Politechnika Opolska
Wydział Elektrotechniki, Automatyki i Informatyki
Instytut Automatyki i Informatyki

ROZPRAWA DOKTORSKA

Algorytmy sterowania acykliczną sygnalizacją świetlną w zatłoczonej sieci drogowej

mgr inż. Marcin Ruchaj

Promotor:

Prof. dr hab. inż. Krzysztof Latawiec

OPOLE, 2012

Pan Profesor
dr hab. inż. Krzysztof Latawiec

Składam serdeczne podziękowanie za podjęcie trudu kierowania moją pracą
oraz za cenne rady i wskazówki udzielone w toku jej realizacji.

Autor

SPIS TREŚCI

1. SPIS OZNACZEŃ

A sekwencja akcji w metodzie DYPIC $A=\{a_{1,}\ldots,a_N\}$

a_τ w metodzie DYPIC akcja podjęta dla stanu *s* w etapie τ (rozszerzenie lub zakończenie sygnału zielonego) $\tau=1,\ldots,N$

$\bar{b}$ przepustowość ruchu przychodzącego

b przepustowość ruchu wychodzącego

c długość cyklu sygnalizacji [s]

$C_{ss'}$ całkowite opóźnienie skojarzone z przejściem ze stanu *s* w etapie τ do stanu *s'* w etapie τ+1 w metodzie DYPIC

cz sygnał czerwony

d natężenie ruchu [P/h]

dp_{ij} długość pasa *j* skrzyżowania *i* [blok]

dk_τ^j wektor długości kolejek dla każdego pasa do skrzyżowania *i* podczas trwania przedziału τ w metodzie UTOPIA

f współczynnik algorytmu autorskiego, przez który mnożony jest zysk bazowy przypisany do pasa ruchu

F funkcja celu w metodzie TRAVOLUTION

$fw_\tau(s)$ funkcja wartości dla stanu s w etapie τ w metodzie DYPIC

g gęstość ruchu [P/km]

i indeks iteracji

j indeks iteracji

J_κ zyski czasowe w narzędziu sterującym MOVA, gdzie κ=1,2,...,

k indeks iteracji

kier kierunek ruchu na skrzyżowaniu

kol_j kolejka pojazdów na pasie *j*

Kp^j funkcja kosztu dla pasa *j* w metodzie UTOPIA $Kp^j(lp_\tau^j,\nu_\tau^j,\varepsilon_\tau^j)$

Ks^i funkcja kosztu dla skrzyżowania *i* w metodzie UTOPIA $Ks^i(dk_\tau^i,s_\tau^i,\nu_\tau^i,\varepsilon_\tau^i)$

L w metodach SIGSET i SIGCAP całkowita strata czasu w cyklu [s]

lbp_{ij} liczba bloków zajętych przez pojazdy na pasie *j* skrzyżowania *i*

lf_i liczba możliwych faz ruchu skrzyżowania *i*

lg długość chromosomu (liczba genów)

l indeks iteracji

lnu liczby neuronów warstwy ukrytej w algorytmie GenNeural

lpo liczba pojazdów oczekujących w czasie życia grupy osobników w GenNeural

lpp liczba pojazdów poruszających się w ciągu życia grupy osobników w GenNeural

lps liczba pasów wjazdowych danego skrzyżowania (z sygnalizacją), dla którego tworzony jest sterownik w algorytmie GenNeural

lps_i liczba pasów wjazdowych skrzyżowania *i*, dla którego tworzony jest sterownik

$lpss_i$ liczba pasów wjazdowych sąsiadującego skrzyżowania *i* z danym skrzyżowaniem, dla którego tworzony jest sterownik w algorytmie GenNeural

lwe liczba wejść sieci neuronowej w algorytmie GenNeural

m liczba grup sygnalizacyjnych

ms miejsce zajmowane przez pojazd w kolejce przed skrzyżowaniem

n liczba skrzyżowań w sieci lub arterii objętej sterowaniem

N liczba przedziałów całego horyzontu decyzyjnego

nr numer węzła (skrzyżowania), przed którym pojazd się znajduje

o zajętość

$p^a_{ss'}$ prawdopodobieństwo przejścia ze stanu *s* do stanu *s'* po wykonaniu akcji *a*

P prawdopodobieństwo

prz przystosowanie osobnika w algorytmie GenNeural

Q funkcja wartości w metodzie uczenia się ze wzmocnieniem

q_{ij} zysk pasa ruchu *j* skrzyżowania *i*

$r^a_{ss'}$ nagroda przejścia ze stanu *s* do stanu *s'* po wykonaniu akcji *a*

R funkcja nagrody

rb próg losowości

rd() funkcja generująca liczby losowe w zakresie [0,1] wg rozkładu równomiernego

s w metodzie DYPIC stan skrzyżowania, w algorytmach TC-1 Bucket 2.0 i RL SARSA 5 stan pojazdu

S zbiór stanów

s' stan następujący po stanie *s*

Sa_i sygnał *i*-tego skrzyżowania w arterii

semafor losowa liczba rzeczywista w zakresie od 0 do 1 porównywana z progiem losowości *rb* celem podjęcia decyzji przypisywania zysków w sposób losowy lub zgodny z wybranym algorytmem sterowania

Sg kolor sygnału (zielony lub czerwony)

s_τ^i sygnał dla skrzyżowania *i* podczas przedziału τ w metodzie UTOPIA

T długość przedziału czasu

v prędkość pojazdu

V(s) funkcja wartości w algorytmie uczenia się ze wzmocnieniem dla stanu *s*

$V^*(s)$ optymalna funkcja wartości dla stanu *s*

w współczynnik w TRPS zwiększający wrażliwość działania systemu na zajętość

W_{sg} czas oczekiwania przed grupą sygnalizacyjną sg w metodzie TRAVOLUTION

wtt współczynnik algorytmu autorskiego odpowiadający liczbie cykli pojazdów na danym pasie, po przekroczeniu którego zwiększany jest zysk przypisany do danego pasa *f*-krotnie

wyj numer węzła wyjazdowego z sieci, który jest miejscem docelowym podróży

x_k wartość na wejściu *k* sieci neuronowej w algorytmie GenNeural $k = 1, \ldots,$ *lwe*

y_l wyjście neuronu *l* warstwy ukrytej w algorytmie GenNeural $l = 1, \ldots,$ *lnu*

Y_j wyjście *j* sieci neuronowej w algorytmie GenNeural $j = 1, \ldots,$ *lps*

z sygnał zielony

Z_{ik} zbiór pasów ruchu skrzyżowania *i*, które w fazie ruchu *k* mają sygnał zielony

Z_s zbiór skrzyżowań sąsiadujących z danym skrzyżowaniem w GenNeural

α tempo uczenia się $0 < \alpha < 1$

β_{ij} w metodzie SIGSET współczynnik, który przyjmuje wartość równą 1, jeżeli strumień *j* ma pierwszeństwo przejazdu dla grupy sygnal. *i* lub 0, gdy go nie ma

γ współczynnik dyskontowy $\gamma \in [0,1)$

δ_{u_l} wartość progowa wyjścia neuronu *l* warstwy ukrytej w algorytmie GenNeural

ε_τ^j współczynnik odnoszący się do natężenia nasycenia na pasie *j* podczas trwania przedziału τ w metodzie UTOPIA

η_j natężenie nasycenia *j*-tego pasa ruchu

ϑ gen służący ustaleniu czasów trwania faz albo punktów startu przejścia faz w metodzie TRAVOLUTION $\Theta \in [0,1)$

ϑ_j w metodzie SIGCAP margines przepustowości dla pasa ruchu *j*

$\lambda_1, .., \lambda_m$ w metodach SIGSET i SIGCAP splity dla *m* grup

μ współczynnik poddawany maksymalizacji w metodzie SIGCAP $\mu \geq 1$

v_τ^j współczynnik odnoszący się do średniej ogólnej prędkości na pasie *j* podczas trwania przedziału τ w metodzie UTOPIA

ξ_j zgłoszenie dla strumienia *j*

o gen służący ustaleniu lokalnego offsetu w metodzie TRAVOLUTION $o \in [0,1)$

$\pi_{u_{lk}}$ wartość wagi wejścia *k* neuronu *l* warstwy ukrytej w algorytmie GenNeural

ϖ_{sg} waga dla grupy sygnalizacyjnej *sg* w metodzie TRAVOLUTION

σ gen służący ustaleniu kolejności faz ruchu w metodzie TRAVOLUTION $\sigma \in [0,1)$

τ indeks kroku czasowego (etapu)

φ funkcja przejścia neuronu w algorytmie GenNeural

ψ gen służący ustaleniu wspólnego czasu trwania cyklu w metodzie TRAVOLUTION

ω gen służący ustaleniu globalnego offsetu w metodzie TRAVOLUTION $\omega \in [0,1)$

2. NAZEWNICTWO, PARA-DEFINICJE, POJĘCIA, SLOGANY

ALGORYTM STEROWANIA

Uporządkowany zbiór poleceń, który opisuje logikę sterowania ruchem na skrzyżowaniu, tj. zestaw wszystkich możliwych faz ruchu lub głównych grup sygnalizacyjnych, warunki i czas realizacji poszczególnych faz ruchu oraz stan ustalony sygnalizacji.

BLOK

Dyskretna jednostka długości w aplikacji GLD. Pojazd osobowy posiada długość dwóch bloków. 1 [blok] odpowiada długości 2,37 [m].

CYKL SYMULACJI

Jednostka miary czasu w aplikacji GLD. Jest to czas próbkowania, czyli np. czas między kolejnymi odczytami wskazań detektorów lub czas pomiędzy kolejnymi decyzjami sterującymi. W aplikacji założono: 1 [cykl] odpowiada 5 [s].

CYKL SYGNALIZACJI

Czas wyświetlania pełnej sekwencji sygnałów świetlnych, obejmujący minimalny powtarzalny uporządkowany zbiór sygnałów w programie sygnalizacji o określonej strukturze, zapewniający każdemu z uczestników ruchu co najmniej jednokrotne otrzymanie sygnału zielonego.

CZAS WYKONANIA ALGORYTMU

Czas mierzony w [ns] potrzebny do wykonania algorytmu sterowania sygnalizacją.

DETEKTOR

Urządzenie służące do wykrywania przejazdu lub obecności uczestników ruchu za pomocą czujnika.

FAZA RUCHU

Stan ruchu na skrzyżowaniu, w którym przynajmniej jeden z potoków ruchu pojazdów lub pieszych ma dozwolony przejazd albo przejście przez skrzyżowanie.

GRUPA SYGNALIZACYJNA

Zbiór sygnalizatorów wyświetlających te same sygnały w trakcie trwania programu sygnalizacji.

INFRASTRUKTURA SIECI

Zbiór dróg, skrzyżowań oraz urządzeń sterujących i detektorów.

NATĘŻENIE NASYCENIA

Maksymalny możliwy odpływ pojazdów z kolejki na pasie ruchu w czasie sygnału zielonego.

NATĘŻENIE RUCHU

Liczba pojazdów jadących w strumieniu przez poprzeczny przekrój drogi w jednostce czasu.

OFFSET

Odstęp czasowy pomiędzy początkami nadawania sygnałów zielonych na dwóch sąsiednich skrzyżowaniach dla kierunku skoordynowanego, zredukowany do wartości nieprzekraczającej cyklu sygnalizacji.

PLAN SYGNALIZACJI

Harmonogram pracy programów sygnalizacji w zestawie skrzyżowań skoordynowanych wraz z przesunięciami fazowymi.

PROGRAM SYGNALIZACJI

Określony w czasie sposób cyklicznego sterowania ruchem opisany w poszczególnych chwilach sterowania zestawem nadawanych sygnałów. Program podaje sekwencję sygnałów dla uczestników ruchu określoną przez: czas trwania cyklu, strukturę i splity – realizowane przez pojedynczy sterownik.

PRÓG LOSOWOŚCI

Ustalona liczba rzeczywista w zakresie od 0 do 1, która jest porównywana z losową liczbą rzeczywistą. Jeśli wygenerowana liczba losowa jest większa od progu losowości, wówczas zyski przypisywane są według określonego algorytmu, w przeciwnym razie zyski przypisywane są losowo.

PRZEPUSTOWOŚĆ SKRZYŻOWANIA

Liczba pojazdów przejeżdżających przez skrzyżowanie w jednostce czasu [P/h].

SPLIT

Podział czasu cyklu na poszczególne fazy. Wyraża się go zbiorem udziałów czasów trwania poszczególnych faz w czasie trwania cyklu.

STAN RUCHU W SIECI

Informacja ilościowa określająca ruch w sieci w danej chwili za pomocą natężenia ruchu, gęstości ruchu itp.

STAN SKRZYŻOWANIA

Informacja ilościowa określająca skrzyżowanie w danej chwili za pomocą wybranej fazy ruchu, zgłoszeń, liczby pojazdów na pasach wjazdowych itp.

STEROWNIK

Urządzenie realizujące program sygnalizacji.

STEROWANIE

Podjęcie i wykonanie decyzji sterującej o wyborze fazy ruchu na skrzyżowaniu z sygnalizacją świetlną.

STRATEGIE STEROWANIA

Centralny element pętli sterowania, którego zadaniem jest określenie w czasie rzeczywistym wejść sterujących na podstawie pomiarów, estymacji i predykcji parametrów tak, by otrzymać określone cele (np. całkowity spędzony czas) mimo wpływu różnego rodzaju zakłóceń.

SYGNALIZACJA ŚWIETLNA

Zestaw urządzeń służących do sterowania kolizyjnymi potokami ruchu (pojazdów i pieszych) obejmujący: urządzenia sterujące (sterowniki), urządzenia wykonawcze (sygnalizatory wraz z elementami wsporczymi i urządzeniami łączności), urządzenia detekcyjne rejestrujące parametry ruchu (detektory, przyciski), informacyjne (wyświetlacze prędkości) i transmisyjne.

SYGNALIZACJA ACYKLICZNA

Sygnalizacja charakteryzująca się zmienną, zależną od potrzeb ruchu strukturą, ze zmienną sekwencją faz. Jest ona w pełni zależna od ruchu: fazy mogą być tworzone w niej na bieżąco (z pomijaniem fazy włącznie), a czas ich trwania jest zmienny i zależy od określonych charakterystyk ruchu.

SYGNAŁ

Kolor światła sygnalizatora (zielony, żółty, czerwony).

ŚREDNI CZAS OCZEKIWANIA NA SKRZYŻOWANIU

Średni czas oczekiwania wszystkich pojazdów w sieci na skrzyżowaniu mierzony w jednostce czasu [cykl] aplikacji symulacyjnej GLD.

ZAJĘTOŚĆ

Część długości trwania fazy ruchu, w którym detektor jest zajęty przez pojazd. Liczba rzeczywista w przedziale od 0,0 do 1,0.

ZATŁOCZENIE

Warunki ruchu, w których pojazdy mają utrudnienia lub brak możliwości wyprzedzania, zmiany pasa ruchu.

ZATOR

Brak możliwości ruchu w kolumnie pojazdów.

ZGŁOSZENIE

Informacja dotycząca zapotrzebowania na sygnał zielony.

ZŁOŻONOŚĆ CZASOWA ALGORYTMU

Czas wykonania algorytmu podawany w [s] lub [ns].

ZYSK

Liczba rzeczywista przypisana do każdego wjazdowego pasa ruchu z sygnalizacją świetlną, która jest podstawą przy wyborze fazy ruchu w aplikacji symulacyjnej GLD.

3. WSTĘP

Znaczny wzrost liczby samochodów na drogach jest powodem zatłoczenia. Zatłoczenie występuje w 10% sieci drogowej krajów UE-27, a powodowane przez nie koszty szacowane są na 0,9-1,5% unijnego PKB [1,2]. Udział transportu drogowego w emisji gazów cieplarnianych w całym sektorze transportu wynosi 72% [1,2]. Dodatkowo zatłoczenie jest powodem wzmożonego hałasu w miastach.

Głównymi przyczynami występowania zatłoczenia są drogi oraz skrzyżowania o małej przepustowości. W miastach bardzo rzadko istnieje możliwość wybudowania dodatkowego pasa ruchu, dlatego redukcja zatłoczenia sprowadza się do zmiany organizacji ruchu w sieci dróg, a szczególnie do sterowania ruchem na skrzyżowaniach. W małych miasteczkach sterowanie ruchem odbywa się przy pomocy znaków drogowych. W dużych aglomeracjach miejskich same znaki poziome i pionowe są niewystarczające. W związku z tym wprowadzane są sygnalizacje świetlne na skrzyżowaniach. Skrzyżowania są koordynowane celem zmniejszenia liczby zatrzymań. W złożonych sieciach miejskich o dużym natężeniu ruchu stosuje się systemy scentralizowane, które sterują ruchem w całej sieci a nie tylko na pojedynczym skrzyżowaniu. Do sterowania ruchem wykorzystywane są coraz częściej acykliczne sygnalizacje świetlne, których poprawne i efektywne działanie zapewnia zarówno pewna i niezawodna detekcja wszystkich uczestników, jak i prawidłowo przygotowany algorytm sterowania.

W drogowych sieciach miejskich istnieją drogi łączące dwa skrzyżowania o długości kilkudziesięciu metrów, a niekiedy kilkuset metrów, które są zajmowane przez pojazdy na całej swojej długości. Powoduje to powstawanie kolejki na pasach ruchu, z których pojazdy kierują się na tę zatłoczoną drogę. Jak się okazuje jedna taka droga w sieci może mieć bardzo duży wpływ na przepływ pojazdów w innych częściach sieci. Nie wszystkie algorytmy sterowania radzą sobie z tym problemem. Nawet tak złożone obliczeniowo algorytmy jak algorytmy sieci neuronowych, genetyczne i algorytmy uczenia się ze wzmocnieniem mają kłopoty z rozładowaniem zatłoczenia w takich miejscach sieci [3,4].

W sterowaniu ruchem drogowym znaczenie ma również złożoność czasowa algorytmów sterowania. W największych polskich miastach jest kilkaset skrzyżowań z sygnalizacją świetlną, np. Warszawa – ok. 700, Kraków i Wrocław – ok. 180, Poznań – ok. 240 [5]. Natomiast w zachodnich państwach liczba skrzyżowań z sygnalizacją świetlną w mieście przekracza tysiąc. Skrzyżowania łączą ze sobą wielopasmowe drogi. Pociąga to za sobą zapotrzebowanie adaptacyjnych, skoordynowanych systemów sterowania ruchem na znaczną moc obliczeniową centrów sterowania. Rozwiązaniami tego problemu są zdecentralizowane i hierarchiczne struktury systemów [6]. Przykładami takich systemów są OPAC [7], RHODES [8], SCATS [9], SCOOT [10], UTOPIA [11].

Wykorzystanie w algorytmie sterowania sygnalizacją świetlną metody programowania dynamicznego pozwala osiągnąć optymalne sterowanie. Jednak złożoność czasowa tej metody wyklucza ją w wielu przypadkach z zastosowania w systemach czasu rzeczywistego. Przykładem jest tu metoda DYPIC [12]. Twórcy systemów sterowania ruchem drogowym poszukują algorytmów realizujących polecone im zadania w możliwie szybki sposób. Efektem jest np. powstanie metody COP [7].

W systemach sterowania ruchem drogowym znalazły również zastosowanie metody lokalnego przeszukiwania np. Hill-Climbing (BALANCE) [13], które również pozwalają osiągnąć optymalne sterowanie, jednak charakteryzują się dużą złożonością czasową. Innymi metodami wykorzystywanymi do optymalizacji sterowania ruchu drogowego są algorytmy genetyczne (GALOP) [13] i sieci neuronowe (Tahere Royani i in.) [14].

Niniejszą pracę poświęcono problemowi ograniczenia mocy obliczeniowej algorytmów sterowania ruchem drogowym. Najpierw poddano analizie dotychczasowe, wybrane algorytmy sterowania ruchem drogowym, a następnie zaproponowano nowy, autorski algorytm sterowania charakteryzujący się prostotą obliczeniową. Po wprowadzeniu w niniejszym rozdziale w istotę problemu sterowania ruchem drogowym, w Rozdz. 4. przedstawiono motywacje, cele oraz etapy pracy. W kolejnym Rozdz. 5. opisano proces ruchu drogowego oraz podział metod sterowania wraz z charakterystyką wybranych systemów i metod. Modele symulacyjne

wykorzystane w niniejszej pracy, tj. automaty komórkowe oraz systemy multiagentowe, przedstawiono w Rozdz. 6. W oparciu o te modele stworzona została aplikacja symulacyjna GLD (Green Light District), wybrana do porównawczych badań symulacyjnych algorytmów sterowania na poczet tej pracy, opisana w Rozdz. 7. wraz z dokonanymi w niej przez autora pracy modyfikacjami. W aplikacji porównano sześć algorytmów sterowania, których specyfikacje opisano w Rozdz. 8. Rozdział ten przedstawia również etapy powstawania autorskiego algorytmu In-and-Outbound Lane Control. W badaniach symulacyjnych wykorzystano do generowania ruchu w sieciach wyniki rzeczywistych pomiarów ruchu dokonanych w Opolu oraz w Krapkowicach, które zawarto w Rozdz. 9. Wyniki badań symulacyjnych algorytmów sterowania przeprowadzonych dla czterech sieci drogowych w różnych warunkach ruchu oraz podsumowanie badań autorskiego algorytmu sterowania przedstawiono w Rozdz. 10. Podsumowanie całościowe badań, wnioski końcowe oraz kierunki dalszych prac zebrano i przedstawiono w Rozdz. 11. W końcowej części pracy zawarto bibliografię oraz dwa dodatki obejmujące spis rysunków i spis tabel.

