNIKI



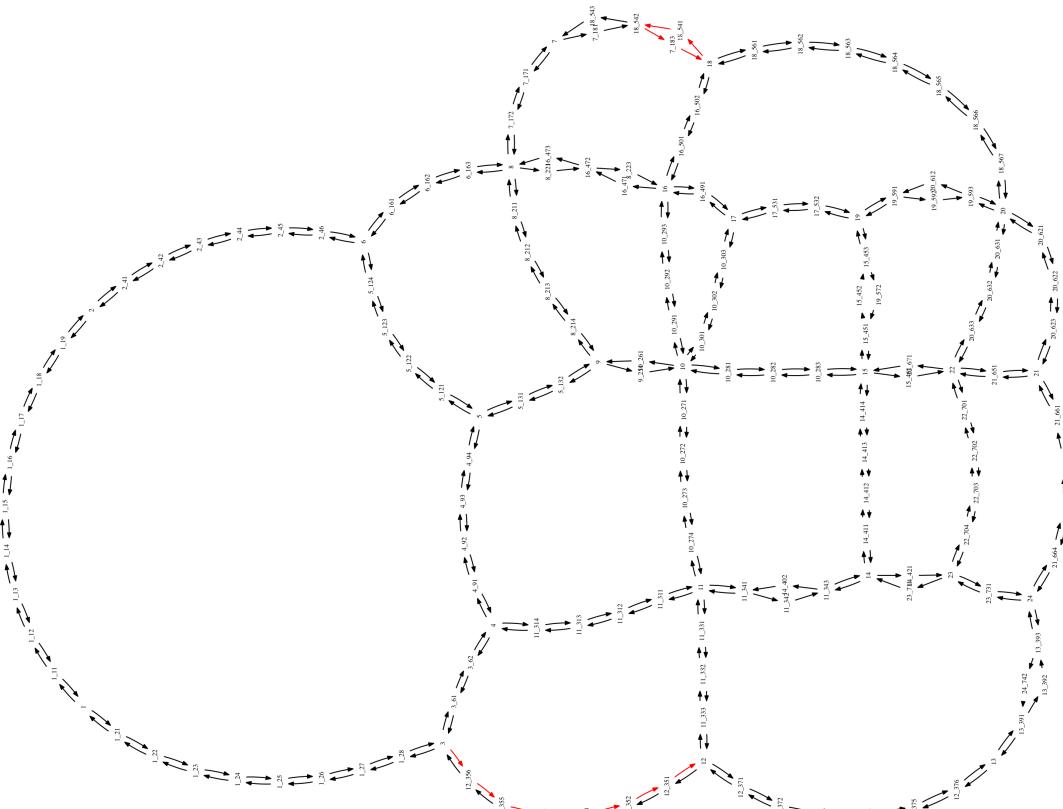
Rys. 4.13: Sieć miasta Sioux Falls, rozwiązanie nr. 12



Rys. 4.14: Sieć miasta Sioux Falls, rozwiążanie nr. 13

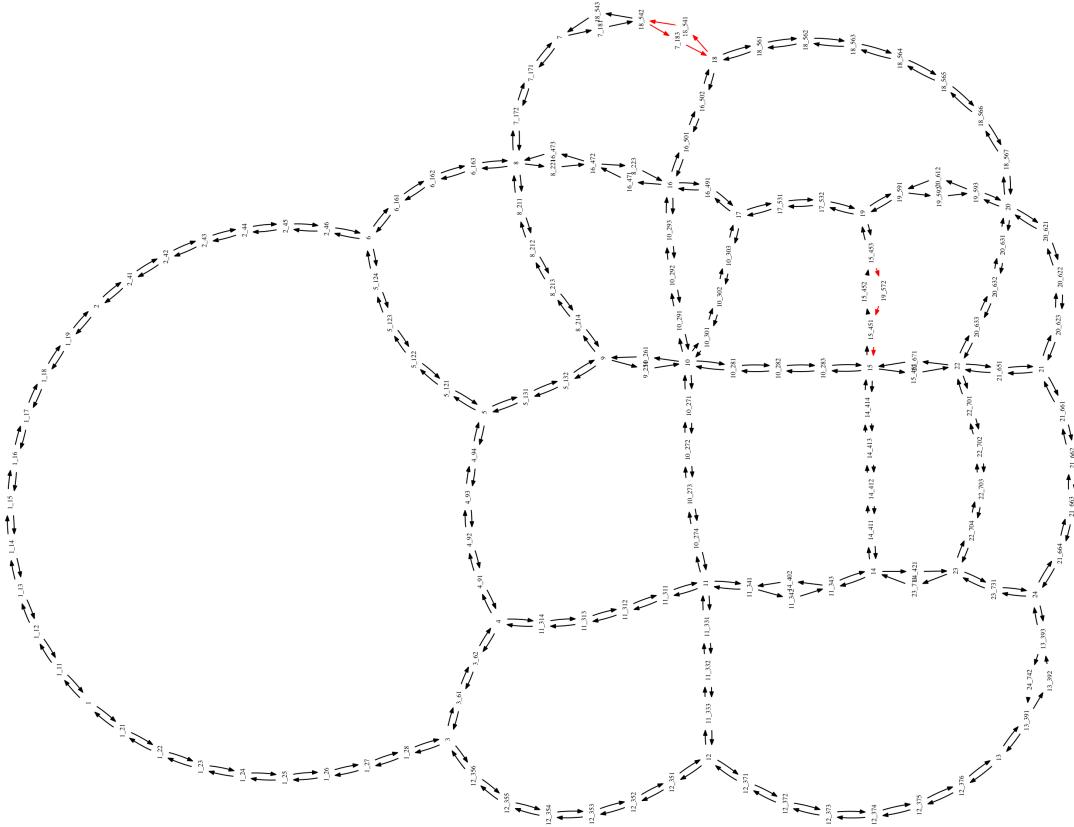


Rys. 4.15: Sieć miasta Sioux Falls, rozwiązanie nr. 14

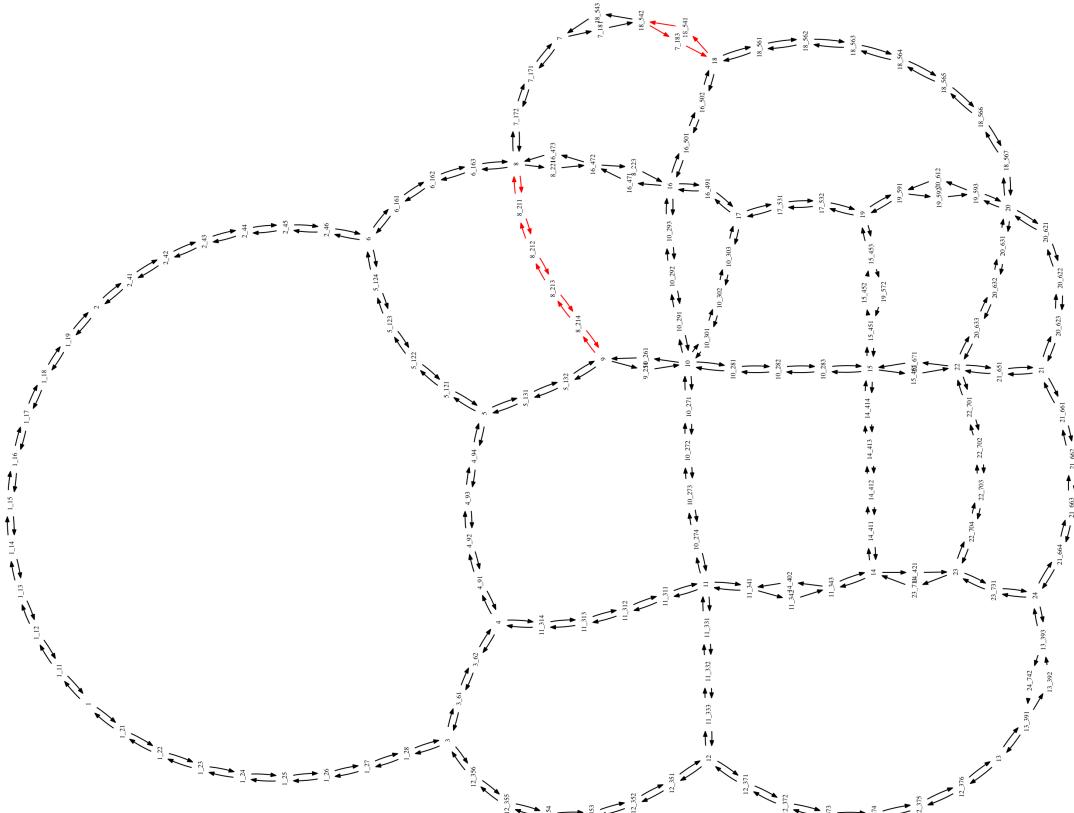


Rys. 4.16: Sieć miasta Sioux Falls, rozwiązanie nr. 15

4.3. WYNIKI



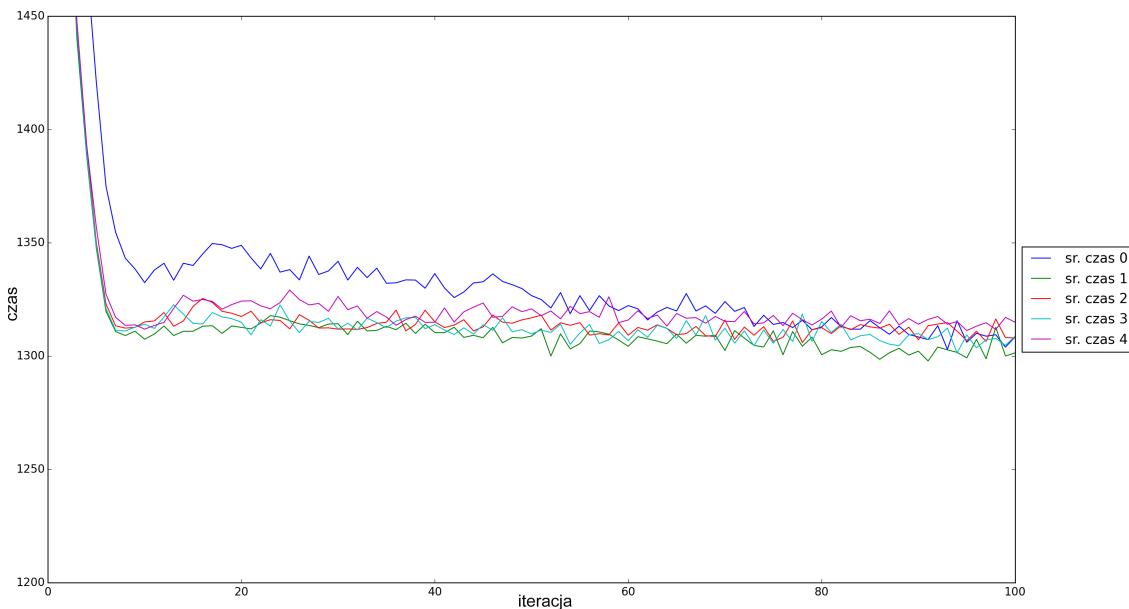
Rys. 4.17: Sieć miasta Sioux Falls, rozwiązańe nr. 16