Dodatek 1: Spis rysunków

Dodatek 2: Spis tabel

4. TEZA, CEL I ZAKRES PRACY

4.1. Motywacja

W zatłoczonych miastach powszechne stają się acykliczne sygnalizacje świetlne, które dopasowują parametry sterowania takie jak długość sygnału zielonego do aktualnego stanu ruchu pojazdów na skrzyżowaniach. Celem rozładowania zatorów w sieci miejskiej sygnalizacje poszczególnych skrzyżowań są ze sobą koordynowane. W sieci miejskiej istnieje wiele skrzyżowań. Do każdego skrzyżowania doprowadzonych jest kilka lub kilkanaście pasów ruchu. W obrębie skrzyżowania znajdują się również przejścia dla pieszych. W całości tworzy to złożony system wymagający odpowiednich algorytmów sterowania. Dlatego temat niniejszej rozprawy brzmi:

„Algorytmy sterowania acykliczną sygnalizacją świetlną

w zatłoczonej sieci drogowej".

W literaturze opisane są algorytmy o małej złożoności czasowej, np.: Most Cars, który na podstawie liczby pojazdów w kolejce wybiera odpowiednią fazę ruchu, oraz algorytmy o dużej złożoności czasowej, np.: Local Hill-Climbing, który wielokrotnie oblicza globalny zysk dla różnych faz ruchu, by ostatecznie wybrać konfigurację faz dającą największy zysk. Algorytmy o dużej złożoności czasowej dają jednak w niektórych przypadkach kilkukrotnie mniejsze czasy oczekiwania pojazdów przed skrzyżowaniami.

4.2. Pomysł

W niektórych sieciach miejskich w warunkach dużego zatłoczenia nawet złożone czasowo algorytmy uczące się nie potrafią rozładować zatorów na krótkich pasach ruchu. Wprowadza to często zaburzenie do całej sieci. Okazuje się, że algorytmy o małej złożoności czasowej (np.: Most Cars) po wprowadzeniu modyfikacji dają podobne średnie czasy oczekiwania na skrzyżowaniach, a w niektórych sieciach nawet krótsze od algorytmów uczących się, neurogenetycznych itp. Na podstawie przeprowadzonych badań sformułowano następującą tezę:

Istnieje możliwość opracowania prostszych obliczeniowo algorytmów przydzielania priorytetów dla kolejek pojazdów przed skrzyżowaniami w drogowej sieci miejskiej sterowanej acykliczną sygnalizacją świetlną od algorytmów uczących się, neurogenetycznych, które to prostsze algorytmy dają podobne lub mniejsze średnie czasy oczekiwania pojazdów oraz umożliwiają rozładowanie powstających zatorów.

Sformułowano również cel główny:

Stworzenie algorytmu prostszego pod względem złożoności obliczeniowej od algorytmów uczących się, neurogenetycznych, lokalnych metod przeszukiwania, który będzie rozładowywał zatory w drogowej sieci miejskiej i dawał podobne średnie czasy oczekiwania pojazdów przed skrzyżowaniami,

oraz cele pomocnicze:

1. testowanie opracowanego algorytmu w sieciach drogowych o różnej strukturze,
2. optymalizacja opracowanego algorytmu pod względem doboru współczynników do struktury sieci,
3. porównanie złożoności czasowej autorskiego algorytmu z innymi stosowanymi algorytmami (Reinforcement Learning, SARSA, Hill-Climbing itp.).

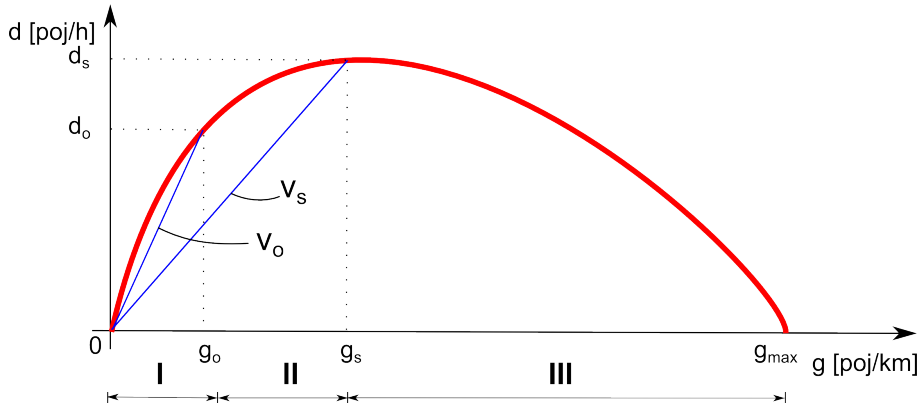
4.3. Najważniejsze Etapy

Przeprowadzone badania zaowocowały powstaniem autorskiego algorytmu, który nazwano In-and-Outbound Lane Control. Algorytm ten testowano w różnych infrastrukturach sieci drogowych. Został porównany pod względem efektywności sterowania (średnie czasy oczekiwania przed skrzyżowaniami, rozładowywanie zatorów w całej sieci drogowej, liczba pojazdów osiągających cel podróży w określonym czasie), jak również złożoności czasowej z algorytmem Most Cars, algorytmem Local Hill-Climbing, algorytmami uczącymi się TC-1 Bucket 2.0 i RL SARSA 5 oraz algorytmem neurogenetycznym GenNeural. W dalszych badaniach testowano algorytm autorski w zamodelowanych fragmentach sieci drogowej Krapkowic oraz Opola, odosobnionego skrzyżowania oraz arterii. Wyniki otrzymane z wykonanych badań zachęciły do przeprowadzenia próby dopasowywania wartości współczynników algorytmu do testowanej struktury sieci, co przedstawiono w niniejszej pracy.

5. STEROWANIE RUCHEM DROGOWYM

5.1. Wprowadzenie

Przemieszczanie ludzi i towarów w sieci ulic miasta jest procesem, który charakteryzuje się znacznym rozproszeniem jak również nieliniowością charakterystyk, niestacjonarnością i niejednorodnością [15,16,17]. Jedną z podstawowych charakterystyk służących opisowi procesu ruchu pojazdów jest zależność natężenia ruchu *d(g)* od jego gęstości *g* [18,19]. Przykładową zależność przedstawiono na Rys. 1.



Rys. 1: Podstawowy wykres zależności natężenia ruchu od gęstości ruchu

Zależność ta charakteryzuje stany ruchu, jakie mogą wystąpić w sieci ulic. Prędkości v_s, v_o, są średnimi prędkościami przestrzennymi, które odpowiadają tangensom nachylenia siecznych przechodzących przez początek układu współrzędnych i punkt na krzywej *d(g)*, względem osi gęstości ruchu. Na wykresie można określić trzy przedziały I – stan swobodny strumienia ruchu (warunki nienasycenia), II – stan częściowo wymuszony (warunki nasycenia) i III – stan wymuszony (warunki przesycenia). W przedziale I średnie prędkości przekraczają wartości dopuszczalne na danym odcinku drogi, więc powinna być utrzymana stała prędkość. W przedziale II wzrostowi natężenia ruchu towarzyszy spadek prędkości. Ważnym punktem charakterystyki jest punkt g_s (gęstość krytyczna). Od tego punktu zaczyna się przedział III, w którym ze wzrostem gęstości następuje zmniejszenie przepustowości i płynności ruchu. W tym obszarze stany ruchu powinny być zabronione poza punktem wartości maksymalnej gęstości d_s. Przedziałem roboczym jest zatem przedział II, gdzie należy poszukiwać kompromisu pomiędzy płynnością ruchu i jego przepustowością. Należy

unikać stanów z pozostałych przedziałów lub w razie wystąpienia możliwie szybko przetransformować w stany z przedziału roboczego. Do utrzymania procesu ruchu w stanach roboczych, szybkiego reagowania na zmiany wynikające nie tylko z jego niestacjonarności i różnorodności, a także wpływania na jego strukturę służą systemy zarządzania ruchem [18,19]. W pracy rozpatrywano sterowanie ruchem w obszarze III.

Po raz pierwszy sygnalizację świetlną zastosowano w sterowaniu ruchem drogowym w 1868 roku w Londynie. Latarnie wyposażone były w lampy gazowe. Elektryczną sygnalizację świetlną zastosowano po raz pierwszy w 1914 roku w Cleveland. Do roku 1918 sygnalizatory były dwukolorowe, tj. światło czerwone i światło zielone. Po raz pierwszy trójkolorowe sygnalizacje wprowadzono w Londynie [20,21].

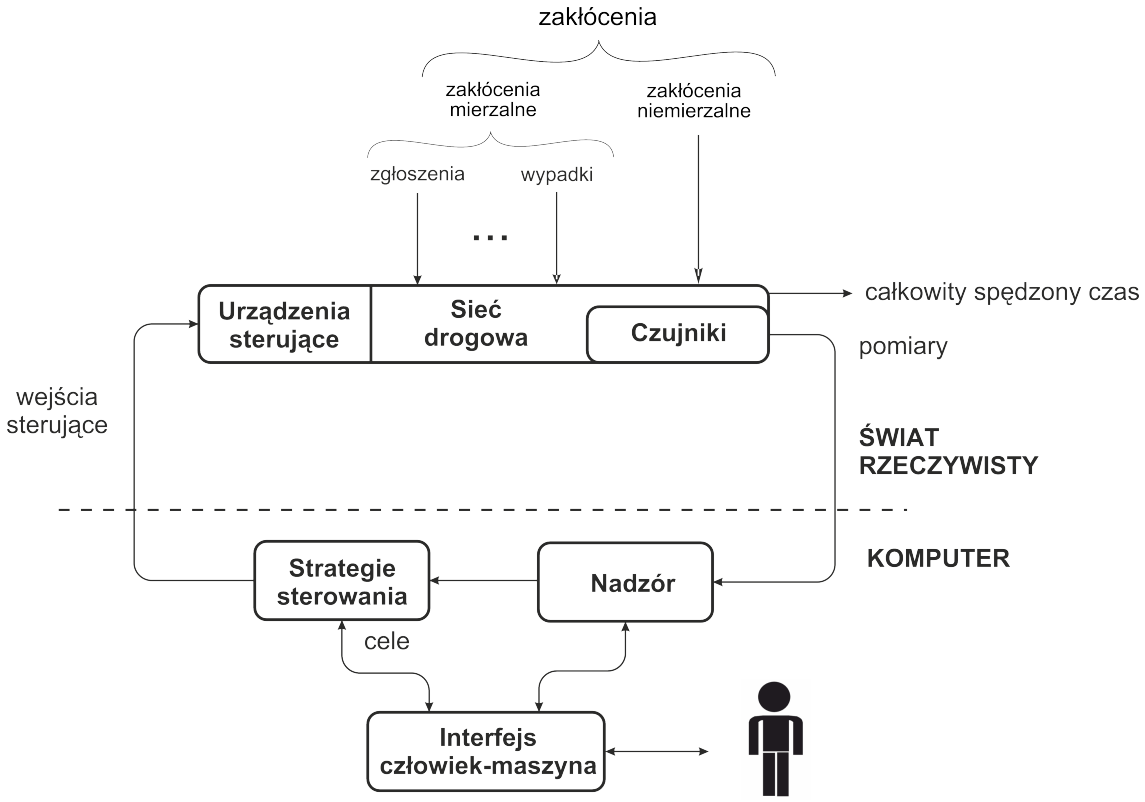
Z punktu widzenia sterowania sygnalizacje świetlne były rozwijane pod różnymi aspektami. Początkowo sterowniki nie posiadały wbudowanego modelu. Obecnie rozwijane są sterowniki pracujące w oparciu o modele. Pierwsze sterowniki były heurystyczne. W chwili obecnej sterowniki zarządzają ruchem wykorzystując optymalizację. Zadaniem sterowników jest nie tylko zarządzanie ruchem w warunkach nienasycenia, ale także w warunkach nasycenia i przesycenia. Obszar sterowania ruchu może obejmować pojedyncze odosobnione skrzyżowanie, arterię lub całą sieć drogową.

Na początku stosowano tylko sygnalizacje stałoczasowe [6,17,22,23]. Sterowniki stałoczasowe, charakteryzują się: wykonywaniem programu o stałej strukturze, ustaloną długością cyklu sygnalizacji, ustaloną długością trwania fazy ruchu, ustaloną długością poszczególnych sygnałów, tj. czerwonego, żółtego i zielonego. Działania sterowania stałoczasowego są ustalane na podstawie historycznych informacji o ruchu. Sterowanie stałoczasowe jest więc rodzajem sterowania w układzie otwartym (bez sprzężenia zwrotnego), które nie adoptuje swoich działań do aktualnych warunków ruchu. Sterowniki stałoczasowe ewaluowały do sterowników zmiennoczasowych [6,17,22,23], działających w układzie zamkniętym z ujemnym sprzężeniem zwrotnym. Sterowniki zmiennoczasowe dzielą się na akomodacyjne (cykliczne) [6,17,22,23], czyli dopasowujące długość trwania faz do warunków ruchu bez zmiany ich sekwencji

i adaptacyjne (acykliczne) [6,17,22,23], charakteryzujące się zmienną sekwencją faz ruchu zależną od ruchu i obliczające parametry, jak np. długość sygnału zielonego.

Sterowanie zmiennoczasowe zalicza się do kategorii sterowania z zamkniętą pętlą (ujemnego) sprzężenia zwrotnego, które dopasowuje działania sterujące do aktualnych warunków ruchu. Strategie sterowania mogą nie być tworzone na podstawie modelu, lecz na podstawie danych historycznych ruchu (dla sterowania stałoczasowego) lub na podstawie aktualnie zmierzonych parametrów ruchu (dla sterowania zmiennoczasowego). Tego typu strategie sterowania rozważają jedynie przeszłe i obecne warunki ruchu, nie biorąc pod uwagę przyszłości. Rozwinięciem tych strategii są strategie oparte na modelu. Pierwsze systemy sterowania ruchem obejmowały jedno skrzyżowanie tworząc zdecentralizowaną strukturę. Lokalne sterowniki korzystają z informacji lokalnych nie komunikując się z innymi skrzyżowaniami. Takie rozwiązanie jest wystarczające w warunkach niskiej gęstości ruchu, kiedy lokalny sterownik na bieżąco reguluje lokalny ruch. Jednakże w przypadku, gdy gęstość ruchu wzrasta lub zgłoszenia ruchu występują nierównomiernie z różnych kierunków, wówczas lokalne informacje są niewystarczające, ponieważ lepsza wydajność lokalnego sterowania nie zawsze oznacza lepszą globalną wydajność sterowania. Oznacza to, że czasami zredukowane lokalne opóźnienia ruchu mogą wywoływać większe opóźnienia i powstawanie zatłoczenia w innych miejscach tej samej sieci. Skłoniło to inżynierów do tworzenia koordynacji skrzyżowań wzdłuż arterii (np. autostrady lub głównej drogi w obszarze miejskim), bądź do koordynacji skrzyżowań na obszarze całej sieci ruchu drogowego [24]. Powstało wiele różnych strategii sterowania skoordynowaną siecią miejską. Stałoczasowe skoordynowane metody sterowania [17,22] podejmują decyzje w oparciu o parametry przepływu ruchu zebrane i zgromadzone w przeszłości. Natomiast zmiennoczasowe [17,22] skoordynowane metody sterowania mogą mierzyć w czasie rzeczywistym stan w sieci i dopasować plan sterowania do tego stanu. Skoordynowane metody sterowania oparte na modelu wprowadzają nie tylko sterowanie z zamkniętą pętlą sprzężenia zwrotnego, by dopasować w czasie rzeczywistym decyzje sterowania do aktualnych warunków, ale dodatkowo przewidują warunki ruchu w przyszłości przy wykorzystaniu modelu predykcyjnego celem podejmowania dobrych decyzji w dłuższym horyzoncie czasowym. Pod względem struktury skoordynowane systemy sterowania dzielą się na scentralizowane,

zdecentralizowane i hierarchiczne [25,26,27,28,29]. Systemy scentralizowane optymalizują całą sieć i szukają globalnego optymalnego rozwiązania dla całej sieci. Systemy zdecentralizowane rozkładają ciężar sterowania na lokalne sterowniki i koordynują je przez wymianę informacji. Natomiast systemy hierarchiczne dzielą całkowity skomplikowany problem sterowania na kilka poziomów, z których każdy rozwiązuje szczególny problem [24].



Rys. 2: Pętla sterowania ruchem drogowym w układzie zamkniętym

Rys. 2 przedstawia podstawowe elementy pętli sterowania w układzie zamkniętym (z ujemnym sprzężeniem zwrotnym). Przepływ ruchu w sieci drogowej zależy od kilku czynników, które są podzielone na trzy grupy: *wejścia sterujące*, *zakłócenia* i *wyjście sieci*. Wejścia sterujące są bezpośrednio powiązane z odpowiednimi urządzeniami sterującymi (wykonawczymi) takimi jak sygnalizacja świetlna lub znaki zmiennej treści (tablice elektroniczne) itp. Z*akłócenia mierzalne* są zakłóceniami, na których wartość nie można wpływać, ale można je mierzyć (np. zgłoszenia, czyli wystąpienie ekstra zapotrzebowania na sygnał zielony) lub takie, które można wykryć (np. wypadki i inne zdarzenia losowe) lub można przewidzieć w pewnym przyszłym horyzoncie czasowym. *Zakłócenia niemierzalne* to zakłócenia, których nie można mierzyć lub przewidzieć (w rozsądnych kosztach), np. opady deszczu, śniegu,

zlodowacenia nawierzchni, pojawienie się zwierząt na drodze itp. *Wyjście sieci* lub wydajność jest mierzona przez odpowiednie wskaźniki takie jak *całkowity spędzony czas* przez wszystkie pojazdy w sieci przez pewien okres czasu. Zadaniem *Nadzoru* jest zwiększenie i rozszerzenie informacji dostarczanych przez odpowiednie czujniki (np. pętle indukcyjne) wymagane przez następujące strategie sterowania i ludzi nadzorujących. Centralnym elementem pętli sterowania jest blok *Strategie sterowania*, których zadaniem jest określenie w czasie rzeczywistym wejść sterujących na podstawie pomiarów, estymacji i predykcji parametrów tak, by otrzymać określone *cele* (np. całkowity spędzony czas) mimo wpływu różnego rodzaju zakłóceń. Stosowność i efektywność strategii sterowania w dużej mierze determinuje wydajność całego systemu sterowania.

5.2. Acykliczna Sygnalizacja Świetlna

Sygnalizacje acykliczne [17] mogą być sterowane według jednej z dwóch zasad: sterowanie fazami lub sterowanie grupami. Pierwsze podejście polega na tym, że sterownik ruchu rejestrujący zgłoszenia zapotrzebowania na sygnał zielony ma możliwość doboru długości sygnału zielonego i operowania zadeklarowanymi fazami ruchu w dowolnej, zmiennej w czasie i zależnej od potrzeb kolejności. Podejście to powoduje równouprawnienie w uzyskaniu pierwszeństwa dla poszczególnych użytkowników ruchu drogowego. Natomiast sterowanie grupami polega na tym, że parametry sterowania określa się dla poszczególnych grup sygnalizacyjnych, a nie dla zadanych faz ruchu. Podejście to umożliwia uwzględnienie preferencji dla wybranych użytkowników np. priorytetową obsługę pojazdów komunikacji zbiorowej. Działanie sygnalizacji zmierza do dynamicznego tworzenia faz oraz maksymalizacji dozwolonych nakładek sygnałów zielonych dla poszczególnych grup sygnalizacyjnych. Sterowniki są w stanie kierować ruchem na skrzyżowaniu bez konieczności założenia długości trwania poszczególnych faz ruchu. Sygnał zielony dla każdej grupy sygnalizatorów występuje w ściśle określonym czasie i o ustalonej na podstawie sytuacji ruchowej długości czasu jej trwania. Cechą charakterystyczną sterowania acyklicznego jest stan ustalony sygnalizacji, który polega na ciągłym nadawaniu na każdym sygnalizatorze ustalonego sygnału stałego lub przerywanego. Występuje on w przypadku braku zgłoszeń od

użytkowników ruchu i trwa do chwili zarejestrowania zgłoszenia od dowolnego użytkownika dojeżdżającego do skrzyżowania, po którym sygnalizacja przechodzi do stanu wzbudzonego – realizującego obsługę zgłoszenia. Stan ustalony może przebiegać według następujących strategi sterowania:

– wszystkie czerwone,

– obsługa wybranej fazy ruchu,

– obsługa ostatniej wybranej fazy ruchu (użyta w badaniach symulacyjnych w tej pracy),

– praca cykliczna.

Poprawne i efektywne działanie sygnalizacji acyklicznej zapewnia zarówno pewna i niezawodna detekcja wszystkich uczestników ruchu, jak i prawidłowo przygotowany algorytm sterowania, rozumiany jako uporządkowany zbiór poleceń, który opisuje logikę sterowania ruchem na skrzyżowaniu. Uwzględniając uwarunkowania wynikające z geometrii skrzyżowania, organizacji ruchu i przyjętych grup sygnalizacyjnych, przygotowuje się zestaw wszystkich możliwych faz ruchu lub głównych grup sygnalizacyjnych, jako podstawowe elementy sterowania. Należy również opracować warunki i czas realizacji poszczególnych faz ruchu oraz wybrać stan ustalony sygnalizacji. Stanowią one elementy opisowe logiki sterowania na skrzyżowaniu z sygnalizacją świetlną.

5.3. Miary Efektywności Ruchu Na Skrzyżowaniu Z Sygnalizacją

Efektywność funkcjonowania skrzyżowania, w tym jakość warunków ruchu są oceniane przy użyciu miar efektywności. Przy określaniu miar efektywności często występującą jednostką jest pojazd ekwiwalenty (umowny) [E]. Jest to pojazd odpowiadający parametrami geometrycznymi pojazdowi osobowemu. Inne pojazdy przeliczane są według odpowiednich współczynników (np. motocykl 0,3 [E], autobus i samochód ciężarowy 2,0 [E]). Miary efektywności można podzielić na trzy grupy [17,30]:

- związane z przepustowością,
- związane z tworzeniem się kolejek,
- związane z oddziaływaniem na środowisko.

Do miar efektywności związanych z tworzeniem się kolejek zalicza się:

- średnie przeciętne straty czasu przypadające na jeden pojazd [s/E], [s/P],
- średnie czasy zatrzymania [s/E],
- poziom swobody ruchu PSR (miernik jakościowy),
- liczba zatrzymań [z],
- kolejka pojazdów na początku sygnału zielonego (na końcu sygnału czerwonego) [P],
- kolejka na końcu sygnału zielonego (kolejka pozostająca),
- średnia długość kolejki [P],
- maksymalna długość kolejki [P],
- maksymalna kolejka w okresie przeciążenia [P],
- średnie straty czasu pieszych [s/PS],
- straty czasu przypadające na osobę [s/O],
- łączne straty czasu i wynikające z zatrzymań [E/h].

Wymienione miary mogą być obliczane: dla pasów ruchu, wlotu oraz całego skrzyżowania.

Straty czasu jest to dodatkowy czas potrzebny na przejechanie skrzyżowania z sygnalizacją świetlną w porównaniu z czasem przejazdu przez skrzyżowanie bez zakłóceń (bez zatrzymania na wlocie) [17].

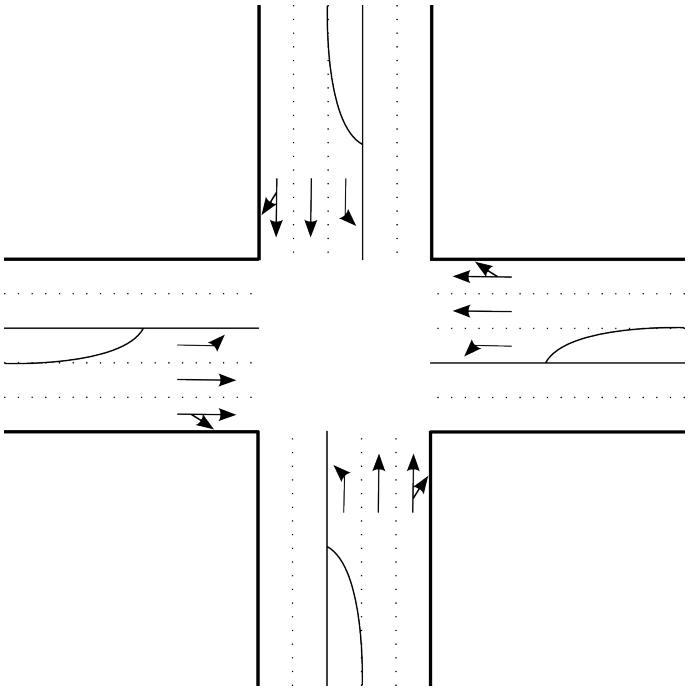
Liczba zatrzymań jako przypadająca średnio na pojazd lub łączna wszystkich pojazdów, z uwzględnieniem wielokrotnych zatrzymań pojazdów, a udział pojazdów zatrzymanych w ogólnej liczbie pojazdów przejeżdżających skrzyżowanie dobrze charakteryzują płynność ruchu [17].

Długość kolejki wyrażona jest liczbą pojazdów i najczęściej jej wielkość rejestrowana jest na początku (kolejka maksymalna) i końcu (kolejka pozostająca) sygnału zielonego. Jest istotna przy wymiarowaniu stref akumulacji dodatkowych pasów ruchu oraz przy projektowaniu koordynacji sygnalizacji. Kolejki pozostające (świadczące o przeciążeniu wlotu) są ważną miarą w ocenie jakości sterowania [17].

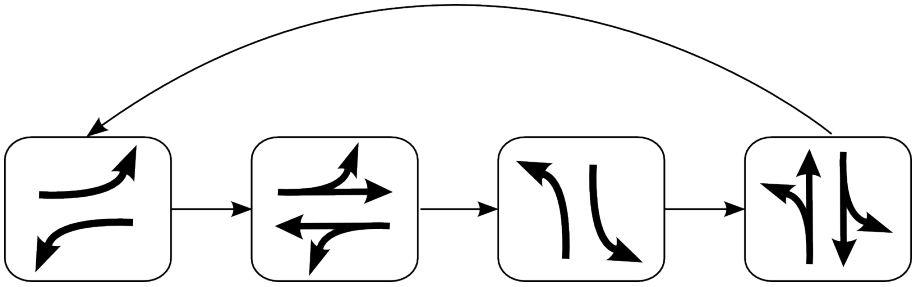
5.4. Sterowanie Stałoczasowe

5.4.1. Odosobnione skrzyżowanie

Rys. 3 przedstawia typowe odosobnione skrzyżowanie z ośmioma pasami wlotowymi. Pas zewnętrzny jest jednocześnie pasem jazdy na wprost, jak i pasem jazdy w prawo. Tego typu skrzyżowanie charakteryzuje się tym, że posiada cztery bezkolizyjne fazy skrętu w lewo (Rys. 4). Sterowanie stałoczasowe [17,22,31] przebiega w sposób cykliczny. W każdym cyklu jest kilka faz ruchu. Dla każdej fazy ruchu jest dozwolony jeden lub więcej bezkonfliktowych strumieni ruchu. W przypadku sterowania stałoczasowego sekwencja faz ruchu, jak i opóźnienia faz są niezmienne. Również długość cyklu dla każdego potoku jest stała.



Rys. 3: Przykładowe skrzyżowanie o czterech wlotach dwupasmowych



Rys. 4: Fazy ruchu dla skrzyżowania o czterech wlotach

Strategie stałoczasowe dla skrzyżowań odosobnionych stosowane są w warunkach ruchu swobodnego. Rozróżnia się dwa rodzaje strategii stałoczasowych: oparte na optymalizacji grup sygnalizacyjnych i oparte na optymalizacji faz ruchu. Strategie oparte na optymalizacji grup sygnalizacyjnych określają optymalne splity i długość cyklu tak, by zminimalizować całkowite opóźnienie lub zmaksymalizować przepustowość skrzyżowania. Natomiast strategie oparte na optymalizacji faz określają nie tylko optymalne splity i czas trwania cyklu, ale także optymalną konfigurację faz ruchu, które ma duże znaczenie dla skomplikowanych skrzyżowań. Popularnymi przykładami strategii opartych na optymalizacji grup są SIGSET [32] i SIGCAP [33]. Zakładając, że jest *m* grup, SIGSET i SIGCAP precyzują splity $\lambda_1,..,\lambda_m$ i długość cyklu *c*, przy czym z definicji [22]

$$\lambda_0+\lambda_1+...+\lambda_m=1 \tag{1}$$

gdzie $\lambda_0=\frac{L}{c}$ i *L* jest to całkowita strata czasu w cyklu. Celem uniknięcia tworzenia się kolejek pojazdów, dla każdego strumienia *j* musi być spełniony warunek [22]

$$\eta_j=\sum_{i=1}^{m} \theta_{ij}\lambda_i \geqslant \xi_j \quad \forall j \tag{2}$$

gdzie η_j – natężenie nasycenia i ξ_j – zgłoszenie dla *j*-tego pasa ruchu, θ_{ij} przyjmuje wartość równą 1 jeżeli strumień *j* ma pierwszeństwo przejazdu w grupie sygnalizacyjnej *i* oraz wartość równą 0 w przeciwnym razie. Nierówność (2) wymaga, żeby zgłoszenie ξ_j. *j*-tego pasa ruchu nie było większe od maksymalnego możliwego natężenie nasycenia tego strumienia. Ostatecznie pod uwagę brane są ograniczenia maksymalnej długości cyklu i minimalnego czasu trwania sygnału zielonego.

Nieliniowa funkcja opóźnienia uzyskana przez Webstera [34] dla warunków ruchu swobodnego została użyta jako funkcja celu w strategii SIGSET. Tak więc, SIGSET rozwiązuje problem programowania nieliniowego z ograniczeniami liniowymi celem zminimalizowania całkowitego opóźnienia na skrzyżowaniu dla danych zgłoszeń ξ_j. Z drugiej strony SIGCAP może zostać użyty do maksymalizacji przepustowości skrzyżowania w następujący sposób. Załóżmy, że rzeczywiste zgłoszenie nie jest ξ_j jak w [35] ale $\mu \cdot \xi_j$ z $\mu \geq 1$. SIGCAP w miejsce ξ_j w [35] wstawia $\mu \cdot \xi_j$ maksymalizuje μ przy

tych samych ograniczeniach jak SIGSET, co prowadzi do problemu programowania liniowego [22].

Warto zauważyć, że z przyczyn wymienionych wcześniej, maksymalizacja przepustowości zawsze prowadzi do maksymalizacji dopuszczalnej długości cyklu. W związku z tym, SIGCAP powinien być użyty dla skrzyżowań o wysokiej zmienności zgłoszeń w celu zapobieżenia przesyceniu, podczas gdy SIGSET może być użyty poniżej dopuszczalnego marginesu przepustowości przez podstawienie w miejsce ξ_j [35] przez $\vartheta_j \xi_j$, gdzie $\vartheta_j < 1$ jest wcześniej określonym parametrem marginesu [22].

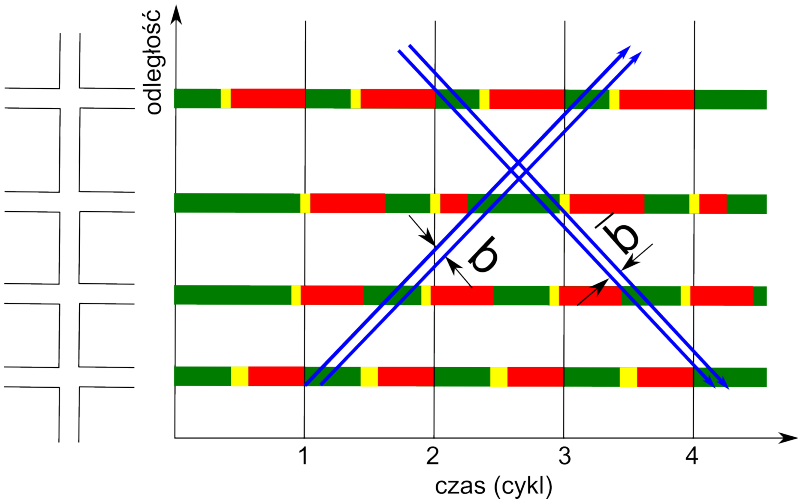
Metody oparte na optymalizacji faz rozwiązują podobny problem, rozszerzony odpowiednio celem rozważenia różnych kombinacji faz ruchu [36]. Metody takie rozpatrują relacje zgodności powiązanych strumieni jako wcześniej określone oraz dostarcza optymalnych konfiguracji faz ruchu, splitów i długości cyklu, celem minimalizacji całkowitego opóźnienia lub maksymalizacji przepustowości skrzyżowania. Powstałe zadanie optymalizacji jest typu mieszanego zadania programowania całkowitoliczbowego, dla którego dokładne rozwiązanie daje metoda podziału i ograniczeń (branch-and-bound) [37,38,39]. Czas obliczeń tej metody jest tu bardzo długi. Jednak dla obliczeń off-line nie ma to większego znaczenia [22].

5.4.2. Skoordynowana arteria

Najbardziej popularnymi reprezentantami tej klasy strategii są MAXBAND [35,40,41] i TRANSYT [42,43]. Ze względu na swą naturę strategie stałoczasowe są stosowane tylko w warunkach nienasyconego ruchu.

5.4.2.1. MAXBAND

Pierwszą wersję MAXBAND [35,40,41] stworzył John Little. MAXBAND rozpatruje dwukierunkową arterię z n sygnałami (skrzyżowaniami) $Sa_1,...,Sa_n$, dla której określa odpowiadające offsety (przesunięcia początków sygnałów zielonych w stosunku do początku cyklu) tak, by zmaksymalizować liczbę pojazdów mogących przebyć daną drogę w pewnym określonym przedziale prędkości bez zatrzymywania się przed skrzyżowaniami (zielona fala) [22].



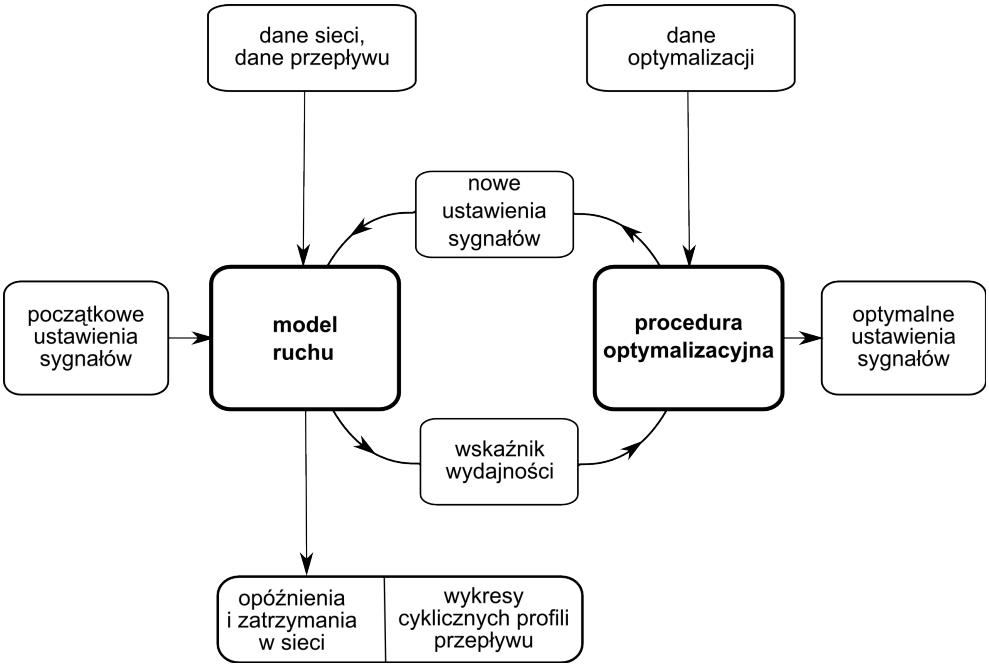
Rys. 5: Koordynacja sygnalizacji w arterii

Splity w MAXBAND są dane (odpowiednio do zgłoszeń bocznej ulicy). Problem jest jednak w umiejscowieniu czasu trwania światła czerwonego (czerwone linie poziome każdego sygnału Sa_i na Rys. 5) sygnałów arterii tak, by zmaksymalizować odpowiednio przepustowość ruchu przychodzącego $\overline{b}$ i wychodzącego b. Dla sformułowania odpowiedniego zadania konieczne jest wprowadzenie kilku binarnych zmiennych decyzyjnych, co prowadzi do mieszanego, całkowitoliczbowego zadania programowania liniowego. Wykorzystywana do znalezienia optymalnego rozwiązania metoda podziału i ograniczeń (branch-and-bound) pozwala zredukować wymagany nakład obliczeniowy. Próby dalszych redukcji nakładów obliczeniowych wymaganych przez tę metodę przedstawiono w [41]. John Little w [44] rozszerzył podstawową metodę MAXBAND przez wprowadzenie kilku ograniczeń czasu trwania cyklu, przez co metodę można zastosować do sieci arterii [22].

Metodę tę wzbogacono o wiele znaczących rozszerzeń celem rozpatrzenia różnorodności nowych aspektów takich jak: czas rozładowania istniejącej kolejki, potoki skrętu w lewo i różne przepustowości dla każdego odcinka arterii (MULTIBAND) [22,45].

5.4.2.2. TRANSYT

Pierwotnie TRANSYT [22,42,43] został stworzony przez D. I. Robertsona. W późniejszym czasie został rozszerzony i usprawniony [46]. Jest to najbardziej znana i najczęściej stosowana strategia sterowania ruchem drogowym. Często stosowana jest jako metoda odniesienia do sprawdzania poprawności włączona w strategiach czasu rzeczywistego. Pierwszy obszar implementacji metody TRANSYT – stworzono plany sterowania, które wskazywały na oszczędność czasu podróży przez sieć o ok. 16%.



Rys. 6: Struktura metody TRANSYT

Rys. 6 przedstawia podstawową strukturę metody: TRANSYT jest inicjowany ustawieniami początkowymi sygnałów włącznie z wstępnie określonym grupowaniem, minimalnym czasem trwania światła zielonego dla każdej grupy i dla każdego skrzyżowania, a także początkowy wybór splitów, offsetów i długości cyklu. Unikatowa długość cyklu *c* lub *c/2* jest rozpatrywany dla wszystkich skrzyżowań w sieci celem umożliwienia koordynacji offsetów. Dane sieci jak również dane przepływu ruchu zawierają w sobie geometrię sieci, przepływ nasycenia, czas przejazdu odcinkiem między kolejnymi skrzyżowaniami, stały i znany współczynnik skrętów dla każdego skrzyżowania oraz stałe i znane zgłoszenia. Model ruchu zawiera węzły (skrzyżowania) oraz odcinki (drogi łączące sąsiadujące skrzyżowania). Koncepcja rozproszenia strumienia („platoon dispersion” [42]) jest zastosowana do zamodelowania postępu przepływu wzdłuż odcinka drogi. Warunki przesycenia nie mogą być opisane, chociaż pewne ulepszenia zostały osiągnięte w tym aspekcie w ostatniej rozszerzonej wersji programu. Metoda przebiega w sposób iteracyjny: dla danych wartości zmiennych decyzyjnych (wejścia sterujące), tj. splitów, offsetów i czasów trwania cyklu model dynamiczny sieci oblicza odpowiedni indeks wydajności np. całkowitą liczbę zatrzymań pojazdów. Heurystyczny algorytm optymalizacji „hill-climb” wprowadza małe zmiany do zmiennych decyzyjnych i nakazuje uruchomienie nowego modelu i tak dalej, aż zostanie znalezione lokalne minimum [17,22].

5.5. Sterowanie Zmiennoczasowe

5.5.1. Odosobnione skrzyżowanie

Strategie zmiennoczasowe [17,22,31] wykorzystują pomiary zrealizowane w czasie rzeczywistym przez detektory pętli indukcyjnych, które umieszczane są w jezdni. Jedną z najprostszych strategii tej klasy jest metoda odstępów między pojazdami, która jest stosowana dla skrzyżowań dwufazowych. Minimalny czas trwania światła zielonego jest przypisany do obu faz ruchu. Jeżeli podczas minimalnego czasu zielonego żaden pojazd nie przejedzie przez odpowiednie detektory, wówczas strategia przechodzi do drugiego etapu. Jeżeli natomiast zostanie zarejestrowany jakiś pojazd, tworzony jest przedział krytyczny (CI – critical interval), podczas którego wszelki zarejestrowany pojazd powoduje wydłużenie czasu światła zielonego, które pozwoli pojazdowi na przejechanie przez skrzyżowanie. Jeżeli żaden pojazd nie zostanie zarejestrowany podczas CI, strategia przechodzi do następnego etapu, w przeciwnym razie tworzony jest nowy CI i tak dalej, aż zostanie osiągnięta wartość maksymalna światła zielonego. Wersja rozszerzona tej metody bierze pod uwagę zapotrzebowanie ruchu na podejście antagonistyczne, celem podjęcia decyzji, kiedy powinno nastąpić przejście do następnej fazy, a kiedy przejścia nie powinno być.