Rys. 4.18: Sieć miasta Sioux Falls, rozwiązańe nr. 17

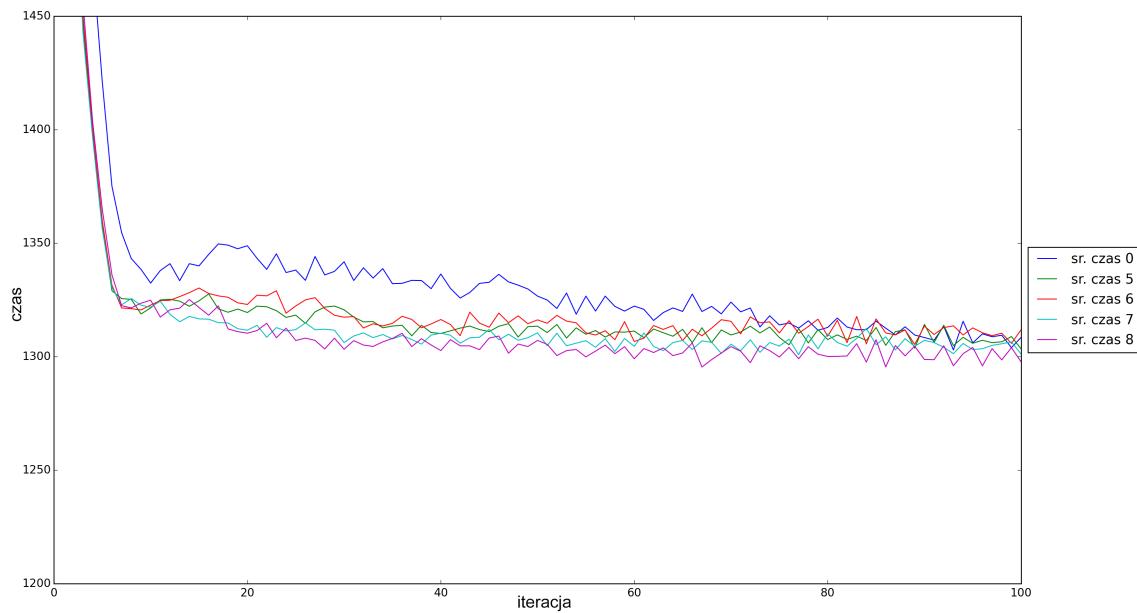
4.4 Analiza symulacji

Zgodnie z wcześniejszymi założeniami, podczas poszukiwań wybrałem ustawienie 10 iteracji symulacji do oceny sieci. W celu zweryfikowania wyników, dla najlepszych 17 sieci przeprowadziliśmy ponowną symulację dla 100 iteracji. Wyniki te porównujemy z wynikami sieci wejściowej. Rysunki 4.19 - 4.23 przedstawiają wyniki tych porównań. Dla poprawienia czytelności wykresy zostały rozdzielone. Numer sieci 0 odnosi się do sieci wejściowej, natomiast pozostałe numery, do poszczególnych numerów ID sieci wybranych przez algorytm.

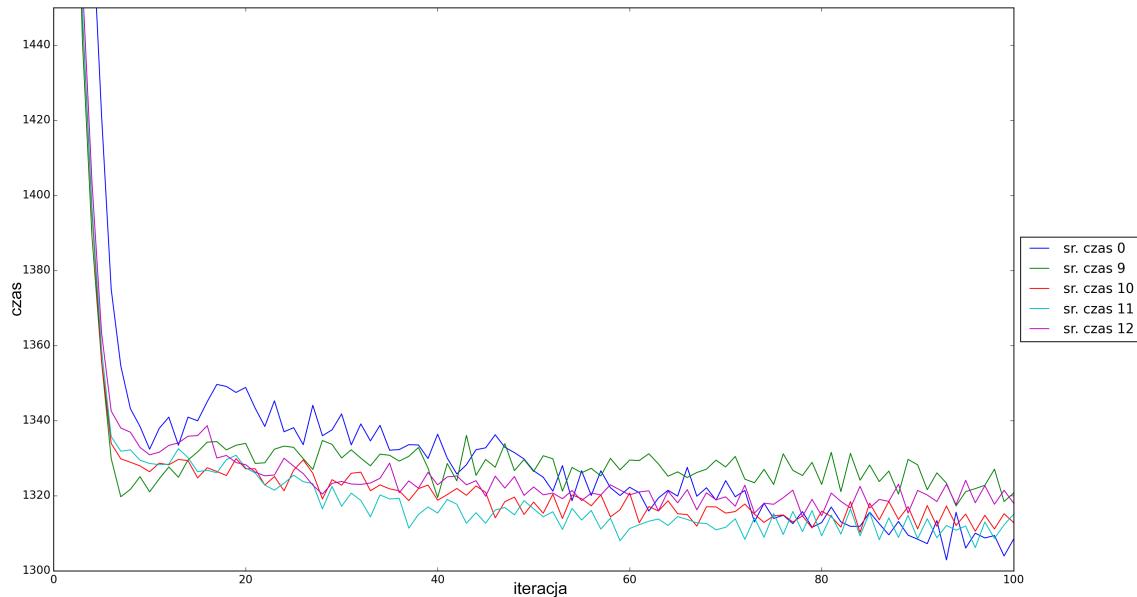


Rys. 4.19: Wykres zmiany średniego czasu przejazdu od iteracji dla sieci wynikowych z ujęciem sieci wejściowej