Bardziej wyszukana wersja tego rodzaju strategii została zaproponowana przez Alana Millera [47] i jest zaimplementowana w narzędziu sterującym MOVA [48]. Strategia Millera odpowiada co T sekund (np. $T=2s$) na pytanie: *Czy przełączenie do następnej fazy powinno nastąpić teraz czy należy je przesunąć o czas T?* Celem udzielenia odpowiedzi na to pytanie, strategia oblicza (przy pewnych uproszczonych założeniach) zyski i straty czasowe powstałe przy wszystkich podejściach, jeżeli podjęcie decyzji zostaje przesunięte przez $\kappa \cdot T$ sekund. Odpowiednie zyski czasowe J_κ, $\kappa=1,2,...,$ są łączone w jedno kryterium $J=max\{J_\kappa, \kappa=1,2,...\}$. Jeżeli $J<0$ przełączenie następuje natychmiastowo, w przeciwnym razie decyzja jest przesuwana do następnego kroku czasowego. Ocenę porównawczą tych prostych algorytmów przedstawiono w [49].

5.5.2. Skoordynowane strategie zmiennoczasowe

Wybór planu (TRPS – Traffic responsive plan selection [50]) następuje w urządzeniu master lub w centralnym systemie komputerowym. Po wybraniu przez

urządzenie centralnego sterowania planu sterowania, przesyłana jest informacja do wszystkich sterowników w grupie skoordynowanej, celem poinstruowania o jednoczesnym przejściu do nowego planu. W ten sposób zostanie zachowana koordynacja. Centralne urządzenie sterujące monitoruje dane natężenia i/lub zajętości z wielu detektorów. Dane z detektorów są ważone, dzielone oraz przetwarzane celem obliczenia wartości dla kilku kluczowych parametrów, które są porównywane do założonych progów. Kiedy próg jest przekroczony, na podstawie aktualnych warunków wybierany jest nowy plan. Różni dostawcy systemów sterowania rozpatrują różne parametry ruchu oraz wykorzystują różne algorytmy sterowania. Większość algorytmów obejmuje oddzielnie obliczenia i porównania do progów dla każdej długości cyklu (całkowite natężenie ruchu), offsetu (kierunek strumienia ruchu o największym natężeniu) i splitu (względna zajętość i/lub natężenie na różnych drogach). Niezależnie od użytego algorytmu wybór planu sterowania wymaga od użytkownika wprowadzenia potencjalnie skomplikowanych parametrów konfiguracyjnych. Podczas konfigurowania TRPS może być potrzebny znaczny wkład użytkownika w identyfikację detektorów pojazdów, który zapewni odpowiednią reprezentację warunków ruchu, celem ustalenia odpowiednich wartości parametrów związanych z tymi detektorami, jak również ustalenia odpowiednich progów i skojarzonych z nimi planów, oraz dostrojenia konfiguracji na podstawie jego działania, które zostało zrealizowane przez TRPS. Dane historyczne natężenia ruchu (a najlepiej zajętości) powinny być dostępne dla kandydujących detektorów zanim rozpocznie się wybór detektora i proces konfigurowania. Często konieczne jest wielokrotne dopasowywanie parametrów, w szczególności progów, w oparciu o obserwacje obliczonych wartości w stosunku do aktualnych warunków ruchu, dopóki zostanie ustalona efektywna konfiguracja. Detektory stosowane w TRPS najczęściej umiejscawiane są z dala od linii zatrzymania. Stosowane są detektory na wjazdach do skrzyżowania lub na wyjazdach ze skrzyżowania. Detektory powinny być tak skonfigurowane, by kontroler mógł wygenerować w miarę bezbłędne dane natężenia i zajętości oddzielnie dla każdego kierunku ruchu, nawet jeśli obsługiwane są przez ten sam sygnał, a najlepiej dla każdego pasa ruchu. Takie detektory są często określane mianem detektorów systemowych. Detektory stosowane w TRPS muszą być aktywnie monitorowane, niezawodne. Usterki muszą być szybko naprawiane [50].

W przypadku, gdy czas potrzebny na przejechanie pojazdów stojących przed skrzyżowaniem przekracza długość czasu trwania światła zielonego, kolejka pojazdów może wydłużać się przez wiele cykli. W tym przypadku pomiar natężenia ruchu obsługiwanego przez daną fazę ruchu nie stwierdzi warunków przeciążenia. Zajętość zarejestrowana przez detektor wjazdowy w obszarze kolejkowania może być wykorzystany do wykrycia wystąpienia dodatkowych kolejek nawet wtedy, gdy wielkość ruchu nie ulega zmianie. Zajętość może być wykorzystana do wyboru planu sterowania z odpowiednią długością cyklu i splitu, który dopasuje się do zgłoszenia nadmiernego ruchu. Z drugiej strony jednak, zajętość jest względnie niezależna od zmian wielkości ruchu w warunkach swobodnego przepływu, kiedy kolejki nie wydłużają się.

Algorytmy zazwyczaj zezwalają na użycie kombinacji natężenia ruchu i zajętości przy wyborze planu. Jedną z technik jest ich połączenie w jedną bezwymiarową wartość przez zsumowanie natężenia ruchu *d* i wielokrotności zajętości *o*. Ten proces określany jest często przez wyrażenie $d + w \cdot o$ [50].

Zwiększenie wartości współczynnika *w* zwiększa wrażliwość na zmiany zajętości w stosunku do natężenia ruchu.

Algorytmy stosowane w TRPS zawierają histerezę, by zapobiec oscylacjom wokół wartości progowej. Ważne jest również, by plany czasowe nie były zmieniane zbyt często. Konsekwencją tego jest nieefektywność ze względu na przejścia offsetu. Z drugiej strony obliczenia wartości potrzebnych do podjęcia decyzji wyboru nie mogą trwać zbyt długo, by system nie zmieniał zbyt wolno planów w porównaniu do nagłych zmian natężenia ruchu, które mogą być efektem nagłego wypadku. Niemniej jednak dobrze skonfigurowany TRPS może poprawić sterowanie ruchem w porównaniu z systemami zmieniającymi plany w zależności od pory dnia, szczególnie dotyczy to skoordynowanych arterii, które są narażone na nieprzewidywalne zmiany przepływu ruchu. Podobnym problemem jest zmiana planu w warunkach szczytowego ruchu, kiedy offset ma większy negatywny wpływ [50].

Jeżeli aktualne warunki ruchu są zupełnie inne niż te, dla których jeden z dostępnych planów został zaprojektowany, może być konieczne opracowanie planów dla nietypowych warunków. Przykładowo specjalny plan może służyć dodatkowym dużym natężeniom ruchu (w jednym kierunku) pozostawiając pewien obszar

wydarzeniom sportowym lub obejściu blokady autostrady. TRPS może automatycznie wprowadzić taki plan, gdy występują takie warunki [50].

TRPS tylko wybiera plan do pracy, ale nie dokonuje zmian parametrów planów [50]. Zmiana parametrów należy do adaptacyjnych systemów sterowania.

5.5.3. UTCS – Urban Traffic Control System

Począwszy od lat siedemdziesiątych Departament Transportu Stanów Zjednoczonych (USDOT – U.S. Department of Transportation) przeprowadził kilka projektów badawczych na systemach sterowania ruchem w miastach (UCTS – Urban Traffic Control System [7]). Strategie sterowania skrzyżowaniami zaproponowane i ocenione w tych projektach można w dużej mierze podzielić na cztery kategorie: sterowanie pierwszej generacji, sterowanie drugiej generacji, sterowanie trzeciej generacji i sterowanie czwartej generacji [17]. Strategie sterowania pierwszej generacji tworzą plany sterowania ruchem w oparciu o historyczne dane uśrednionego natężenia ruchu. W zależności od pory dnia wybierane są i wprowadzane do działania odpowiednie wcześniej przygotowane plany sterowania. Plany zwykle są zmieniane co 15 minut. Strategie drugiej generacji optymalizują plany sterowania co 5 minut w oparciu o przewidywane dane natężenia ruchu zamiast danych historycznych. Zmiana planów sterowania nie może następować w dłuższych odstępach czasu niż 10 minut, aby uniknąć zakłóceń spowodowanych przejściami z jednego planu do drugiego. Strategie sterowania trzeciej generacji są podobne do strategii drugiej generacji z tą różnicą, że przedział czasu uaktualniania planu jest zawarty między 3 minuty a 5 minut. Dodatkową różnicą generacji trzeciej jest obliczanie prócz długości splitu i offsetów również długości cyklu. Zarówno metody drugiej jak i trzeciej generacji tworzą plany on-line [51,52]. W sterowaniu czwartej generacji uaktualnianie parametrów następuje z cyklu na cykl, obliczany jest moment zmiany fazy. Cykl jest zmienny w czasie.

5.5.4. SCOOT – Split, Cycle and Offset Optimization Technique

P. B. Hunt i inni [53] opracowali system SCOOT, który zalicza się do metod czwartej generacji [17]. W metodzie SCOOT skrzyżowania pogrupowane są w wiele podobszarów. Sterowniki w każdym podobszarze pracują z tą samą długością cyklu. SCOOT dokonuje częstych i małych zmian parametrów sterowania sygnału takich jak

długość sygnału, czas trwania fazy ruchu i offsetu planu offline w oparciu o aktualne zmiany przepływu ruchu [54,55]. Dopasowanie parametrów sterowania sygnałem jest oparte na modelu ruchu, który przewiduje opóźnienia i zatrzymania powstałe w wyniku działania różnych planów sterowania. Plan, który najlepiej redukuje opóźnienia i zatrzymania jest wybierany i wprowadzany do działania [53].

SCOOT posiada trzy procedury optymalizacyjne, za pomocą których dopasowuje czasy sygnałów: optymalizator splitów, optymalizator offsetów i optymalizator długości cyklu [56]. Stąd pochodzi nazwa systemu Split Cycle and Offset Optimization Technique. Każdy optymalizator ocenia wpływ małej przyrostowej zmiany czasu sygnału na ogólną wydajność w sieci danego regionu. Wykorzystywany jest wskaźnik wydajności, oparty na prognozach opóźnień pojazdów i zatrzymań na każdym odcinku między skrzyżowaniami.

Optymalizator splitów działa przy każdej zmianie etapu przez analizę obecnego czasu światła czerwonego i zielonego, by określić, czy czas zmiany etapu powinien być przesunięty do przodu, do tyły czy ma pozostać niezmieniony. Optymalizator splitów pracuje krokowo w odstępach co 1 do 4 sekund. Moduł ten działa analizując obecną sytuację na każdym skrzyżowaniu przy pomocy cyklicznych profili przepływu prognozowanych dla każdego odcinka z poprzedniego lub następnego skrzyżowania. Następnie dokonuje oceny w czterosekundowych krokach, czy istniejący czas działania powinien być rozszerzony, skrócony czy powinien pozostać niezmieniony [56].

Optymalizator cyklu działa na poziomie regionu raz na pięć minut lub co dwie i pół minuty, gdy czas cyklu gwałtownie wzrasta. Określa „skrzyżowanie krytyczne" w danym regionie i stara się dopasować czas cyklu tak, by zachować to skrzyżowanie z dziewięćdziesięcioprocentowym nasyceniem odcinków na każdym etapie. Jeśli stwierdzi potrzebę zmiany czasu cyklu, może ją zwiększyć lub zmniejszyć w cztero-, ośmio- lub szesnastosekundowych krokach [56].

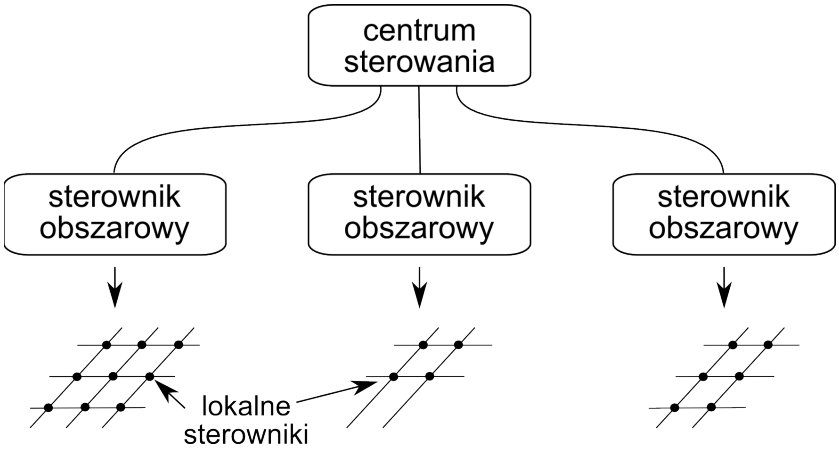
Przez połączenie stosunkowo niewielkich zmian taktowania sygnału ruchu, SCOOT może odpowiedzieć na krótkoterminowe, lokalne, szczytowe zgłoszenia ruchu, jak również na następujące tendencje w czasie i zachować stałą koordynację sieci.

Optymalizator offsetu pracuje raz w ciągu jednego cyklu dla każdego skrzyżowania.

SCOOT jest szeroko stosowany w Wielkiej Brytanii. Jest wiele implementacji tego systemu w różnych krajach. Ostatnią wersją SCOOT jest SCOOT MC3 [10], który ma kilka nowych własności takich jak zdolność przeskoczenia fazy ruchu celem udzielenia priorytetu autobusom.

5.5.5. SCATS – Sydney Coordinated Adaptive Traffic System

SCATS [9,57,58] został opracowany przez australijskich naukowców. Jest to metoda podobna do SCOOT, zaliczana do adaptacyjnych metod sterowania i klasyfikowana do trzeciej generacji [17]. Zasadniczą różnicą między SCATS a SCOOT jest to, że SCATS nie posiada modelu ruchu ani optymalizatora planów sterowania. SCATS wybiera najlepszy czas trwania fazy ruchu i offsetu z kilku predefiniowanych planów [54] w oparciu o rzeczywiste warunki przepływu ruchu.



Rys. 7: Struktura SCATS

SCATS posiada hierarchiczną strukturę systemu, która posiada trzy poziomy jak pokazano na Rys. 7. Najniższy poziom zawiera lokalne sterowniki umieszczone przy każdym skrzyżowaniu. Ich zadaniem jest zbieranie danych, wstępne przetwarzanie danych i ocenę awarii detektorów. Poziom środkowy obejmuje regionalne nadrzędne urządzenia sterujące, które są podstawą SCATS. Każdy regionalny nadrzędny sterownik zarządza pracą do kilkuset lokalnych sterowników. Natomiast lokalne sterowniki są pogrupowane w systemy lub podsystemy. Podsystem zazwyczaj zawiera kilka skrzyżowań i jest najmniejszym elementem sterującym na tym poziomie. Najwyższy poziom stanowi centrum sterowania, który w rzeczywistości nie wykonuje operacji sterujących. Zadaniem centrum sterowania jest głównie monitorowanie całego systemu [51].

SCATS dopasowuje długość cyklu na skrzyżowaniu w oparciu o parametr podobny do stopnia nasycenia, który odpowiada maksymalnemu możliwemu odpływowi pojazdów z kolejki na pasie ruchu w czasie sygnału zielonego [17]. Wykorzystuje stosunek efektywnie wykorzystanego czasu światła zielonego do całkowitego dostępnego czasu światła zielonego dla każdej fazy ruchu. Do pomiaru tego stosunku służą detektory zatrzymania. W oparciu o zmierzony stopień nasycenia, długość cyklu podsystemu dla następnego cyklu zmienia co najwyżej o ±6 sekund. Ponadto dostępne są cztery zaprogramowane wartości długości cyklu: minimalna, średnia, maksymalna i długość cyklu, w którym bardzo nasycona faza (stretch phase – faza rozciągnięta) otrzymuje dodatkowy czas [58].

System SCATS posiada w sumie cztery dostępne plany splitów. Plany określają normalną sekwencję faz ruchu, które mogą być zmieniane między planami a opcją przeniesienia niewykorzystanego czasu z jednej fazy ruchu do drugiej. Jedna z faz ruchu w każdym planie jest wybrana jako faza rozciągnięta, która przydziela więcej czasu światła zielonego dla bardziej nasyconej fazy. Wraz z długością cyklu, split jest zmieniany przy użyciu algorytmu „głosowania planów splitów". Plan ten uwzględnia fazę z najwyższym stopniem nasycenia. Wybierany jest plan, na który zostały oddane dwa głosy w trzech kolejnych cyklach. Połączenie zmian długości cyklu i planu splitu ma na celu wyrównanie stopnia nasycenia na wszystkich strategicznych wlotach [58].

Celem obliczenia planów offsetu, SCATS wykorzystuje czas licznika długości cyklu od zera do zakończenia wybranej fazy ruchu. W sąsiadujących podsystemach wybierane jest skrzyżowanie krytyczne i czas od zera do zakończenia wybranej fazy ruchu jest używany jako punkt odniesienia do obliczenia zewnętrznych offsetów. Na zasadzie systemu głosowania jest wybierany odpowiedni plan offsetowy z pięciu dostępnych planów. System ten wybiera plan offsetowy, jeśli otrzymuje on cztery z pięciu kolejnych głosów za jego przyjęciem [58].

Dodatkowo, podsystemy są koordynowane przy użyciu systemu głosowania odcinków (dróg łączących skrzyżowania). Licznik odcinka jest zachowany, który koordynuje sąsiednie podsystemy, jeśli osiąga liczbę cztery i przerywa koordynację, jeśli jego wartość wynosi zero. Przewidziano również załączenie koordynacji między

podsystemami, jeśli przepływ mierzony przez detektory należące do warstwy drugiej sterowania (obszarowego) przekracza zadaną wartość [58].

System SCATS wdrożono w 2005 roku w Rzeszowie [59]. Obecnie system jest rozszerzany. Kolejne skrzyżowania są dołączane do systemu. Oprócz Rzeszowa kolejnymi miastami, w których zastosowano SCATS w Polsce są: Łódź i Olsztyn [5,60,61].

5.5.6. DYPIC – Dynamic Programmed Intersection Control

D. I. Robertson i R. D. Bretherton [12] opracowali metodę optymalnego sterowania zwaną DYPIC opartą na programowaniu dynamicznym dla skrzyżowania odosobnionego. Autorzy przedstawili działanie metody na prostym skrzyżowaniu z dwoma kolizyjnymi potokami. Ze względu na tylko dwa kolizyjne potoki, decyzjami sterującymi były rozszerzenie lub zakończenie aktualnego sygnału zielonego. W swoich rozważaniach autorzy założyli, że informacja o ruchu jest znana w kilku najbliższych minutach (w horyzoncie decyzyjnym). Jednakże jest to niemożliwe w rzeczywistych aplikacjach. Z tego powodu metoda DYPIC była używana głównie do teoretycznych rozważań oraz do porównań z innymi praktycznymi metodami sterowania.

W metodzie DYPIC cały horyzont decyzyjny jest podzielony na *N* przedziałów. Każdy przedział ma długość pięciu sekund. Na końcu każdego przedziału (punkt decyzyjny) następuje podjęcie decyzji dotyczącej przedłużenia aktualnej fazy zielonej lub zakończenia jej i przydzielenia sygnału zielonego dla innego potoku. W założeniach nie było ograniczeń minimalnego ani maksymalnego czasu trwania światła zielonego. D. I. Robertson i R. D. Bretherton sformułowali to sterowanie skrzyżowaniem jako problem programowania dynamicznego. W szczególności punkty decyzyjne odpowiadały koncepcji etapów w programowaniu dynamicznym; stany każdego etapu były określone przez sygnał (zielony lub czerwony) i kolejki na każdym dojeździe. W założeniach przyjęto, że informacje o dokładnym ruchu dopływającym do skrzyżowania jest znana dla całego horyzontu decyzyjnego oraz długości kolejek na każdym dojeździe w każdym etapie mogą być oszacowane przy użyciu modeli ruchu. Celem optymalizacji jest znalezienie optymalnej strategii sterowania zawierającej sekwencję akcji $A=\{a_1,..,a_N\}$, która minimalizuje całkowite opóźnienie. W oparciu o początkowe sygnały, długości kolejek na dojazdach i informacjach o przyszłych

napływach ruchu, całkowity proces podjęcia decyzji może zostać zilustrowany jako drzewo decyzyjne pokazane na Rys. 8 [51].

W metodzie DYPIC szukanie optymalnej strategii sterowania dokonuje się przy pomocy funkcji opisanej następującym wzorem:

$$fw_{\tau}(s)=\min_{a_{\tau}}\left\{C_{ss'}+fw_{\tau+1}(s')\right\},\ \tau=1,\dots,N,\ s\in S_{\tau},\ s'\in S_{\tau+1} \qquad (3)$$

gdzie:

S_τ – wszystkie możliwe stany w etapie τ,

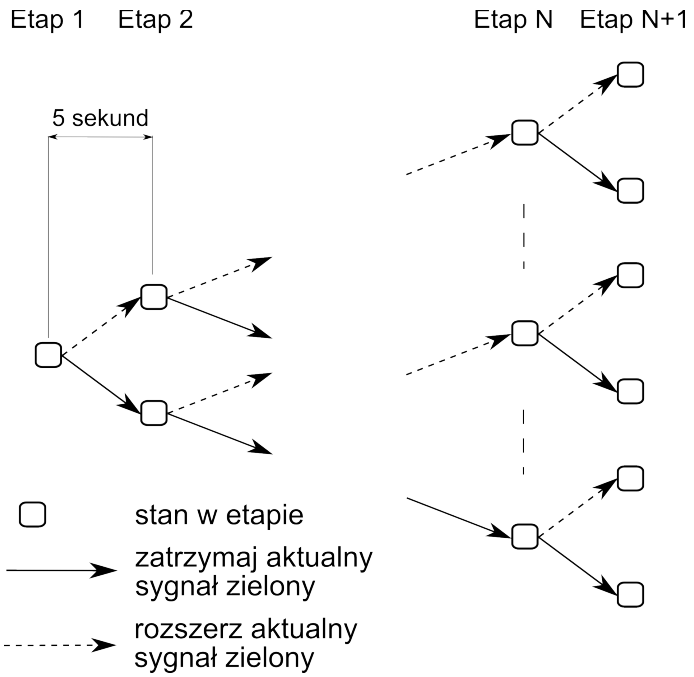
$S_{\tau+1}$ – wszystkie możliwe stany w etapie $\tau+1$,

$C_{ss'}$ – całkowite opóźnienie skojarzone z przejściem ze stanu s w etapie τ do stanu s' w etapie $\tau+1$,

$fw_\tau(s)$ – funkcja wartości dla stanu s w etapie τ,

a_τ – akcja podjęta dla stanu s w etapie τ (rozszerzenie lub zakończenie),

N – liczba etapów minus 1.



Rys. 8: Drzewo decyzyjne wykorzystywane w metodzie DYPIC

Wartości każdego stanu na etapach od 1 do N można uzyskać przy pomocy równania (3). Algorytm pracuje od etapu N do etapu 1. Wartość dla stanu początkowego w rzeczywistości jest minimalnym opóźnieniem wynikającym z optymalnej strategii sterowania. Postępując ścieżką prowadzącą do wartości stanu początkowego, można znaleźć najlepszą strategię sterowania. Ta metoda jest często

określana programowaniem dynamicznym z metodą przeglądu wstecznego. Jest ona oparta na zasadzie optymalności Bellmana [62], która mówi, że bez względu na poprzednie decyzje, pozostałe akcje muszą być optymalne ze względu na aktualne stany [51].

Istnieją cztery główne problemy metody DYPIC. Po pierwsze każda akcja może spowodować przejście do jednego z dwóch stanów w następnych etapie. Jeśli istnieje *N+1* etapów, maksymalna możliwa liczba stanów w etapie końcowym jest 2^N. Jeżeli horyzont czasu wynosi jedną minutę a długość przedziału czasu zgodnie z założeniem 5 sekund, to maksymalna możliwa liczba stanów może wynosić $2^{60/5}$=4096. Natomiast w przypadku, gdy horyzont czasu wynosi 2 minuty a długość przedziału czasu 5 sekund, wtedy maksymalna możliwa liczba stanów w etapie końcowym osiągnie wartość $2^{120/5}$=16777216. Chociaż programowanie dynamiczne teoretycznie może dać globalnie optymalne rozwiązanie problemu, to czas potrzebny na uzyskanie rozwiązanie może być poważnym problemem dla sterowania ruchem w czasie rzeczywistym. Po drugie w tym prostym przykładzie było rozważane tylko odosobnione skrzyżowanie z dwoma potokami. W praktycznych zastosowaniach jest zazwyczaj osiem potoków i na każdym etapie może być wiele różnych akcji. Po trzecie przykład zakłada znane dopływy ruchu w całym horyzoncie decyzyjnym. To jest niemożliwe w rzeczywistości. W końcu, DYPIC zakłada deterministyczne przejście stanów. Nowy stan określony jest na podstawie danych aktualnych długości kolejek, sygnały, informacji o dopływach ruchu i akcjach, które mogą być wykonane. Takie założenie może w rzeczywistości być nieprawdziwe, ponieważ zachowanie kierowców jest bardzo skomplikowane i żaden model nie jest w stanie perfekcyjnie przewidzieć przyszłych stanów ruchu [51].

Warto zauważyć, iż w metodzie sterowania DYPIC, czas trwania fazy ruchu jak i sekwencja faz nie jest stała. Również długość cyklu nie jest stała. Wyróżnia to tę metodę w porównaniu do stałoczasowych metod sterowania, skoordynowanego zmiennoczasowego sterowania, SCOOT i SCATS. D. I. Robertson i R. D. Bretherton [12] dokonali porównania metody DYPIC ze sterowaniem stałoczasowym na skrzyżowaniu odosobnionym. Przetestowano metodę w dwóch rodzajach warunków ruchu, którymi był przypływ pojazdów losowy i cykliczny. Dla warunków losowego napływu ruchu metoda zredukowała opóźnienie o co najmniej 50 procent. W warunkach cyklicznego

napływu ruchu metoda DYPIC zredukowała średnie opóźnienie o 3 sekundy na pojazd [51].

5.5.7. OPAC – Optimized Policies for Adaptive Control

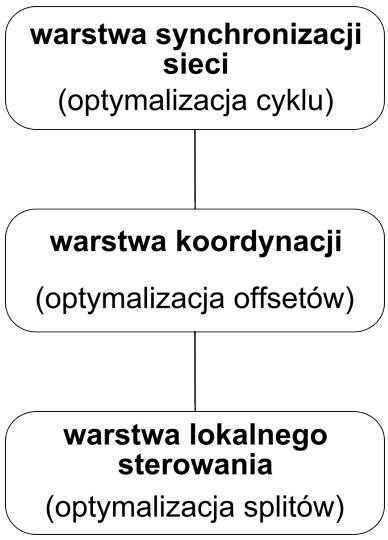
Strategie drugiej i trzeciej generacji miały lepsze wyniki niż strategia pierwszej generacji. Wydaje się, że lepiej reagują na zmieniające się warunki ruchu wykorzystując przy tym dane z detektorów, jak również dane przewidywalne w przyszłości. W pewnych jednak warunkach okazuje się jednak, że strategia pierwszej generacji przewyższa pozostałe [52,63,64]. Ze względu na niezadowalające wyniki strategii drugiej i trzeciej generacji N. Gartner [52] zaproponował strategię sterowania rzeczywiście reagującą na zgłoszenia, która nie jest ograniczona do klasycznej koncepcji długości cyklu i czasu trwania fazy [7]. W swoim opracowaniu N. Gartner zaprezentował przykład sterowania ruchem na odosobnionym skrzyżowaniu przy użyciu metody programowania dynamicznego, nazwany później OPAC-1 [51,58,65], który był podobny do metody DYPIC [12]. stwierdził jednak, że pomimo możliwości uzyskania globalnej optymalności, to jednak metoda programowania dynamicznego nie nadaje się do zastosowań w sterowaniu w czasie rzeczywistym ze względu na długi czas obliczeń jak również wymaganie dokładnych danych dotyczących napływu ruchu. Bazując na OPAC-1, N. Gartner zaproponował prostszy algorytm sterowania wykorzystując algorytm optymalnego, sekwencyjnego wyszukiwania z ograniczeniami (OSCS – Optimal Sequential Constrained Search) w miejsce programowania dynamicznego [7]. Ta nowa metoda została nazwana OPAC-2. Algorytm OSCS wymaga mniej czasu na obliczenia daje wyniki bliskie optymalnym uzyskanym przy pomocy metody programowania dynamicznego. Jednakże algorytm OSCS jest bardziej skomplikowany niż metoda programowania dynamicznego. Algorytm OSCS składa się z trzech następujących kroków [7,51]:

Krok 1: Cały horyzont czasu składa się z kilku etapów. Każdy etap ma długość od 50 do 100 sekund.

Krok 2: W każdym etapie sygnał musi być raz zmieniony i mogą być dokonane co najwyżej trzy zmiany sygnałów. Może być wiele różnych scenariuszów zmiany sygnału na każdym etapie. Dla każdego scenariusza obliczane jest uzyskane opóźnienie.

Krok 3: Dla każdego etapu stosowany jest algorytm OSCS. Optymalny scenariusz zmiany sygnałów jest określany niezależnie dla każdego etapu. Wejścia etapu pośredniego zawierają kolejki wszystkich wjazdów na końcu poprzedniego etapu, aktualny sygnał oraz ostatnią zmianę sygnału. Opóźnienia dla każdego możliwego scenariusza zmian sygnałów są oszacowywane i porównywane. Wybierany jest scenariusz, któremu odpowiada najmniejsza wartość opóźnienia i zachowywana jest jako optymalne rozwiązanie dla aktualnego etapu. Uzyskane kolejki, sygnały oraz informacje o ostatniej zmianie sygnału są przekazywane do dalszych obliczeń w następnym etapie.

W tym samym artykule [7] N. Gartner zaproponował metodę ze zmiennym horyzontem celem predykcji napływu ruchu. W ten sposób zostało usunięte ograniczenie związane z potrzebą znajomości dokładnego napływu ruchu. Dodanie metody predykcji zmiennego horyzontu rozwinęło OPAC-2 do OPAC-3 [51,58,65]. N. Gartner przeprowadził analizę OPAC-3 przy użyciu specjalnej wersji symulatora



Rys. 9: Struktura hierarchiczna metody OPAC

NETSIM. Analiza została przeprowadzona na rzeczywistych danych zgromadzonych na skrzyżowaniu w Tucson w Arizonie. Wyniki analizy wykazały, że OPAC w porównaniu z istniejącą metodą sterowania zredukował średnie opóźnienie o 30-50% oraz zwiększył średnią prędkość o 10-20% [51].

W kolejnym studium, N. Gartner podsumował rozwój metody OPAC oraz wyniki zastosowania wersji OPAC-4 [51,58,65]. OPAC-4 ma zastosowanie już nie tylko

dla pojedynczego skrzyżowania, ale również dla arterii i całej sieci. OPAC-4 wykorzystuje technikę wirtualnego stałego cyklu (VFC – Virtual-Fixed-Cycle). Stąd zwany jest często VFC-OPAC. VFC-OPAC ma hierarchiczną strukturę z trzema poziomami jak przedstawiono na Rys. 9. Warstwa synchronizacji oblicza co kilka minut VFC. W oparciu o VFC warstwa koordynacji optymalizuje offset na każdym skrzyżowaniu. Warstwa lokalnego sterowania optymalizuje zmiany sygnału w odniesieniu do VFC i ograniczeń offsetu z warstw synchronizacji i koordynacji.

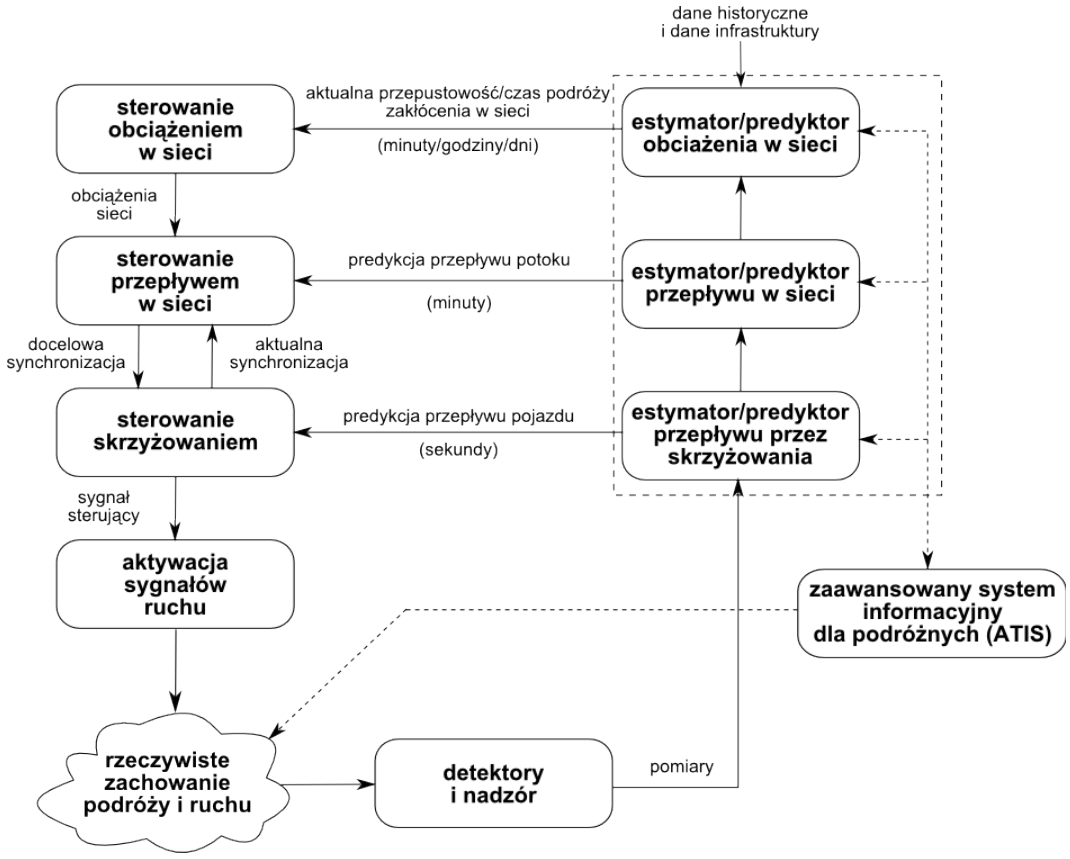
OPAC-4 został później przetestowany na arterii w Reston w Virginii. Test został przeprowadzony w dwóch etapach. W pierwszym etapie istniejący system koordynacji stałoczasowej został przestrojony i zgromadzono dane wydajnościowe. W drugim etapie zastosowano OPAC-4 i również zebrano dane wydajnościowe. Porównanie wykazało, że wydajność istniejącego systemu i wydajność OPAC-4 nie różnią się od siebie znacząco. N. Gartner w artykule [65] stwierdził, że taki rezultat testu mógł być spowodowany różnymi przepływami ruchu w obu okresach zbierania danych [51].

Obecnie rozwijana jest piąta wersja metody OPAC [58]. Celem jest stworzenie proaktywnego sterowania. OPAC ma być zintegrowany z DTA (Dynamic Traffic Assignment). DTA stanowią modele i algorytmy predykcyjne. Stwierdzono bowiem, że sterowanie w czasie rzeczywistym jest już nie wystarczające. Dodatkowe moduły mają za zadanie przewidywać przyszłe warunki ruchu.

5.5.8. RHODES – Real-Time Hierarchical Optimized Distributed Effective System

RHODES jest adaptacyjnym systemem sterowania ruchem ze struktura hierarchiczną. Został opracowany na Uniwersytecie w Arizonie [8,66,67]. RHODES składa się z dwóch podstawowych modułów: predykcji i sterowania (Rys. 10). Zadaniem modułu predykcji jest przewidywanie informacji o napływającym ruchu w przyszłości, kiedy i jak dużo pojazdów przybędzie. Natomiast moduł sterowania służy do sterowania przepływami ruchu na skrzyżowaniu i w sieci. Logika sterująca wykorzystuje algorytm opracowany przez S. Sena i K. L. Heada [68]. Algorytm ten został nazwany sterowaną optymalizacją faz (COP – Controlled Optimization of Phases) i jest oparty na metodzie programowania dynamicznego. Logika sterowania ruchem w sieci zastosowana w RHODES opiera się na COP i REALBAND [69]. Algorytm REALBAND został zastosowany

celem stworzenia zakresu postępu obserwowanego potoku w sieci. Ten zakres postępu jest następnie użyty jako ograniczenie dla COP, by określić optymalną strategię sterowania dla indywidualnego skrzyżowania [51].



Rys. 10: Architektura systemu RHODES

Chociaż algorytm COP jest oparty na programowaniu dynamicznym, używa definicji etapów, stanów i akcji, to jest on zupełnie inny niż metody DYPIC i OPAC. W COP etapy są zdefiniowane jako sekwencje faz ruchu, stany w każdym etapie są zdefiniowane jako liczba kroków czasowych, które mogłyby być przypisane do aktualnego etapu. Celem optymalizacji jest znalezienie optymalnego planu przypisania kroków czasowych do każdego etapu (fazy) tak, by całkowite opóźnienie pojazdu/liczba zatrzymań/długość kolejki mogłaby być zminimalizowana. Ta metoda modelowania jest podobna do zastosowania programowania dynamicznego do problemów związanych z alokacją zasobów. Sukcesem zarówno modelu OPAC jak i RHODES jest dokładne przewidywanie napływu ruchu w całym horyzoncie decyzji, co jest trudne w rzeczywistości [51].

5.5.9. UTOPIA – Urban Traffic Optimization by Integrated Automation

UTOPIA jest adaptacyjną metodą sterowania ruchem drogowym opracowanym przez włoskich naukowców [11,17,70]. Cechą charakterystyczną tej metody jest uwzględnienie priorytetu transportu publicznego. Struktura systemu jest podzielona na dwa poziomy: obszarowy i lokalny.

Celem sterowania obszarowego jest minimalizacja następującej funkcji

$$\min_{\delta_\tau^j, \varepsilon_\tau^j} \sum_{\tau=1}^{N} \sum_{j=1}^{lps} Kp^j(lp_\tau^j, \nu_\tau^j, \varepsilon_\tau^j) \tag{4}$$

gdzie:

j – indeks pasa $j = 1, \ldots, lps$,

$Kp^j(lp_\tau^j, \nu_\tau^j, \varepsilon_\tau^j)$ – funkcja kosztu dla pasa j,

lp_τ^j – liczba pojazdów na pasie j podczas trwania przedziału τ,

ε_τ^j – współczynnik odnoszący się do natężenia nasycenia na pasie j podczas trwania przedziału τ,

τ – indeks przedziału czasu,

ν_τ^j – współczynnik odnoszący się do średniej ogólnej prędkości na pasie j podczas trwania przedziału τ.

Sterownik obszarowy stale zapewnia zoptymalizowane wartości ν_τ^j i ε_τ^j dla sterowników lokalnych. Natomiast lokalne sterowniki próbują znaleźć optymalne rozwiązanie następującego problemu optymalizacyjnego:

$$\min_{s_\tau^i} \sum_{\tau=1}^{N} Ks^i(dk_\tau^i, s_\tau^i, \nu_\tau^i, \varepsilon_\tau^i) \tag{5}$$

gdzie:

dk_τ^j – wektor długości kolejek dla każdego pasa do skrzyżowania i podczas trwania przedziału τ,

i – indeks skrzyżowania,

Ks^i – funkcja kosztu dla skrzyżowania i,

s_τ^j – sygnał dla skrzyżowania i podczas przedziału τ,

τ – indeks przedziału czasu τ=1,...,N.

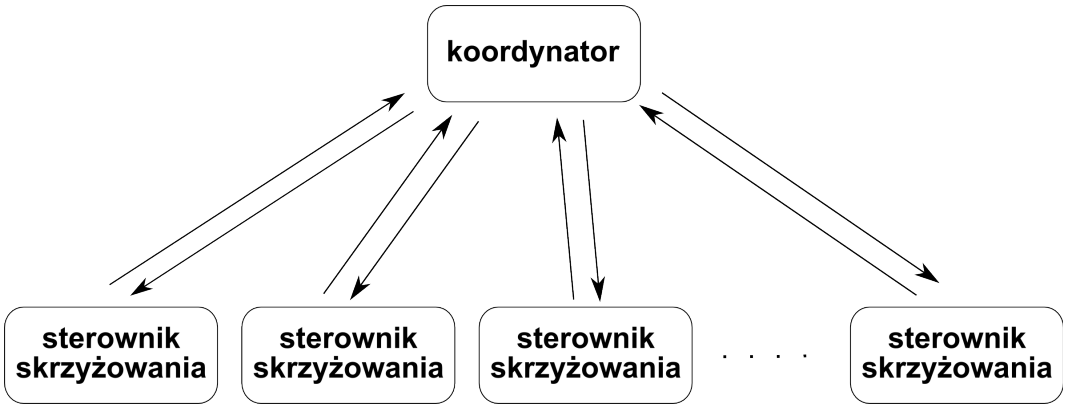
UTOPIA modeluje sterowanie odosobnionego skrzyżowania jako wieloetapowy problem decyzyjny. Całkowity horyzont decyzyjny wynosi zazwyczaj 120 sekund. Każdy etap (przedział) ma długość sześciu sekund. W metodzie UTOPIA jest zaadoptowana również technika zmiennego horyzontu [11].