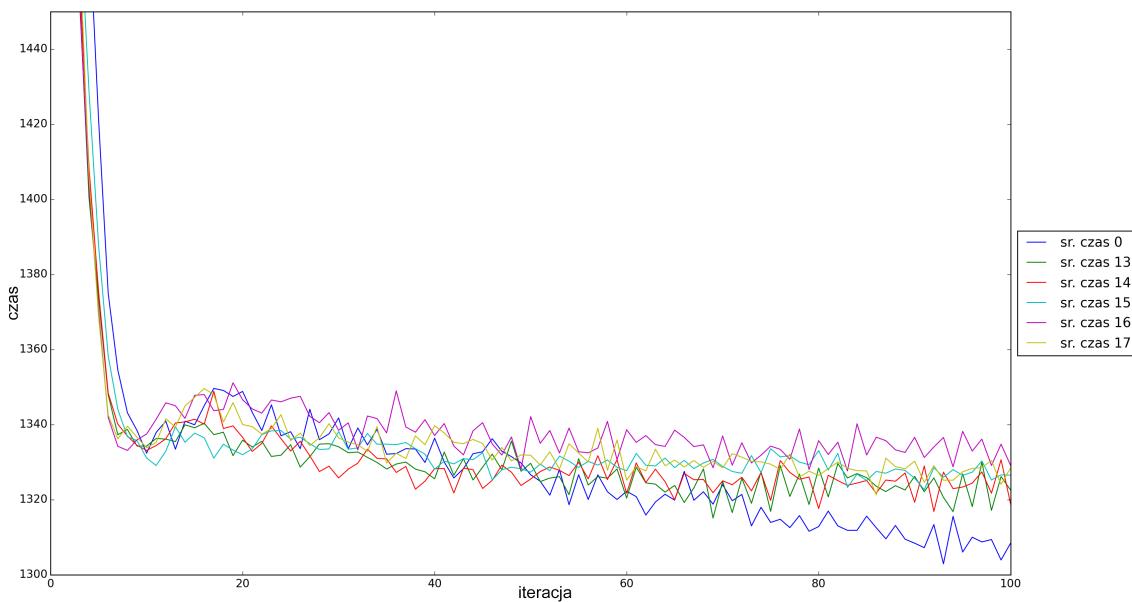
4.4. ANALIZA SYMULACJI



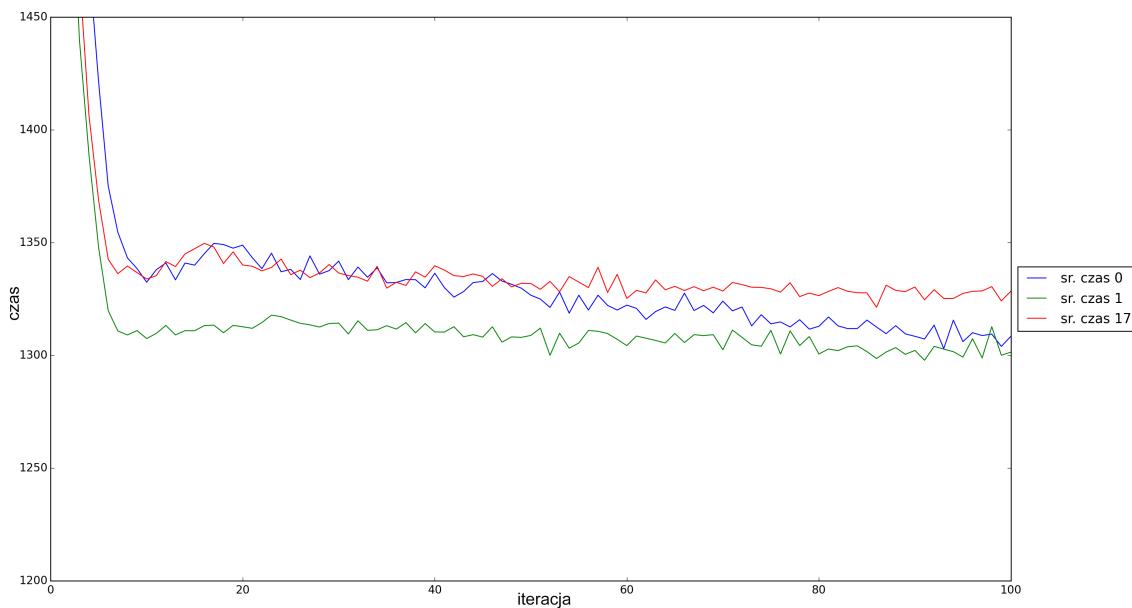
Rys. 4.20: Wykres zmiany średniego czasu przejazdu od iteracji dla sieci wynikowych z ujęciem sieci wejściowej



Rys. 4.21: Wykres zmiany średniego czasu przejazdu od iteracji dla sieci wynikowych z ujęciem sieci wejściowej



Rys. 4.22: Wykres zmiany średniego czasu przejazdu od iteracji dla sieci wynikowych z ujęciem sieci wejściowej



Rys. 4.23: Wykres zmiany średniego czasu przejazdu od iteracji dla sieci najlepszej z ujęciem sieci wejściowej

4.4. ANALIZA SYMULACJI

Rozdział 5

Podsumowanie

Uzyskane wyniki potwierdzają możliwość dostosowania sieci drogowej do zadanych potrzeb mieszkańców przy pomocy algorytmu genetycznego. Dla zadanej sieci udało się zoptymalizować ruch drogowy, stosując zamknięcie określonych węzłów. Trzeba jednak pamiętać, że wynik jest czysto teoretyczny. Symulacja nie odzwierciedla prawdziwego ruchu drogowego w mieście a ponadto zastosowaliśmy uproszczenie, w którym nie są obecne wszystkie węzły rzeczywistej sieci drogowej. Warto zaznaczyć, że w projekcie można zastosować inną funkcję celu dla oceny osobników. Wpływ to oczywiście bezpośrednio na otrzymane wyniki.

5.1 Dyskusja wyników

Gdyby założyć, że system ten miałby dostarczyć pewnej sugestii w sprawie podjęcia decyzji dotyczącej usprawnienia ruchu, spełnia on swoje zadanie. Przeszukiwanie przestrzeni rozwiązań wykorzystując algorytmy genetyczne, dostarcza wyników, które mogą być przedmiotem dalszej analizy.

Symulator MATSim zadziałał zgodnie z oczekiwaniemi, wykazując dużą zdolność adaptacji agentów do sieci. Przypadek ten można opisać, zadając sobie pytanie, skąd właściwie biorą się korki i zatory drogowe? Są one zwykle efektem braku poprawnego planu podróży uczestników ruchu. Symulator potrafi znaleźć taki plan, który w sposób idealny, dobiera czas podróży oraz najlepszą drogę. Jest to oczywiście możliwe dzięki powtarzającym się w każdej iteracji **tym samym** warunkom drogowym, które panują w sieci. Scenariusz jest bowiem pozbawiony losowych wypadków oraz czynników ludzkich. Nie jest to więc sytuacja rzeczywista.

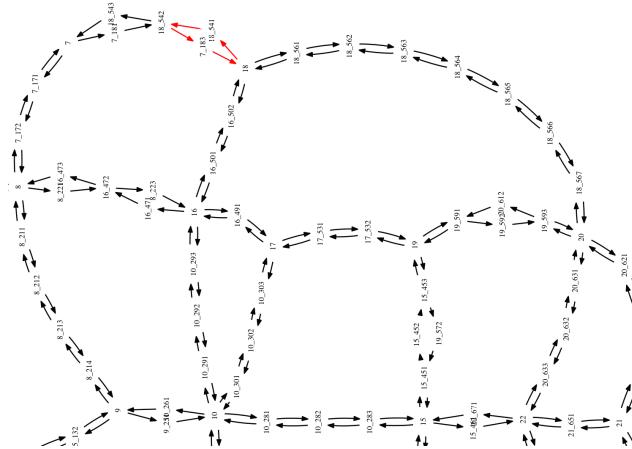
Mając na uwadze ten „idealny” aspekt symulacji, wybory w iteracjach wcześniejszych są mniej idealne, można powiedzieć, bardziej ludzkie. Wykresy 4.19 - 4.23 wyraźnie pokazują, że dla zmodyfikowanej sieci, znacznie trudniej jest popełnić błąd przy

planowaniu trasy, mając ograniczoną wiedzę na temat warunków drogowych. Wymuszenie określonych dróg na agentach pozwoliło im szybciej dokonywać prawidłowych wyborów.