Równania (4) i (5) są bardzo ogólnym opisem modelu optymalizacji zastosowanym w UTOPIA. W artykule F. Davidssona i C. D. Taranto [71] wspomniano o zastosowaniu algorytmu podziału i ograniczeń w UTOPIA celem rozwiązania problemu optymalizacji lokalnego sterownika [51].

5.5.10. PRODYN

PRODYN jest adaptacyjną metodą sterowania ruchem opracowaną przez zespół pod kierownictwem J. J. Henryego [51,72,73,74], która również jest oparta o programowanie dynamiczne. Podobnie jak inne adaptacyjne metody sterowania ruchem, PRODYN nie posiada stałej sekwencji faz ruchu, czasów trwania fazy i długości cyklu oraz używa hierarchicznego algorytmu do sterowania ruchem w sieci.

Powstały dwie wersje metody PRODYN [72,73,75]. Pierwsza wersja tej metody ma hierarchiczną strukturę z dwoma poziomami [72]. Zadaniem niższego poziomu jest sterowanie skrzyżowaniem, natomiast poziom wyższy ma za zadanie koordynację arterii i sieci. Proces optymalizacji pierwszej wersji jest pokazany na Rys. 11



Rys. 11: Budowa pierwszej wersji systemu PRODYN

Poziom niższy zawiera wiele sterowników skrzyżowania, które generują początkowe plany sterowania, które są następnie przesyłane do koordynatora wyższego poziomu. Koordynator wyższego poziomu przesyła do każdego skrzyżowania informacje korygujące, które sterowniki skrzyżowania wykorzystują do poprawy początkowego

planu sterowania. Jest to proces iteracyjny, który trwa do momentu, kiedy zostanie osiągnięty konsensus między koordynatorem a sterownikami niższego poziomu.

Hierarchiczna struktura pokazana na Rys. 11 została porzucona w późniejszej wersji PRODYN [73,75]. Nowa wersja wykorzystuje algorytm forward metody programowania dynamicznego (forward dynamic programming) [76] dla indywidualnego sterowania skrzyżowaniem i zdecentralizowaną strukturę dla koordynacji różnych sterowników skrzyżowania. Sterowanie pojedynczym skrzyżowaniem jest zamodelowane jako wieloetapowy problem decyzyjny, celem użycia algorytmu forward metody programowania dynamicznego. Podobnie do metody OPAC, horyzont decyzyjny w metodzie PRODYN jest podzielony na wiele małych przedziałów czasowych. Każdy przedział jest etapem. Stany są scharakteryzowane przez numer zmiennych łącznie z aktualną fazą oraz długościami kolejek. Na każdym etapie jest podejmowana decyzja pozostania przy aktualnej fazie zielonej lub przełączenia do następnej fazy. Ponadto PRODYN korzysta z metody zmiennego horyzontu. Załóżmy, że długość każdego etapu (przedziału czasu) jest *T*, która zazwyczaj wynosi 5 [s]. Długość horyzontu decyzyjnego jest *NT*, gdzie *N* jest wartością całkowitoliczbową, która może wynosić 15. W etapie τ najlepsza polityka sterowania w przedziale czasu [$\tau T, (\tau+N)T$] jest określana przez aktualne stany skrzyżowania i napływy ruchu w przedziale [$\tau T, (\tau+N)T$].

PRODYN korzysta ze zdecentralizowanej metody koordynacji. W oparciu o kilka głównych określeń w [73,75], tę metodę zdecentralizowanej koordynacji można krótko opisać następująca procedurą:

Krok 1: wybierz jedno skrzyżowanie i zoptymalizuj jego sterowanie w całym horyzoncie decyzyjnym;

Krok 2: w oparciu o zoptymalizowane decyzje sterowania, symuluj ruch na wyjściu (wyjazdach) tego skrzyżowania w całym horyzoncie decyzyjnym;

Krok 3: wyślij otrzymane w wyniku symulacji wyjścia strumieni ruchu do niżej położonych (poniższych) skrzyżowań i przejdź do niżej położonych (poniższych) skrzyżowań;

Krok 4: w oparciu o wyjście strumieni otrzymane w wyniku symulacji i wysłane ze skrzyżowań położonych powyżej (powyższych), optymalizuj sterowanie ruchem dla aktualnego (poniższego) skrzyżowania;

Krok 5: przejdź do kroku 2.

Problemem tej procedury jest wybór pierwszego skrzyżowania. Innym problemem jest to, że skrzyżowanie poniższe może być także skrzyżowaniem powyższym, jeśli łączy drogi dwukierunkowe, co jest częstym przypadkiem. Rozważmy dwa sąsiadujące skrzyżowania. Jeśli wyjście strumienia ruchu otrzymane w wyniku symulacji dla skrzyżowania A jest wykorzystane do optymalizacji sterowania skrzyżowaniem B, wówczas wyjście strumienia ruchu ze skrzyżowania B będzie działać wstecznie na optymalność już zoptymalizowanego planu sterowania skrzyżowaniem A. W ten sposób interakcje między skrzyżowaniami A i B tworzą krąg. Proces interakcji może być potrzebny w algorytmie koordynacji metody PRODYN celem zagwarantowania, że równowaga może być ostatecznie osiągnięta.

5.5.11. ALLONS-D – Adaptive Limited Look-ahead Optimization of Network Signals – Decentralized

Isaac Porche w swojej rozprawie doktorskiej zaproponował zdecentralizowaną, adaptacyjną metodę sterowania ruchem drogowym zwaną ALLONS-D [51,77,78,79,80]. Jest ona oparta na algorytmie podziału i ograniczeń wykorzystującym strategie w głąb i wykorzystuje drzewo decyzyjne, które pomaga znaleźć najlepszą sekwencję sterowania [77]. Drzewo decyzyjne jest podobne do drzewa zastosowanego w metodzie DYPIC pokazanym na Rys. 8, w którym każdy węzeł odpowiada punktowi decyzyjnemu i posiada skojarzoną z nim wartość kosztu, podczas gdy łuk stanowi akcję sterowania. Rys. 8 przedstawia drzewo decyzyjne dla odosobnionego skrzyżowania z dwufazowym sterowaniem. W przypadku skrzyżowań z czterema lub więcej fazami, rozmiar drzewa decyzyjnego sprawia, że wyczerpujące metody przeszukiwania stają się niewykonalne w zastosowaniach w czasie rzeczywistym. Celem poprawienia efektywności przeszukiwania, ALLONS-D wykorzystuje do znalezienia najlepszej sekwencji sterowania metodę podziału i ograniczeń oraz specjalną technikę „obsłuż największy koszt” (STLC – Serve the Largest Cost). Cały proces optymalizacji ALLONS-D

może być podzielony na dwie części: budowanie początkowej ścieżki decyzji (sekwencji) oraz cofanie i eksploracja [51].

W części budowania ścieżki decyzji, realne ścieżki decyzji są tworzone przy użyciu techniki STLC. W zakresie techniki STLC, w każdym punkcie decyzyjnym faza sterowania, która ponosiła największe opóźnienia w ostatnim czasie powinna zostać przełączona na zieloną. Postępując za zasadą STLC, sekwencja decyzji jest tworzona do momentu, aż kolejka początkowa i przewidywany napływ ruchu będzie rozładowany. Zdaniem autora [77] technika STLC pozwala osiągnąć wynik bliski optymalnemu. Łatwo zauważyć, że im bliżej ścieżka początkowa jest ścieżki optymalnej, tym mniej jest potrzebnego czasu obliczeń na zadania wycofywania i eksploracji w drugiej części.

Najczęściej początkowa ścieżka decyzji nie jest optymalna. Z tego powodu potrzebny jest proces powrotu i eksploracji celem poprawienia początkowej ścieżki decyzji. Proces cofania jest podobny do rekursywnej metody wstecz stosowanej do rozwiązywania problemu programowania dynamicznego pokazane w równaniu (3) na str. 38. Jednak dodanie procesu eksploracji odróżnia go od rekursywnej metody wstecz. Cofanie i eksploracja jest rekursywnym procesem, który zaczyna się od ostatniego węzła początkowej ścieżki decyzji. Ostatniemu węzłowi odpowiada wartość kosztu zero i zakłada się, że wszystkie kolejki są rozładowane w tym punkcie. Początkowa ścieżka decyzji jest ustawiona jako Aktualnie Najlepsza Ścieżka Decyzji (CBDP – Current Best Decision Path). Proces przesuwa się wstecz jeden przedział na każdą iterację i oblicza wartość kosztu w aktualnym węźle. Dla każdego węzła z wyjątkiem ostatniego, wszystkie gałęzie wyrastające z niego są oceniane i porównywane z CBDP. Wartość kosztu jest zdefiniowana dla łuku i ścieżki. Skumulowane w czasie trwania każdego przedziału opóźnienie jest zdefiniowane jako koszt łuku. Natomiast koszt ścieżki jest sumą kosztów łuków, które należą do tej ścieżki. Jeśli jakaś gałąź ma mniejszy koszt niż gałąź w CBDP, wówczas ją zastępuje. W przeciwnym razie eksploracja w tym węźle będzie zakończona a proces przejdzie o jeden przedział wstecz i ustawi jako aktualny węzeł macierzysty.

Przedstawiony powyżej algorytm ALLONS-D dotyczy sterowania odosobnionym skrzyżowaniem. Dla sterowania ruchem w arterii Isaac Porche rozważał dwie metody [77]. Pierwsza metoda koordynacji przypisywała różne wagi dla każdego

kierunku. Przykładowo jeżeli główna droga jest w kierunku wschód-zachód, to większą wagę przypisywano dla kierunku wschód-zachód. I. Porche testował tę metodę sterowania dla arterii złożonej z trzech skrzyżowań. Jednakże wydaje się, że ta metoda nie sprawdza się najlepiej. Drugą metodą zaproponowaną przez I. Porche była teoria gier [81]. Została jednak przedstawiona jedynie koncepcja zastosowania teorii gier w sterownikach koordynujących. Nie przedstawiono żadnego eksperymentu, który mógłby potwierdzić możliwość zastosowania tej metody w sterownikach koordynujących. Nie ma dowodów na to, czy można stosować metodę teorii gier w systemach adaptacyjnych [51].

Terence Kelly w [80] rozważał zastosowanie różnych technik przeszukiwania drzewa decyzyjnego od AI do A* celem znalezienia optymalnej koordynacji skrzyżowań. Jednak stwierdził, że te techniki wymagają zbyt dużego rozmiaru pamięci. Ostatecznie przyjął technikę przeszukiwania w głąb.

5.5.12. Proces decyzyjny Markowa i programowanie dynamiczne (MDP&DP – Markov Decision Process and Dynamic Programming)

X.-H. Yu i W. W. Recker [82] opracowali stochastyczny, adaptacyjny model sterowania sygnalizacją świetlną. Autorzy sformułowali sterowanie ruchem drogowym jako proces decyzyjny Markova z czasem dyskretnym (MDP – Markov Decision Process) i rozwiązali go za pomocą algorytmu programowania dynamicznego. MDP jest to stochastyczny proces scharakteryzowany przez zbiór stanów (*S*), akcji (*A*), funkcję nagrody (*r*) oraz funkcję przejścia stanów (*p*). W odniesieniu do sterowania ruchem na pojedynczym skrzyżowaniu, zmiennymi stanu są długości kolejek wszystkich wjazdów; zmiennymi akcji są akcje sterowania, które mogą być przeprowadzone dla każdego stanu; funkcja nagrody zwraca natychmiastową nagrodę za wykonanie każdej akcji w określonym stanie; funkcja prawdopodobieństwa przejścia stanu jest zmienna w czasie i zależna od aktualnego napływu ruchu. Pierwszym krokiem w celu rozwiązania problemu zamodelowanego jako MDP jest znalezienie optymalnej funkcji wartości *V*(s)* w oparciu o równanie [62]

$$V^*(s) = \max_{a \in A(s)} \left\{ \sum_{s' \in S} p^a_{ss'} r^a_{ss'} + \gamma \sum_{s' \in S} p^a_{ss'} V^*(s') \right\}, \forall s \in S \qquad (6)$$

gdzie $a \in A(s); s \in S$ jest aktualnym stanem i $s' \in S$ jest kolejnym stanem po wykonaniu akcji *a*, $p^a_{ss'}$ i $r^a_{ss'}$ są prawdopodobieństwem przejścia i nagrodą odpowiednio ze stanu *s* do stanu *s'* po wykonaniu akcji *a*; $\gamma \in [0,1)$ jest współczynnikiem dyskontowym. Równanie (6) jest określane jako równanie optymalności Bellmana [62]. W oparciu o równanie Bellmana, X.-H. Yu i W. W. Recker [82] wykorzystali metodę programowania dynamicznego do rozwiązania optymalnej funkcji wartości $V^*(s)$. Po znalezieniu $V^*(s)$, problemy sterowania sprowadzają się jedynie do identyfikacji aktualnego stanu systemu *s* i zastosowania akcji sterowania $a \in A(s)$, co prowadzi do optymalnej funkcji wartości $V^*(s)$.

Chociaż algorytm programowania dynamicznego może być użyty do rozwiązania tego problemu MDP i jest zagwarantowane znalezienie optymalnej polityki (odwzorowania ze stanu systemu do akcji) [83], to jednak jest potrzeba dobrego zdefiniowania funkcji prawdopodobieństwa przejścia stanu. W praktyce funkcja prawdopodobieństwa przejścia stanu jest trudna i skomplikowana do zdefiniowania. W przypadku sterowania ruchem na skrzyżowaniu na funkcję prawdopodobieństwa przejścia stanu mają wpływ aktualne napływy ruchu i często jest zmienna w czasie. Z tego względu jest jeszcze trudniej przeprowadzić dokładną estymację. Na dodatek w zastosowaniach w sterowaniu ruchem na skrzyżowaniu liczba stanów jest zazwyczaj bardzo duża. Sprawia to, że czas obliczeń algorytmu programowania dynamicznego staje się poważnym problemem [82,83,84]. W przeciwieństwie do metod DYPIC i OPAC, które zakładają deterministyczne przejście stanów, MDP pośrednio potwierdza niepewność przejścia stanu i odzwierciedla tę niepewność funkcją prawdopodobieństwa przejścia stanu [51].

5.6. Sterowanie Ruchem Drogowych Z Wykorzystaniem Logiki Rozmytej I Reguł

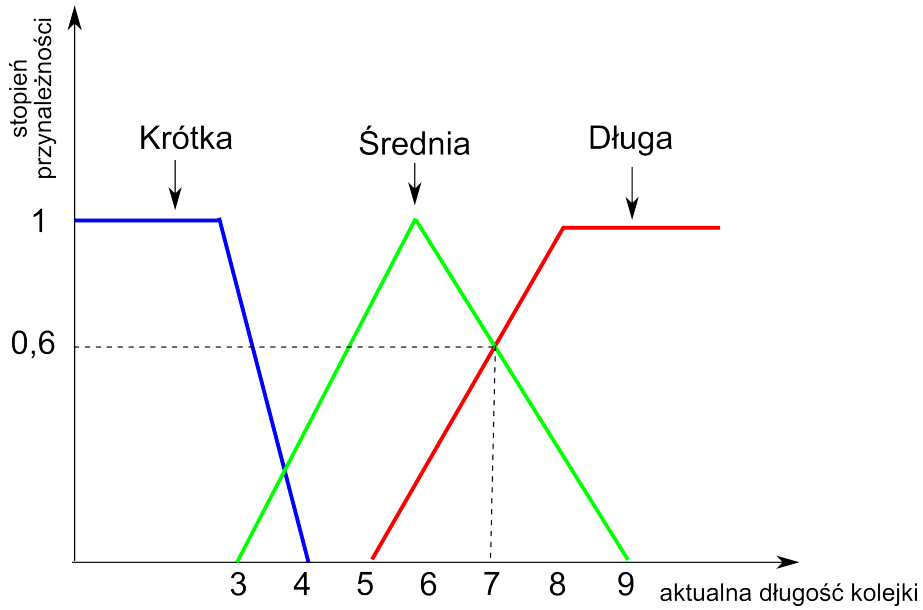
Logika rozmyta była stosowana wielokrotnie w sterowaniu ruchem drogowym [14,85,86,87,88,89,90,91,92,93,94]. Te metody logiki rozmytej wykorzystują długości kolejek oraz napływ ruchu na wszystkich dojazdach jako wejścia. Natomiast akcje sterowania są zazwyczaj określone na podstawie pewnej liczby reguł rozmytych. Poniżej

przedstawiono dwie przykładowe reguły rozmyte [95] użyte do rozszerzenia aktualnej fazy zielonej.

IF aktualna długość kolejki jest {Krótka} **AND** napływ jest {Niski} **AND** długość kolejki kolizyjnej jest {Średnia}, **THEN** rozszerzenie jest {Krótkie}

IF aktualna długość kolejki jest {Średnia} **AND** napływ jest {Wysoki} **AND** długość kolejki kolizyjnej jest {krótka}, **THEN** rozszerzenie jest {Długie}

Na początku wejścia systemu sterowania z zastosowaniem logiki rozmytej są rozmywane tak, by mogły być wykorzystane w regułach rozmytych. Rozmywanie wejść jest realizowane przez funkcje przynależności. Rys. 12 przedstawia rozmytą funkcję przynależności.



Rys. 12: Przykład rozmytej funkcji przynależności aktualnej kolejki pojazdów

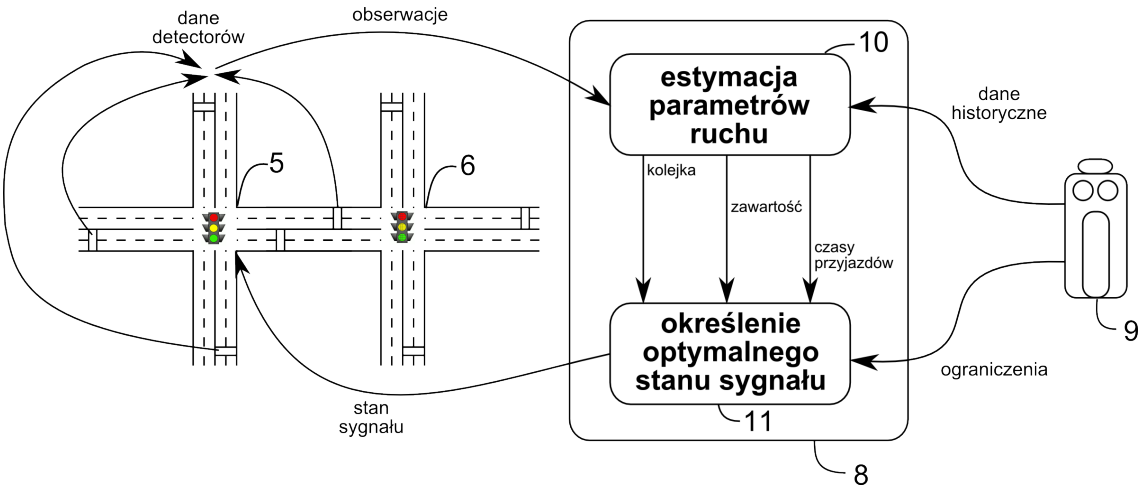
W oparciu o funkcję przynależności z Rys. 12, dla aktualnej długości kolejki o wartości 7, przynależność, że aktualna długość kolejki należy do {Krótka}, {Średnia} i {Długa} wynosi odpowiednio 0,0, 0,6 i 0,6. Podobnie można zastosować tę samą procedurę dla dwóch innych zmiennych wejściowych, napływu i długości kolizyjnej kolejki i uzyskać w ten sposób odpowiednie wartości przynależności. Wszystkie wartości przynależności będą użyte do obliczenia mocy każdej reguły rozmytej, która jest często nazywana siłą aktywacji (FS – firing strength) [96]. Każda reguła rozmyta odpowiada wyjściowemu zbiorowi rozmytemu. Te wyjściowe zbiory rozmyte wraz z odpowiadającymi im siłami aktywacji są wykorzystywane w celu uzyskania wartości pojedynczych (crisp value), ponieważ nie jest możliwe bezpośrednie użycie wyjść

lingwistycznych takich jak „rozszerzenie jest {Długie}” lub „rozszerzenie jest {Krótkie}” dla praktycznego sterowania ruchem.

Oczywistą zaletą wykorzystania logiki rozmytej dla sterowania ruchem drogowym jest minimalne zapotrzebowanie na zasoby obliczeniowe. Podobnie do sterowania stałoczasowego i zmiennoczasowego jest dużo bardziej efektywne obliczeniowo niż inne metody adaptacyjne. Inną dobrą własnością logiki rozmytej jest to, że może lepiej reprezentować aktualny stan systemu. Dla przykładu jak pokazano na Rys. 12 długość kolejki 7 nie może być jednoznacznie sklasyfikowana jako {Średnia} lub {Długa}. Przeciwnie, należy do obu {Średniej} i {Długiej}, każda z rangą 0,6. Te rozmyte wartości przynależności mogą przyczynić się do lepszej zdolności generalizacji rozmytego sterownika ruchu drogowego [51].

5.6.1. GASCAP – Generalized Adaptive Signal Control Algorithm Project

Wielu naukowców proponuje adaptacyjne systemy sterowania ruchem w oparciu o reguły i wiedzę [97,98,99,100,101,102,103]. Przykładowo Larry E. Owen i Charlie M. Stallard [97,98] opatentowali metodę GASCAP (Generalized Adaptive Signal Control Algorithm Project – Uogólniony Projekt Algorytmu Adaptacyjnego Sterowania Sygnałami). Cały proces GASCAP składa się z trzech podstawowych elementów. Pierwszym elementem jest wyszukana metoda estymacji kolejki. Drugim jest zbiór reguł dla warunków braku zatłoczenia, które wykorzystują oszacowania kolejek do określenia sygnałów na skrzyżowaniu. Trzecim elementem jest metoda obliczania splitów, offsetów i długości cykli, jeśli skrzyżowanie jest zatłoczone. Autorzy tej metody uznają skrzyżowanie za zatłoczone, jeżeli zajętość powyżej detektorów na przeciwstawnych wjazdach (np. w kierunku zachodnim i w kierunku południowym) jest większa niż 0,5. Zajętość detektora oznacza ilość czasu, jaką detektor jest aktywny do całkowitego czasu obserwacji. Zazwyczaj zajętość jest obliczana w piętnastominutowym przedziale czasu [97].



Rys. 13: Przebieg procesu GASCAP

Proces GASCAP przebiega jak pokazano na Rys. 13. Dane detektora są zbierane przez detektory i wejścia lokalnego sterownika ruchu 8. Sterownik ruchu 8 ma wbudowany procesor, na którym jest uruchomiony proces GASCAP. Sterownik ruchu 8 jest połączony przez sieć z centralnym komputerem 9, który dostarcza historyczne dane ruchu, takich jak procent skrętów oraz ograniczenia w pracy sygnalizacji świetlnej. Proces GASCAP pobiera w czasie rzeczywistym dane detektora jak również dane historyczne z centralnego komputera 9 i oszacowuje w bloku 10 długości kolejek lub liczbę pojazdów czekających na pasie przed skrzyżowaniem, zawartość. Długość kolejki odnosi się do liczby pojazdów na pasie, które czekają przed sygnalizatorem. Długość kolejki jest oszacowywana na podstawie aktywacji górnych detektorów. Zawartość pasa odnosi się do liczby pojazdów, które nie są w kolejce, między linią stopu a górnym detektorem i jest oparta na kombinacji aktywacji detektora, estymacji kolejki i współczynnika natężenia przepływu. Estymowane informacje o parametrach ruchu są następnie wykorzystane w bloku 11 do określenia optymalnego sygnału w oparciu o warunki ruchu w chwili obecnej.

Reguły dla sterowania w warunkach braku zatłoczenia z użyciem GASCAP można kategoryzować w cztery różne zbiory: reguły zgłoszenia, reguły postępu, reguły pilności i reguły współpracy. Piąty zbiór reguł, zwany regułami bezpieczeństwa, jest nierozerwalnie związany z innymi czterema zbiorami reguł.

- Reguły zgłoszenia wykorzystują informacje dotyczące kolejek i zawartości dla każdego potoku i wybierają jeden, który ma największą potrzebę rozładowania.
- Reguły postępu pozwalają na koordynację przyległych węzłów lub skrzyżowań.

- Reguły pilności sprawdzają, czy któryś z detektorów górnych nie był aktywny w sposób ciągły przez 15 minut. Jest to wymagane, gdy zgłoszenia ruchu i nasycenie na wjazdach wzrasta.
- Reguły współpracy pozwalają na współpracę sąsiadujących ze sobą skrzyżowań.
- Reguły bezpieczeństwa służą do określenia czasu trwania oczyszczania skrzyżowania, czuwania, by nie było stanów z kolizyjnymi fazami ruchu.

5.6.2. SPPORT – Signal Priority Procedure for Optimization in Real-Time

SPPORT (Signal Priority Procedure for Optimization in Real-Time – Procedura nadawania priorytetów sygnałom dla optymalizacji w czasie rzeczywistym) model został opracowany przez François Diona, i Bruca Hellinga [47,104,105,106,107,108] celem rozwiązania dwóch problemów. Autorzy zauważyli, że stosowane metody adaptacyjnego sterowania nie rozważają wpływu transportu komunikacji miejskiej (pierwszeństwa przejazdu, zatrzymywania w miejscach oznaczonych) na ruch. Zatrzymanie autobusu przy przystanku może częściowo lub całkowicie zablokować ruch na pasie. Powoduje to niewydajne wykorzystanie czasu światła zielonego. Ponadto przyznanie pierwszeństwa przejazdu pojazdowi uprzywilejowanemu często następuje bez rozważenia wpływu nagłej zmiany sygnałów na ogólny ruch.

Początkowo pomysł owocnie zastosowano w sterowaniu odosobnionym, dwufazowym skrzyżowaniem. Zastosowano wtedy heurystyczną procedurę operowania sygnałami opartą na regułach, która tworzy plany czasowe na podstawie pewnej liczby zaproponowanych strategii. Procedura opiera się na stwierdzeniu, że przełączenie sygnału następuje po wystąpieniu określonych, dyskretnych zdarzeń ruchu drogowego. Poprzez ignorowanie zdarzeń, które nie mają znaczenia dla działania sygnału, następuje redukcja liczby potencjalnych kombinacji przełączania, które należy rozważyć celem znalezienia rozwiązania problemu sterowania ruchem. Ze względu na to, że różne zdarzenia mają różną ważność, SPPORT wymaga od użytkownika nadania priorytetów zdarzeniom. Pozycja na liście ma wpływ na to, czy zdarzenie otrzyma sygnał zielony, czy nie. Im wyższa pozycja tym większe prawdopodobieństwo na otrzymanie sygnału zielonego. Program ma możliwość wstępnej oceny każdej sekwencji faz ruchu powstałej na podstawie odpowiedniej listy priorytetów przy użyciu predefiniowanej funkcji kosztu. Umożliwia to dynamiczne wybieranie najbardziej obiecującego planu.

Proces optymalizacji zawiera następujące reguły [106]:

- reguły przełączania sygnałów (odpowiedź na zatrzymanie kolejki, odpowiedź na potencjalne zablokowanie kolejki, odpowiedź na przyjazd grupy pojazdów, odpowiedź na przyjazd pojazdu komunikacji miejskiej),
- reguły nadawania priorytetu – dla każdego odcinka drogi użytkownik ma obowiązek przypisać dodatnią wartość liczbową dla każdej z reguł. Im większa wartość tym większy priorytet zdarzenia.

Rozszerzenia modelu do zastosowań dla skrzyżowań ze sterowaniem wielofazowym dokonał M. Conrad [47,104]. Modyfikacja dotyczyła rozszerzenia procedury optymalizacji sygnałów w oparciu o reguły oraz ponownego zaprojektowania modułu mikrosymulacji ruchu. Przeprowadzone badania dla skrzyżowania odosobnionego wykazały wyższość modelu nad sterowaniem stałoczasowym i zmiennoczasowym w różnych warunkach ruchu [107].

Kolejnymi modyfikacjami było rozszerzenie procesu estymacji zgłoszeń ruchu i modułu optymalizacji sygnałów w oparciu o reguły celem przystosowania modelu SPPORT do sterowania skrzyżowaniem w skoordynowanej sieci [105,108]. Moduł optymalizacji sygnału został rozszerzony o metodę określania najwłaściwszych czasów dla decyzji przełączenia sygnałów, które rozważają wpływ obustronnej koordynacji.

5.7. Sterowanie Ruchem Drogowym Z Wykorzystaniem Sieci Neuronowych I Algorytmów Genetycznych

Badacze w przeciągu ostatnich kilkunastu lat podjęli wiele prób wykorzystania sieci neuronowych i algorytmów genetycznych w sterowaniu ruchem drogowym. W jednych projektach zastosowanie znalazły sieci neuronowe [109,110,111,112,113] lub algorytmy ewolucyjne [13,114,115,116], w innych zastosowanie sieci neuronowych zostało połączone z zastosowaniem algorytmów genetycznych [14,117,118] lub logiki rozmytej [14,51,117,119].

5.7.1. Metoda Tahere Royani i in.

Tahere Royani, Javad Haddadnia i Mohammad Alipoor w [14] zaproponowali zastosowanie rozmytej sieci neuronowej do sterowania sygnałami w czasie

rzeczywistym na przykładzie odosobnionego skrzyżowania z czterema trzypasmowymi wjazdami i czterema wyjazdami. Zastosowanie rozmytych sieci neuronowych łączy korzyści wynikające z zastosowania rozmytego systemu ekspertowego (wnioskowanie rozmyte) i sztucznej sieci neuronowej (samokształcenie). Autorzy tego pomysłu uważają, że rozmyte sieci neuronowe mogą być zastosowane do optymalnego sterowania zmiennego natężenia ruchu w warunkach przesycenia lub nietypowego przeciążenia. Celem było zwiększenie przepustowości oraz zmniejszenie opóźnień. Reguły rozmytego sterownika sformułowano według tego samego protokołu, którym posługują się operatorzy do sterowania przedziałami czasu sygnalizacji świetlnej. Dopasowanie parametrów rozmytej sieci neuronowej przeprowadzono za pomocą algorytmu genetycznego.

Zmienne wejściowe sterownika rozmytego zaprojektowano w następujący sposób: jednym wejściem jest liczba pojazdów przyjeżdżających w trakcie trwania aktualnej fazy zielonej; drugą zmienną jest liczba pojazdów w kolejce w trakcie trwania aktualnej fazy zielonej; trzecią zmienną jest liczba pojazdów w kolejce w trakcie trwania aktualnej fazy czerwonej. Zmienną wyjściową jest długość czasu rozszerzenia aktualnej fazy zielonej. Dla tych wejść zaproponowano 48 reguł. Dla przykładu wybrano regułę:

IF kolejka fazy zielonej jest {Krótka} **AND** kolejka fazy czerwonej jest {Krótka} **AND** liczba nadjeżdżających pojazdów w fazie zielonej jest {Niska} **THEN** długość czasu rozszerzenia fazy zielonej jest {Krótka}

Zastosowana rozmyta sieć neuronowa posiada pięć warstw:

- pierwsza warstwa wejściowa; węzły reprezentują lingwistyczne zmienne wejściowe (długość kolejki w trakcie trwania aktualnej fazy zielonej, długość kolejki w trakcie trwania aktualnej fazy czerwonej, liczba pojazdów przyjeżdżających w trakcie trwania aktualnej fazy zielonej)
- warstwa druga jest warstwą funkcji przynależności (membership function); każdy węzeł tej warstwy reprezentuje funkcję przynależności wartości lingwistycznej skojarzonej z wejściową zmienną lingwistyczną;
- warstwa trzecia odpowiada bazie reguł (rule base);
- warstwa czwarta odpowiada konkluzji reguł (rule consequents);
- warstwa piąta jest warstwą wyostrzania (defuzzification) .

Uczenie sieci przeprowadzono z użyciem algorytmu wstecznej propagacji z metodą największego spadku. W fazie uczenia wykorzystano koncepcję algorytmu genetycznego celem minimalizacji funkcji najmniejszych kwadratów błędów. W trakcie uczenia wykorzystano próbki rzeczywiste uzyskane z sekwencji wideo ruchu. Fazę uczenia sieci zaproponowaną w tej metodzie można przedstawić w następujących punktach:

Krok 1: Pobierz parametry algorytmu sterowania oraz wartości próbek do uczenia.

Krok 2: Zdefiniuj błąd kwadratowy jako funkcję oceny algorytmu genetycznego i wagi rozmytej sieci neuronowej jako jej zmienne.

Krok 3: Utwórz populację początkową oraz osobniki.

Krok 4: Oceń sprawność wszystkich osobników
Wybierz sprawniejsze osobniki do reprodukcji
Krzyżuj osobniki
Mutuj osobniki
Oceń sprawność zmodyfikowanych osobników
Stwórz nową populację

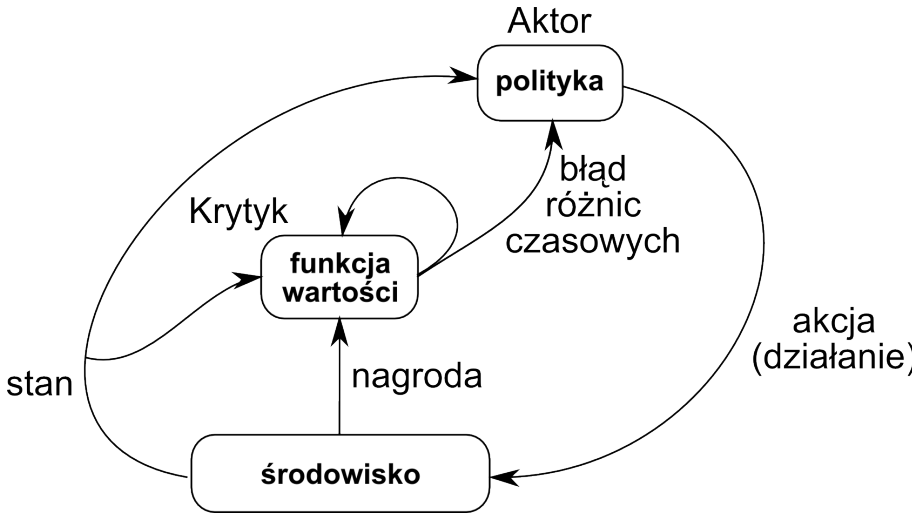
Krok 5: Jeżeli warunki zakończenia są spełnione i błąd uczenia jest mniejszy od wymaganego, wówczas zakończ uczenie sieci i wagi są wyprowadzone. W przeciwnym razie wróć do kroku 4 i kontynuuj uczenie sieci.

W [14] przedstawiono porównanie wyników sterowania uzyskanych w trakcie symulacji z użyciem powyższej metody oraz wyników symulacji przeprowadzonej z użyciem stałoczasowej metody sterowania. Porównanie wykazało prawie dwukrotne zmniejszenie średniego czasu oczekiwania pojazdów (330 sekund dla metody stałoczasowej i 178 sekund dla zaproponowanej metody w [14]). Badania symulacyjne przeprowadzono dla 1000 samochodów.

5.7.2. NFACRL – metoda Y. Xie

Innym przykładem zastosowania logiki rozmytej i sieci neuronowych w sterowaniu ruchem w czasie rzeczywistym jest projekt Yuanchanga Xie zaprezentowany w rozprawie doktorskiej [51]. Praca przedstawia teoretyczne porównanie istniejących metod adaptacyjnego sterowania ruchem drogowym z metodą neuro-rozmytego uczenia się ze wzmocnieniem z algorytmem Aktor-Krytyk

(NFACRL – Neural-Fuzzy Actor-Critic Reinforcement Learning) [120]. Metoda NFACRL została początkowo przetestowana na skrzyżowaniu odosobnionym. Rozpatrzono dwa różne schematy implementacji. Pierwszy schemat wykorzystywał stałą sekwencją faz ruchu i zmienną długość cyklu, podczas gdy drugi optymalizował sekwencję faz w czasie rzeczywistym i nie był ograniczony do koncepcji cyklu. Oba schematy następnie zostały rozszerzone do sterowania arterią, w której każde skrzyżowanie było pod kontrolą sterownika NFACRL. Porównania dokonano przy pomocy aplikacji symulacyjnej VISSIM. Wykorzystano rzeczywiste dane ruchu. Wyniki symulacji sterownika NFACRL porównano z wynikami symulacji przeprowadzonych z użyciem sterowania stałoczasowego i zmiennoczasowego.



Rys. 14: Architektura metody uczenia ze wzmocnieniem Aktor-Krytyk

Metoda NFACRL składa się z czterech warstw:

- pierwsza warstwa wejściowa; węzły tej warstwy odbierają wartości zmiennych stanu (8 zmiennych to długości kolejek wszystkich wjazdów, jedna zmienna to aktualny sygnał dla skrzyżowania z ośmioma pasami wjazdowymi) i przesyłają je do różnych funkcji przynależności warstwy drugiej;
- węzły warstwy drugiej są zbiorami rozmytymi, z których każdy jest skojarzony z rozmytą funkcją przynależności;
- warstwa trzecia odpowiada regułom rozmytym; wyjścia tej warstwy odpowiadają siłom aktywacji;
- czwarta warstwa jest zbiorem węzłów reprezentujących konkluzje reguł; pierwszy węzeł oznacza krytyka (Rys. 14) i jego wartość wyjściowa wskazuje jak

dobry jest aktualny stan; pozostałe węzły odpowiadają dostępnych akcjom, które mogą być powzięte

Y. Xie w swojej pracy [51] uważa, że uczenie ze wzmocnieniem ma potencjał, by mogło być zastosowane w realistycznym, adaptacyjnym sterowaniu ruchu w arterii. Wyniki przeprowadzonych symulacji potwierdzają wzrost średniej prędkości w sieci i redukcję opóźnienia w porównaniu do skoordynowanego sterowania stałoczasowego i zmiennoczasowego. Y. Xie przyznaje, że metoda przez niego zaproponowana nie działa dobrze, jeśli za kryterium weryfikacji zostanie uznane liczba zatrzymań przypadająca na jeden pojazd oraz czas przejazdu przez arterię.

5.7.3. TRAVOLUTION

Projekt TRAVOLUTION (nazwa pochodzi od TRAffic i eVOLUTION) [13,114,121,122,123] był rozwijany w latach 2006-2008 we współpracy Urzędu Miasta Ingolstadt, Audi, GEVAS Software i Uniwersytetu Technicznego w Monachium. Celem tego projektu było stworzenie systemu koordynacji arterii w mieście jak również systemu informowania kierowców. W 2010 r. na targach technologii ruchu INTERTRAFFIC w Amsterdamie projekt TRAVOLUTION otrzymał nagrodę za innowację w kategorii „Systemy Współpracujące”.

Stworzono algorytm genetyczny (GA) do optymalizacji całej sieci drogowej, który został zaimplementowany do zmiennoczasowego sieciowego sterowania ruchem BALANCE [124] i wprowadzony w mieście Ingolstadt. Proces optymalizacji składa się z następujących głównych elementów:

- model ruchu drogowego i model działania [123],
- funkcja celu,
- metoda optymalizacji (GALOP) [123].

Proces sterowania rozpoczyna się rejestrowaniem aktualnego stanu ruchu drogowego za pomocą pomiarów w sieci drogowej. Wyniki pomiarów są następnie poddawane testowi zgodności i grupowane pod względem obszaru.

Na podstawie wyników pomiarów tworzony jest makromodel ruchu drogowego, który stanowi przestrzenno-czasową reprezentację aktualnego natężenia ruchu drogowego. Po czym tworzone są profile strumieni przy pomocy mezoskopowego modelu ruchu strumienia dla wszystkich tras sieci sterowanej cyklicznie [13].

Model działania służy do prognozowania wpływu poszczególnych alternatyw sterowania dla następnego przedziału czasowego. Istotnymi parametrami działania, które można obliczyć są: czasy postoju, ilość zatrzymań i długość kolejek. Parametry działania tworzone są przez dwa częściowe modele [124]:

- z uzyskanych z mezomodelu profili strumienia ruchu obliczany jest deterministyczny udział parametrów działania przez uwzględnienie sygnalizacji świetlnej, czasu podróży i rozproszenia potoku,
- stochastyczne wahania i przeciążenia odwzorowane są za pomocą modelu kolejek.

Dzięki rozdzielczości czasowej o jednosekundowych odstępach jest możliwe modelowanie wpływu ruchu na czas trwania światła zielonego grup sygnałowych oraz przesunięcia czasowego między sąsiednimi sygnalizacjami. Suma oddziaływań modelu deterministycznego i stochastycznego wykorzystywana jest następnie w obliczaniu funkcji celu.

Funkcja celu otrzymuje parametry działania z modelu ruchu oraz modelu działania i zwraca jako wynik sprawność jednego osobnika, tj. jedną wartość skalarną dla jednej alternatywy sterowania. Celem zaś jest minimalizacja czasu oczekiwania. Funkcja celu F została zdefiniowana następująco:

$$\min F = \sum_{sg \in SG} \varpi_{sg} \cdot W_{sg} \tag{7}$$

gdzie:

ϖ_{sg} – waga dla grupy sygnalizacyjną *sg,*

W_{sg} – czas oczekiwania przed grupą sygnalizacyjną *sg,*

SG – zbiór grup sygnalizacyjnych.