W przypadku długiej symulacji, część sieci zmodyfikowanych osiągała gorszy rezultat niż sieć wejściowa. Oprócz wyżej wymienionych czynników, musimy bowiem pamiętać, że generalnie większa liczba węzłów oznacza większe możliwości przepływowne sieci. Efekt dłuższej symulacji można porównać z dopasowaniem sieci drogowej do potrzeb zadanego modelu agentów. Taka sieć spełniałaby idealnie wymogi danej symulacji. Nie sprawdziłaby się, jednak gdyby model został zmieniony. W przypadku długiej nauki agentów, na temat danej sieci, mamy sytuację odwrotną.

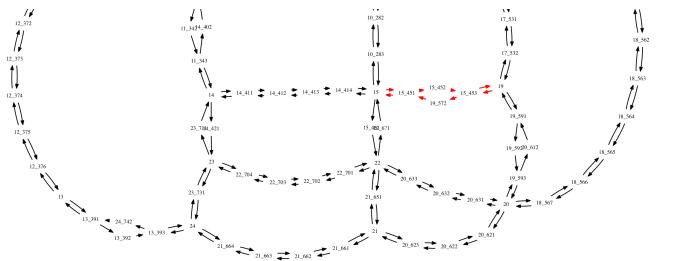
5.2 Analiza wyników

Pierwszym krokiem, który pomaga w analizie otrzymanych wyników jest ich zgrupowanie. Dość intuicyjnym jest grupowanie w zbiory, zamkające podobne obszary (węzły). Na pierwszy rzut oka widać wyraźną zbieżność w jednym rejonie. Niestety, choć zdecydowanie musi mieć on wpływ na poprawę czasu symulacji, nie wydaje się on punktem „strategicznym” komunikacji miejskiej. Mowa oczywiście o zamknięciu węzłów między 18 i 18_542. Przedstawiamy zaznaczony fragment na rysunku 5.1, obróconym o 90° w lewo.



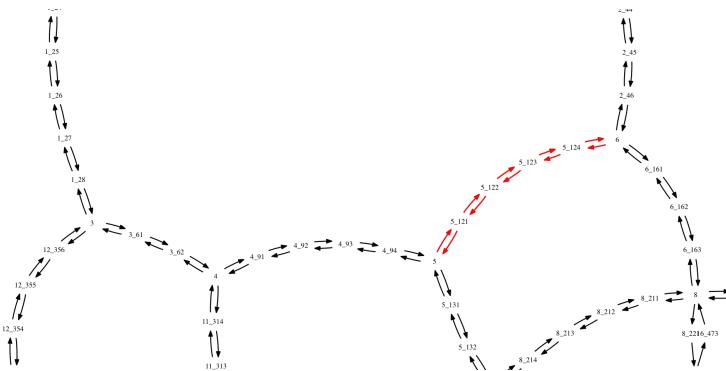
Rys. 5.1: Fragment grafu z zaznaczonym obszarem 1

Ciekawym jest jednak obszar wspólny dla wyników 4.6, 4.8, 4.10, 4.11, 4.12, 4.13, 4.14, 4.15, 4.17. Jest to obszar znajdujący się pomiędzy wspomnianymi wierzchołkami 15 i 19. Przedstawiamy zaznaczony fragment na rysunku 5.2.



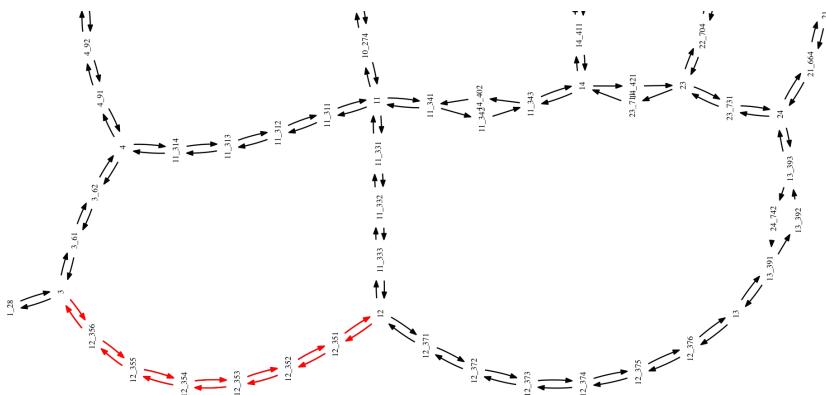
Rys. 5.2: Fragment grafu z zaznaczonym obszarem 2

Drugim obszarem, który został wybrany w przypadku paru rozwiązań, są ulice pomiędzy skrzyżowaniem (wierzchołkiem) 5 i 6. Został on wybrany przez 4.3, 4.5, 4.7, 4.9, 4.13. Przedstawiamy zaznaczony fragment na rysunku 5.3.



Rys. 5.3: Fragment grafu z zaznaczonym obszarem 3

Ostatni obszar, który charakteryzował się wspólnym wynikiem dla więcej niż jednego rozwiązania, tj: 4.4, 4.9, 4.16 znajduje się pomiędzy wierzchołkiem 3 i 12. Przedstawiamy zaznaczony fragment na rysunku 5.4, obróconym o 90° w lewo.



Rys. 5.4: Fragment grafu z zaznaczonym obszarem 4

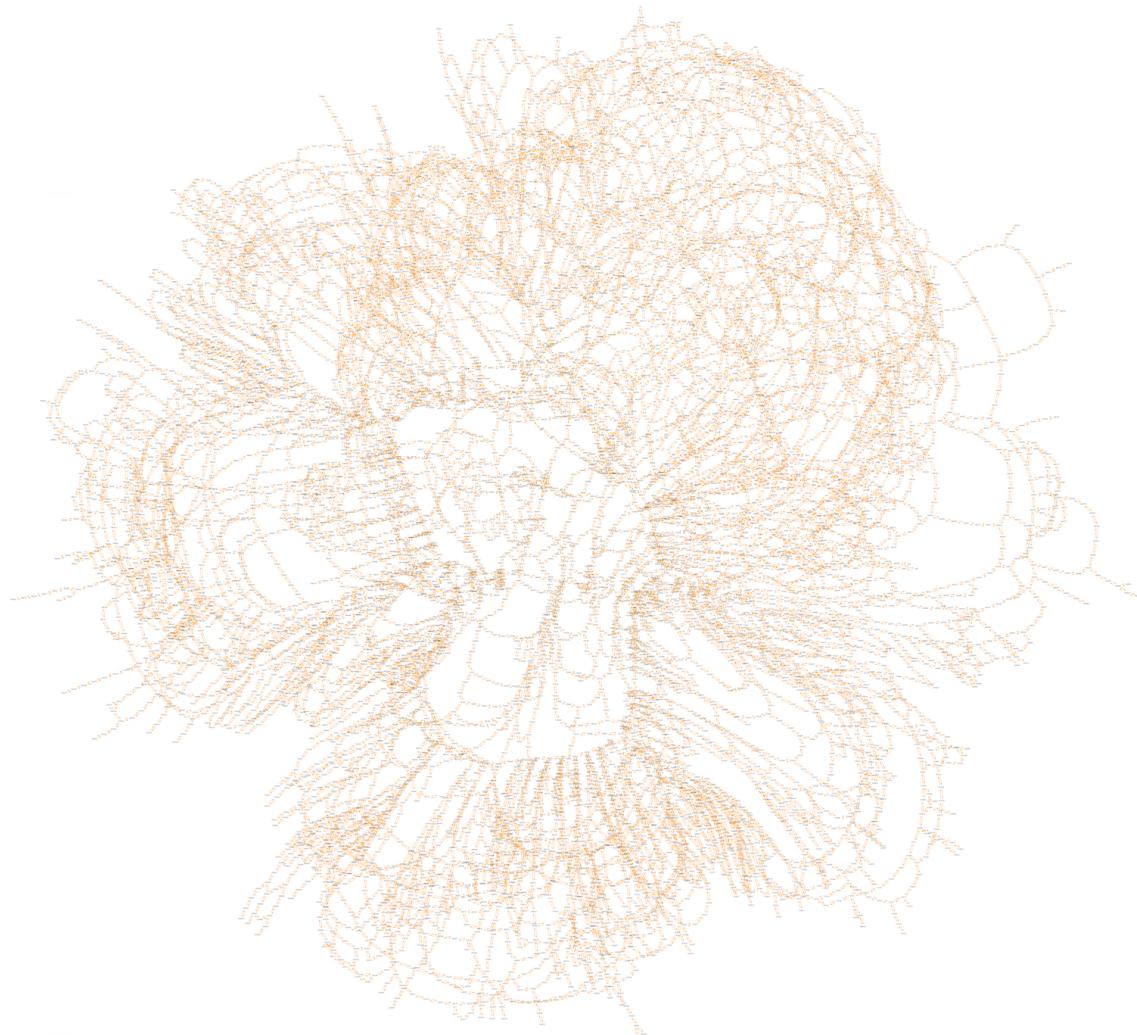
Powyższe sugestie mogą zostać użyte w celu poszerzonych badań nad danymi obszarami, decydując czy faktycznie mają one wpływ na komunikację w mieście. Otrzymane wyniki nie oznaczają również, że nie istnieje bardziej optymalne rozwiązanie. Przestrzeń

przeszukiwań dla powyższego grafu wynosiła 2^{90} , a więc zbadanie wszystkich możliwych rozwiązań problemu jest praktycznie niewykonalne.

5.3 Perspektywy dalszych badań w dziedzinie

Dość oczywistym kierunkiem rozwoju może być oczywiście dalsze przeszukiwanie posiadanego problemu. Jak już zostało wspomniane wyżej, ilość dostępnych rozwiązań jest bardzo duża.

Ciekawszym wydaje się być jednak przeszukiwanie nowych zbiorów. W przypadku symulatora MATSim mamy do swojej dyspozycji dość pokaźną liczbę przykładów opartych o rzeczywiste miasta, na których były prowadzone symulacje. Jeden z dostępnych modeli jest odwzorowaniem miasta Berlin, przy okazji, wykonanym z dużo większą dokładnością niż Sioux Falls 5.5.



Rys. 5.5: Sieć drogowa miasta Berlin w postaci grafu

Ponadto, w pracy można wykorzystać inne twierdzenia i paradoksy dotyczące m.in. transportu miejskiego. Niestety, jak już było wspomniane wcześniej, wiele z tych praw jest opartych o psychologiczne tezy, bez matematycznego podparcia, co powoduje, że wyniki są trudne do przewidzenia.

5.4 Struktura projektu

Załączone na płycie źródła pozwalają na odtworzenie pełnego projektu używając wymienionych wcześniej środowisk programistycznych, Eclipse [17] i PyDev [19]. Poniżej przedstawiona zostaje struktura katalogów, w kolejności alfabetycznej, wraz z krótkim opisem zawartości.

fakematsim/

Jest to sztuczna implementacja symulatora pozwalająca na testowanie działania algorytmu genetycznego w oparciu o losowe wyniki. Działanie opiera się o rozpakowanie gotowego folderu z wcześniej przygotowanymi obliczeniami oraz podmianę wyniku na losową liczbę. Projekt jest przygotowany w oparciu o strukturę projektu *Maven*. Jego skompilowane źródła znajdują się w katalogu *matsim*.

java/

Tutaj znajduje się główny projekt zarządzający aplikacją obliczeniową. Jest on przygotowany w oparciu o strukturę projektu *Maven*. Zawiera przykładowe pliki konfiguracyjne i przygotowane testy *JUnit* pozwalające na ich bezpośrednie wykorzystanie.

literature/

W tym folderze zostały zebrane wszystkie źródła literaturowe wykorzystane przy tworzeniu pracy.

matsim/

Folder ten zawiera skompilowane wersje symulatora MATSim oraz jego falsyfikatu wykorzystywanego podczas testów. Symulator został skompilowany ze źródeł dostępnych na repozytorium głównym projektu¹. Drobne modyfikacje dotyczyły tylko parametrów logowania oraz uruchomienia symulatora. Nie zostały dokonane żadne zmiany ingerujące w przebieg samej simulacji.

¹<https://svn.code.sf.net/p/matsim/source/matsim/trunk>

5.4. STRUKTURA PROJEKTU

sioUX-out/

Jest to domyślny katalog z danymi otrzymanymi w wyniku działania projektu.

paper/

Katalog ze źródłami pracy pisemnej w formacie TeX.

python/

Znajdują się tutaj wszystkie wykorzystane w projekcie skrypty Python wraz z testami jednostkowymi. Jest to również katalog domyślny wywołań skryptów podczas obliczeń.

README.md

Plik zawierający opis instalacji wymaganych przez projekt zależności i bibliotek na podstawie systemu Linux Ubuntu 12.04 LTS.

scenarios/

Zawiera przykładowe scenariusze, które mogą być wykorzystane przy pracy z symulatorem MATSim. Znajduje się tutaj, oprócz wykorzystanego miasta Sioux Falls, Berlin i Bruksela. Ponadto zawiera parę przykładowych plików konfiguracyjnych symulatora.

seminar/

Zawiera prezentację wykorzystaną podczas seminarium dyplomowego, przedstawiającą wstępne założenia projektu.

Spis rysunków

| | | |
|------|---|----|
| 2.1 | Przykładowy graf nieskierowany | 9 |
| 2.2 | Przykładowy graf skierowany | 10 |
| 2.3 | Fragment sieci drogowej miasta Sioux Falls, Południowa Dakota | 10 |
| 2.4 | Sieć drogowa miasta Sioux Falls w postaci grafu | 11 |
| 2.5 | Graf sieci drogowej miasta z dopasowaną geometrią | 11 |
| 2.6 | Wyjściowy układ drogowy | 13 |
| 2.7 | Uzupełniony układ drogowy | 14 |
| 2.8 | Przykładowy graf z zaznaczonymi składowymi silnie spójnymi | 16 |
| 2.9 | Ogólny schemat algorytmu genetycznego | 18 |
| 2.10 | Ogólny schemat operacji krzyżowania | 19 |
| 2.11 | Ogólny schemat operacji mutacji | 19 |
| 2.12 | Fragment sieci w postaci tablicy binarnej | 20 |
| 2.13 | Fragment sieci w postaci grafu | 20 |
| 3.1 | Schemat symulacji | 22 |
| 3.2 | Graf sieci miasta Sioux Falls wykorzystany w badaniach | 23 |
| 3.3 | Rozkład budynków na grafie miasta Sioux Falls | 24 |
| 3.4 | Natężenie ruchu w mieście Sioux Falls w godzinach 6.00-7.00 | 25 |
| 3.5 | Natężenie ruchu w mieście Sioux Falls w godzinach 16.00-17.00 | 25 |
| 3.6 | Graf czasu trwania symulacji MATSim dla stu iteracji | 26 |
| 3.7 | Graf średniego czasu przejazdów dla stu iteracji | 27 |
| 3.8 | Graf wyników agentów dla stu iteracji | 28 |
| 4.1 | Graf wyników algorytmu genetycznego | 34 |
| 4.2 | Sieć miasta Sioux Falls, rozwiązanie nr. 1 | 36 |
| 4.3 | Sieć miasta Sioux Falls, rozwiązanie nr. 2 | 37 |
| 4.4 | Sieć miasta Sioux Falls, rozwiązanie nr. 3 | 37 |
| 4.5 | Sieć miasta Sioux Falls, rozwiązanie nr. 4 | 38 |
| 4.6 | Sieć miasta Sioux Falls, rozwiązanie nr. 5 | 38 |
| 4.7 | Sieć miasta Sioux Falls, rozwiązanie nr. 6 | 39 |

SPIS RYSUNKÓW

| | | |
|------|---|----|
| 4.8 | Sieć miasta Sioux Falls, rozwiązanie nr. 7 | 39 |
| 4.9 | Sieć miasta Sioux Falls, rozwiązanie nr. 8 | 40 |
| 4.10 | Sieć miasta Sioux Falls, rozwiązanie nr. 9 | 40 |
| 4.11 | Sieć miasta Sioux Falls, rozwiązanie nr. 10 | 41 |
| 4.12 | Sieć miasta Sioux Falls, rozwiązanie nr. 11 | 41 |
| 4.13 | Sieć miasta Sioux Falls, rozwiązanie nr. 12 | 42 |
| 4.14 | Sieć miasta Sioux Falls, rozwiązanie nr. 13 | 42 |
| 4.15 | Sieć miasta Sioux Falls, rozwiązanie nr. 14 | 43 |
| 4.16 | Sieć miasta Sioux Falls, rozwiązanie nr. 15 | 43 |
| 4.17 | Sieć miasta Sioux Falls, rozwiązanie nr. 16 | 44 |
| 4.18 | Sieć miasta Sioux Falls, rozwiązanie nr. 17 | 44 |
| 4.19 | Wykres zmiany średniego czasu przejazdu od iteracji dla sieci wynikowych z ujęciem sieci wejściowej | 45 |
| 4.20 | Wykres zmiany średniego czasu przejazdu od iteracji dla sieci wynikowych z ujęciem sieci wejściowej | 46 |
| 4.21 | Wykres zmiany średniego czasu przejazdu od iteracji dla sieci wynikowych z ujęciem sieci wejściowej | 46 |
| 4.22 | Wykres zmiany średniego czasu przejazdu od iteracji dla sieci wynikowych z ujęciem sieci wejściowej | 47 |
| 4.23 | Wykres zmiany średniego czasu przejazdu od iteracji dla sieci najlepszej z ujęciem sieci wejściowej | 47 |
| 5.1 | Fragment grafu z zaznaczonym obszarem 1 | 50 |
| 5.2 | Fragment grafu z zaznaczonym obszarem 2 | 51 |
| 5.3 | Fragment grafu z zaznaczonym obszarem 3 | 51 |
| 5.4 | Fragment grafu z zaznaczonym obszarem 4 | 51 |
| 5.5 | Sieć drogowa miasta Berlin w postaci grafu | 52 |

Spis tabel

| | |
|---|----|
| 4.1 Zbiór najlepszych otrzymanych sieci | 35 |
|---|----|

SPIS TABEL

Spis listingów

| | | |
|-----|---|----|
| 4.1 | Plik konfiguracyjny projektu | 31 |
| 4.2 | Ustawienia algorytmu genetycznego podczas badań | 33 |

SPIS LISTINGÓW

Bibliografia

- [1] Leslie Arthur Keith Bloy, *An investigation into Braess' paradox*, 02/2007
- [2] Rric Pas and Shari Principio *Braess' paradox: Some new insights*, April 1996
- [3] Wataru Nanya, Hiroshi Kitada, Azusa Hara, Yukiko Wakita, Tatsuhiro Tamaki, and Eisuke Kita *Road Network Optimization for Increasing Traffic Flow* Int. Conference on Simulation Technology, JSST 2013.
- [4] Ana L. C. Bazzan and Franziska Klügl *Reducing the Effects of the Braess Paradox with Information Manipulation*
- [5] Dietrich Braess. *Über ein Paradoxon aus der Verkehrsplanung. „Unternehmensforschung“* 12, s. 258–268, 1968 (niem.).
- [6] Mitchell Melanie *An Introduction to Genetic Algorithms* First MIT Press paperback edition, 1998
- [7] M. Rieser, C. Dobler, T. Dubernet, D. Grether, A. Horni, G. Lammel, R. Waraich, M. Zilske, Kay W. Axhausen, Kai Nagel *MATSim User Guide* updated September 12, 2014
- [8] A. Chakirov *Enriched Sioux Falls Scenario with Dynamic Demand* MATSim User Meeting, Zurich/Singapore, June 2013.
- [9] Łukasz Kowalik *Algorytmy i struktury danych, grafy* Wykład, 2003.
- [10] Marta Grzanek *Sztuczna inteligencja, Klasyczny algorytm genetyczny* Wykład, 2007.
- [11] Michalewicz Z., *Algorytmy genetyczne+struktury danych=programy ewolucyjne*, WNT, Warszawa 1999
- [12] Dilvan de Abreu Moreira *Agents: A Distributed Client/Server System for Leaf Cell Generation* Thesis, 1995

BIBLIOGRAFIA

- [13] <http://matsim.org>
- [14] <http://commons.apache.org/proper/commonsmath>
- [15] <http://networkx.github.io>
- [16] <http://www.java.com/pl/>
- [17] <https://eclipse.org>
- [18] <http://pl.python.org>
- [19] <http://pydev.org>
- [20] <http://www.ubuntu.com>
- [21] <http://azure.microsoft.com>
- [22] http://pl.wikipedia.org/wiki/Paradoks_Braessa
- [23] <http://urbnews.pl/paradoksbraessa/>
- [24] http://pl.wikipedia.org/wiki/Paradoks_DownsaThomsona
- [25] http://pl.wikipedia.org/wiki/Prawo_Lewisa_Mogridge'a
- [26] <http://mathworld.wolfram.com/StronglyConnectedDigraph.html>

Abstract

The purpose of the present master thesis was to find an optimal solution of the given road network using genetic algorithm. The optimisation is performed using fixed, set in advance parameters. Paper covers two main fields of the subject. Firstly it introduces the topic of road network optimisation and Braess paradox. Secondly it describes the techniques used to achieve the final goal.

The application performing the optimisation process uses an external ranking system. In this case, a multiagent simulator, MATsim. In paper we describe the aspects of the simulation concering the final results of the project.

The solution is prepared using Java and Python programming languages. During the research, the process was conducted by a machine operating on Linux Ubuntu 12.04 LTS.

The final result proves the validity of the introduced methods. The genetic algorithm, regarding searching of the optimal solution. As well as, the simlation based rank using the multiagent simulator resulted in several solutions of the given network with given parameters.

The discussion regarding the MATSim's iteration results and configuration also explain the differences in long-term evaluation of the simulation. Since the optimal parameters for any simulation are discussable, in the thesis the main goal was to find a simple, fast solution to experienced problem. Therefore, lower average time of travel for agents during longer iterations would be a separate case of study.

Łódź, dnia 25.02.2015 roku

Michał Siatkowski

ul. Tramwajowa 21 m.10
90-132 Łódź

186865

Wydział Fizyki Technicznej
Informatyki i Matematyki
Stosowanej

Informatyka

II stopnia, studia stacjonarne

Oświadczenie

Świadomy/a odpowiedzialności karnej za składanie fałszywych zeznań oświadczam, że przedstawiona praca licencjacka / inżynierska / magisterska¹ na temat

Optymalizacja struktury sieci drogowej
Optimization of structures of road networks

została napisana przeze mnie samodzielnie.

Jednocześnie oświadczam, że ww. praca

- nie narusza praw autorskich w rozumieniu ustawy z dnia 4 lutego 1994 roku o prawie autorskim i praw pokrewnych (j.t. Dz. U. z 2006 r. Nr 90, poz. 631, z późn. zm.) oraz dóbr osobistych chronionych prawem cywilnym, a także nie zawiera danych i informacji, które uzyskałem/am w sposób niedozwolony,
- nie była wcześniej podstawą żadnej innej urzędowej procedury związanej z nadaniem dyplomów wyższej uczelni lub tytułów zawodowych.

.....
(Czytelny podpis Studenta)

¹ Niepotrzebne skreślić

Łódź, dnia 25.02.2015 roku

Michał Siatkowski

ul. Tramwajowa 21 m.10
90-132 Łódź

186865

Wydział Fizyki Technicznej
Informatyki i Matematyki
Stosowanej

Informatyka

II stopnia, studia stacjonarne

Oświadczenie

Świadomy/a odpowiedzialności karnej za składanie fałszywych zeznań oświadczam, że przedstawiona na nośniku CD/DVD praca licencjacka / inżynierska / magisterska¹ na temat

Optymalizacja struktury sieci drogowej
Optimization of structures of road networks

zawiera te same treści, co oceniany przez Opiekuna pracy i Recenzenta wydruk komputerowy.

.....
(Czytelny podpis Studenta)

¹ Niepotrzebne skreślić