Jako metodę optymalizacji wykorzystano algorytm genetyczny. W metodzie optymalizacji GALOP [124] alternatywy sterowania są reprezentowane poprzez tzw. kodowanie w postaci osobników. Jeden osobnik ma następującą postać:

$$\left\{\psi,(\sigma_1,\omega_1,o_1,\Theta_{11},\ldots,\Theta_{1m_1}),(\sigma_2,\omega_2,o_2,\Theta_{21},\ldots,\Theta_{2m_2}),\ldots,(\sigma_n,\omega_n,o_n,\Theta_{n1},\ldots,\Theta_{nlg_n})\right\}$$

Zawiera gen ψ służący ustaleniu wspólnego czasu trwania cyklu oraz *n* chromosomów dla *n* węzłów sieci, która ma być optymalizowana. Każdy chromosom składa się z jednego genu σ celem ustalenia kolejności faz ruchu, jednego genu ω dla

globalnego offsetu, jednego genu o dla lokalnego offsetu i z *lg* genów Θ dla czasów trwania faz albo punktów startu przejścia faz. Każdy gen przyjmuje wartość rzeczywistą pomiędzy 0 i 1 [13].

Z powodu ograniczeń czasów przesunięć i warunków lokalnych sterowań zmiennoczasowych, geny dla globalnego i lokalnego offsetu w realizowanej implementacji algorytmu w TRAVOLUTION nie zostały uaktywnione.

Jeden węzeł w osobniku przedstawiony jest jako zestaw genów (chromosomów). Podczas krzyżowania wykorzystuje się losowo wybrane pojedyncze geny osobników rodzicielskich celem stworzenia osobników potomnych. Operator krzyżowania został rozszerzony w ten sposób, że z prawdopodobieństwem *p* krzyżowanie jest przeprowadzone na wszystkich pozostałych chromosomach (węzłach). Mutacja odbywa się dla jednego genu lub dla całego osobnika, a w razie potrzeby dopasowywana jest adaptacyjnie [124].

Opracowany system przetestowano w Ingolstadt na obszarze liczącym 48 sygnalizacji świetlnych. Przetestowano i porównano wówczas lokalne sterowanie zmiennoczasowe optymalizowane przez BALANCE z algorytmem GALOP, lokalne sterowanie zmiennoczasowe optymalizowane poprzez BALANCE z algorytmem Hill-Climbing oraz sterowanie zastosowane przed wprowadzeniem adaptacyjnego sterowania sieciowego. Szczegóły przeprowadzenia tych badań zostały opisane w [114,124]. W wyniku przeprowadzonych badań na podstawie średniej dobowej osiągnięto redukcję strat czasu w stosunku do algorytmu Hill-Climbing o 10% oraz zmniejszenie liczby zatrzymań o 8%. Bardziej szczegółowe rezultaty badań metody GALOP przedstawiono w [114,124].

5.8. Wnioski

Główną wadą strategii stałoczasowych jest bazowanie na danych historycznych, a nie na danych aktualnych. Jest to dużym uproszczeniem, które w warunkach nasycenia i przesycenia może wpływać niekorzystnie na przepływ ruchu. W drogowych sieciach miejskich ruch jest zmienny w ciągu dnia jak również w przeciągu tygodnia. Okazuje się, że metody zmiennoczasowe również nie są wystarczające, dlatego poszukuje się metod działających w czasie rzeczywistym.

Metody adaptacyjnego sterowania ruchem mają najczęściej złożoną budowę hierarchiczną. Towarzyszą temu skomplikowane algorytmy o dużej złożoności czasowej. Niektóre z użytych algorytmów mają również duże wymagania dotyczące rozmiaru pamięci operacyjnej. Niektóre z wyżej opisanych systemów już w fazie badań dla odosobnionego skrzyżowania okazują się nieprzydatne do zastosowań w czasie rzeczywistym. Inne systemy jak SCATS, SCOOT są rozwijane i znajdują zastosowanie do sterowania w sieciach miejskich liczących nawet kilka tysięcy skrzyżowań. Aktualnie rozwijane są systemy zarządzające całymi sieciami i wykorzystujące stan sieci w czasie rzeczywistym jak również w przyszłym. Poszukiwane są metody potrafiące przetwarzać duże ilości danych z detektorów w względnie krótkim czasie oraz umożliwiające płynność ruchu pojazdów w sieci. Metody sterowania powinny być również przygotowane do sterowania ruchem w przypadku wystąpienia zdarzeń wyjątkowych jak kolizje, remonty itp.

6. MODELE SYMULACYJNE

6.1. Wprowadzenie

Po raz pierwszy symulacje komputerowe do badania procesów ruchu pojazdów w czasie i przestrzeni zastosowano w latach pięćdziesiątych. Głównymi zaletami tej techniki jest powtarzalność i elastyczność eksperymentu. Umożliwia ona badanie skomplikowanych procesów, których nie można rozwiązać metodami analitycznymi oraz zastępuje pracochłonne długotrwałe, a czasem nawet niemożliwe do wykonania obserwacje rzeczywistych procesów ruchu. Wady natomiast wynikają z uproszczenia rzeczywistości. Głównymi wadami jest czasochłonność programowania oraz ograniczone możliwości uogólnień wyników. Najczęściej symulacja komputerowa jest stosowana do testowania istniejących systemów sterowania w różnych zmieniających się warunkach oraz do oceny nietypowych rozwiązań projektowych [17].

Modele symulacyjne, zależnie od ich cech, dzielą się na [17,125,126,127]:

- dynamiczne, w których czynnik czasu ma kluczowe znaczenie. Właściwości i atrybuty systemu są zależne od odwzorowanego w symulacji czasu,
- statyczne, w których czas nie wpływa na wynik symulacji, gdyż stan systemu nie jest zależny od czasu i atrybuty systemu nie zmieniają się wraz z czasem symulacji,
- stochastyczne, w których występują zmienne losowe decydujące o przebiegu odwzorowanych procesów czyli występowania danych zdarzeń,
- deterministyczne, w których nie występują zmienne losowe, a działanie modelu i cechy obiektów są zdefiniowane wcześniej lub obliczane na bieżąco według zadanych funkcji matematycznych. Na przykład model jazdy za liderem [17,126,127] może być sformułowany jako deterministyczny przez zdefiniowanie czasu reakcji kierowcy jako wartości stałej, ale przyjmując dla tego czasu określoną funkcję prawdopodobieństwa, model będzie klasyfikowany jako stochastyczny.

Modele symulacyjne mogą być także klasyfikowane pod względem złożoności i wierności, jaką opisują rozważany system [17,126,127]:

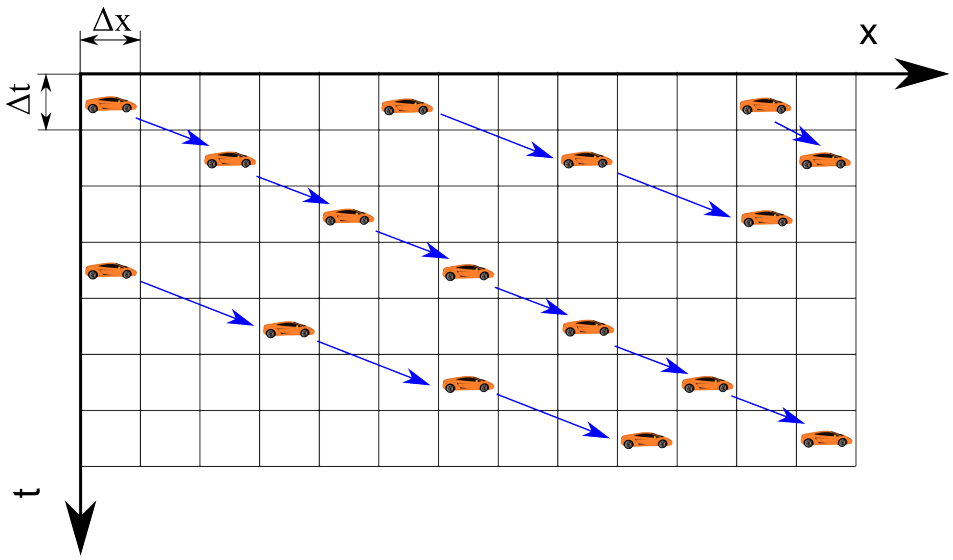
- mikroskopowe opisują każdy pojazd indywidualnie, jego prędkość, położenie, punkt docelowy podróży, manewry kierowcy itp. Wśród tych modeli ważną rolę odgrywają modele odstępów między pojazdami poruszającymi się w kolumnach. Przykładami modeli mikroskopowych są: model jazdy za liderem (car-following model [128]) oraz model teorii kolejek [17].
- mezoskopowe mają na celu zachowanie dokładności odwzorowania rzeczywistości w modelach mikroskopowych oraz zwiększenie efektywności obliczeniowej. Istnieją dwa podejścia do mezoskopowych symulacji ruchu. Pierwsze podejście nie bierze pod uwagę każdego pojazdu z osobna, lecz pojazdy są pogrupowane w tzw. plutony (jednakże pluton może składać się z jednego pojazdu), które poruszają się wzdłuż odcinka drogi (np. CONTRAM [129]). Drugie podejście zakłada opis dynamiki przepływu przez uproszczoną dynamikę każdego pojazdu indywidualnie (np. DTASQ [130], MEZZO [131]).
- makroskopowe rozpatrują strumień ruchu a nie pojazd lub grupę pojazdów (pluton). Podstawowymi charakterystykami strumienia ruchu są: natężenie ruchu, gęstość ruchu i średnia prędkość. Prędkość odnosi się do zbioru pojazdów uczestniczących w ruchu i w opisie strumienia ruchu wyrażona jest przez wartość średnią [17].

6.2. Automaty Komórkowe

Automaty komórkowe [17,132,133,134,135,136,137,138,139] są matematyczną idealizacją fizycznych systemów, w których przestrzeń, czas i inne wielkości fizyczne są dyskretne. Automat komórkowy składa się z regularnej jednolitej siatki (lub tablicy), zwykle o nieskończonym rozmiarze, z dyskretną zmienną w każdej komórce. Stan automatu komórkowego jest określony przez wartości zmiennych w każdej komórce. Stan automatu komórkowego zmienia się w dyskretnych odstępach czasowych. Na wartość zmiennej danej komórki mają wpływ wartości zmiennych sąsiednich komórek w poprzednim kroku czasowym. Zmienne wszystkich komórek automatu uaktualniane są jednocześnie w oparciu o wartości zmiennych komórek sąsiednich z poprzedniego kroku czasowego, zgodnie z przyjętym zestawem reguł [133].

Po raz pierwszy automaty komórkowe zostały użyte w 1948 roku przez Johna von Neumana i Stanisława Ulama w celu zamodelowania biologicznej samoreprodukcji [134]. Natomiast dla celów symulacji ruchu drogowego model automatów komórkowych został wykorzystany w 1992 roku przez Kai Nagela i Michaela Schreckenberga [137]. Stworzyli oni model zdefiniowany jako jednowymiarowa tabela o L komórkach z otwartymi lub periodycznymi warunkami brzegowymi. Droga więc została podzielona na odcinki. Każdy odcinek jest tej samej długości i odpowiada 7,5m (typowa długość pojazdu osobowego według autorów). Każda komórka może być zajęta tylko przez jeden pojazd lub może być pusta. Również czas został podzielony na dyskretne przedziały o wartości 1 sekundy. Każdy pojazd ma całkowitoliczbową prędkość o wartości od zera do v_{max}. Przykładowo wartość prędkości 1 odpowiada 7,5 m/s, natomiast 3 odpowiada 22,5 m/s. Płynny ruch pojazdów w rzeczywistości został zamieniony na ruch skokowy. Każde uaktualnienie systemu składa się z czterech następujących kroków, które wykonywane są równolegle dla wszystkich pojazdów [137]:

- przyspieszenie: jeżeli prędkość v pojazdu jest mniejsza niż v_{max} i odległość do następnego pojazdu z przodu jest większa niż v+1, wówczas prędkość jest zwiększana o 1 [$v \rightarrow v$+1],
- spowolnienie (ze względu na inne pojazdy): jeśli pojazd w komórce x widzi następny pojazd w komórce x+y (z $y \leq v$), wówczas zmniejsza prędkość do y–1 [$v \rightarrow y$–1],
- randomizacja: z prawdopodobieństwem p, prędkość każdego pojazdu (jeśli większa niż zero) jest zmniejszana o 1 [$v \rightarrow v$–1],
- ruch samochodów: każdy pojazd jest przesuwany o v komórek.



Rys. 15: Przykład symulacji ruchu za pomocą automatu komórkowego

Przykładowy ruch pojazdów zamodelowany przy pomocy automatu komórkowego został zilustrowany na Rys. 15.

Model ten jednak nie przewidywał skrzyżowań. Rozszerzeniem tego modelu był model ruchu dla sieci dwuwymiarowych dróg stworzony przez Bastiena Choparda [140]. Model ten zakładał podobne reguły do reguł modelu Nagela–Schreckenberga z wyjątkiem reguł pojazdów znajdujących się w pobliżu skrzyżowania. Celem uproszczenia przejazdu pojazdów przez przecięcia dróg, każde przecięcie zostało zamodelowane jako rondo. Pojazd będący na rondzie poruszał się zgodnie z kierunkiem wskazówek zegara i miał pierwszeństwo przed pojazdami wjeżdżającymi. Więcej reguł zostało opisanych w [140].

Oprócz wcześniej przedstawionych prostych modeli ruchu, w których wykorzystano automaty komórkowe, powstało wiele udoskonaleń i rozszerzeń. Obszerny przegląd automatów komórkowych ruchu drogowego został przedstawiony w [132].

6.3. Systemy Multiagentowe

Określenie agent przypisywane jest do systemów komputerowych umieszczonych w pewnym środowisku, które są zdolne do samodzielnych działań w tym środowisku, by osiągnąć cel projektu [141,142,143]. Agent powinien więc mieć zdolność podejmowania samodzielnych decyzji, które pomogą mu wykonać zadania jako odosobnionej jednostce. Ponadto jest umieszczany w trudnych warunkach

o dynamicznej, nieprzewidywalnej i nierzetelnej charakterystyce. Inteligentny agent powinien cechować się również następującymi właściwościami [142]:

- proaktywność – agent nie tylko powinien reagować w odpowiedzi na zmiany środowiska, ale powinien wykazywać inicjatywę w kierunku dążenia do postawionego celu;
- reaktywność – agent postrzega swoje środowisko (którym może być fizyczne otoczenie, użytkownik przez graficzny interfejs użytkownika, zbiór innych agentów, internet lub połączenie wszystkich poprzednich) i reaguje w odpowiednim czasie do zmian zachodzących w nim;
- umiejętności społeczne – agenci oddziałują wzajemnie na siebie (także z ludźmi) przez pewnego rodzaju język komunikacji agentów [144].

Kiedy w jednym środowisku zostaje umieszczonych kilku agentów, określani są wówczas jako multiagenci. Systemy multiagentowe są częścią sztucznej inteligencji, której celem jest zarówno budowanie kompleksowych systemów złożonych z wielu agentów oraz mechanizmów koordynacji zachowań niezależnych agentów [145]. Koordynacja działań agentów sprzyja eliminacji negatywnego wzajemnego wpływu na aktywność jak również wydobyciu potencjału synergii.

7. ŚRODOWISKO SYMULACYJNE

7.1. Green Light District (GLD)

7.1.1. Ogólna charakterystyka

Badania symulacyjne przeprowadzono za pomocą zmodyfikowanego symulatora GLD [3,146,147,148,149,150,151]. Jest to aplikacja stworzona w języku JAVA [152,153] z otwartym kodem na licencji GPL (GNU General Public License [154]). GLD jest mikroskopowym modelem ruchu, w którym każdy pojazd jest symulowany indywidualnie. Zmienne dynamiczne modelu reprezentują własności mikroskopowe takie jak położenie i prędkość każdego pojazdu. Pojazdy poruszają się przez sieć drogową zgodnie z ich fizykalną charakterystyką (długość, prędkość, liczba pasażerów itp.), podstawowymi regułami ruchu i predefiniowanymi regułami zachowań kierowców. Symulator GLD stworzony jest w oparciu o automat komórkowy, w którym dyskretne i częściowo połączone komórki mogą zajmować różne stany. Komórka drogi może być zajęta przez pojazd lub może być pusta.

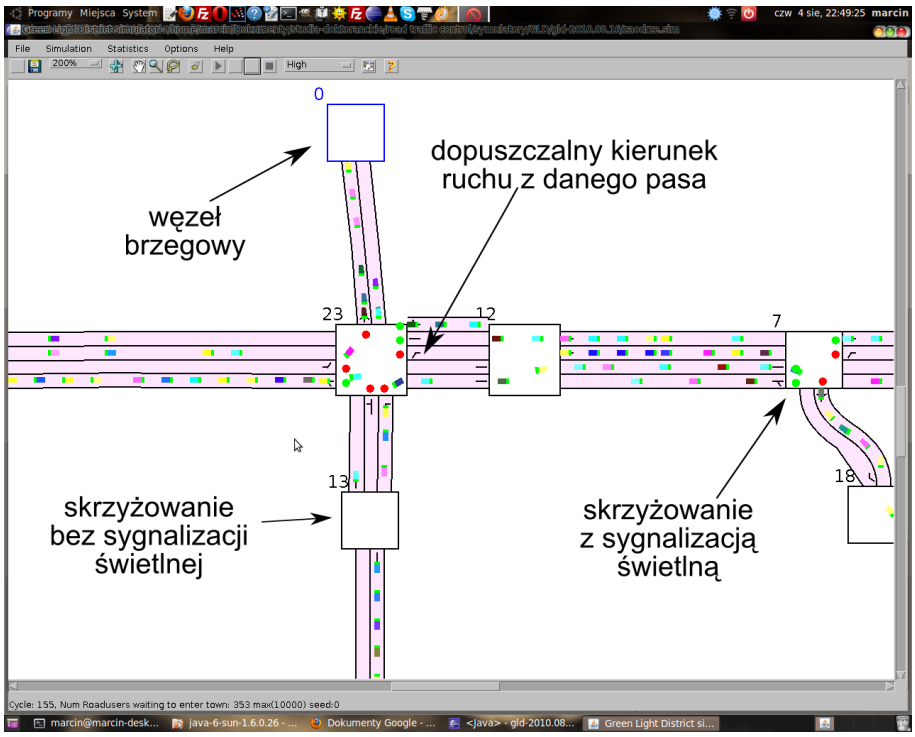
GLD jest systemem multiagentowym, w którym są dwa rodzaje agentów: pojazdy i skrzyżowania. Zgodnie z założeniami systemów multiagentowych zarówno pojazdy jak i skrzyżowania są niezależne a ich stany są aktualizowane w każdym kroku czasowym.

7.1.2. Drogi, węzły i skrzyżowania

Infrastruktura składa się z dróg i węzłów. Droga łączy dwa węzły i może mieć kilka pasów w każdym kierunku (Rys. 16). Długość każdej drogi wyrażona jest w jednostkach odpowiadających komórkom automatu komórkowego. Jedna komórka (blok) odpowiada długości 2,37m. Wynika to z przeliczenia $\frac{4{,}74\,m}{2} = 2{,}37\,m$, gdzie 4,74m jest długością przeciętnego pojazdu osobowego [17], natomiast 2 wynika z zajętości dwóch komórek przez pojazd osobowy. Jednostka długości (punkt) drogi jest równoważna długości 0,237m.

Każdy pas ruchu ma określone dopuszczalne możliwości skrętu na skrzyżowaniu (Rys. 16). Rozróżniono trzy rodzaje węzłów:

- węzły brzegowe (edge node), przez które pojazdy wjeżdżają do sieci lub z niej wyjeżdżają, mają przypisane dla każdego kroku czasowego (cyklu symulacji) pewne progi generowania pojazdów przyjmujące wartości od 0 do 1, mogą być węzłem wjazdowym, wyjazdowym lub jednocześnie wjazdowym i wyjazdowym;
- skrzyżowania bez sygnalizacji, na których dopuszczalne są fazy bezkolizyjne, może służyć do rozdzielenia dwóch odcinków drogi o różnej ilości pasów (węzły nr 12 i 13 na Rys. 16);
- skrzyżowania z sygnalizacją świetlną sterowane za pomocą algorytmów opisanych w rozdziałach 7.1.4. i 8. , łączą co najmniej trzy drogi.



Rys. 16: Fragment przykładowej infrastruktury w programie GLD

7.1.3. Pojazdy

Jest kilka rodzajów pojazdów, określonych przez ich prędkość, długość i liczbę pasażerów. W przeprowadzonych badaniach motocykle miały długość 1 komórki, samochody osobowe długość 2, natomiast autobusy długość 4. Prędkość samochodów osobowych jak i autobusów wynosiła maksymalnie 2, co oznacza dwie komórki w jednym kroku czasowym (cyklu symulacji). W jednym kroku czasowym pojazd może przesunąć się o odległość określoną przez prędkość i otoczenie (tzn. pojazdy

poprzedzające mogą ograniczać odległość, jaką dany pojazd może przebyć) lub może pozostać w tym samym miejscu ze względu na stojący przed nim inny pojazd lub sygnalizację świetlną, która w tym czasie nie zezwala na ruch.

W trakcie przejazdu pojazdu przez skrzyżowanie, polityka jazdy [3] określa pas ruchu, którym pojazd ma się poruszać za skrzyżowaniem. Domyślną polityką jazdy jest algorytm najkrótszej ścieżki polegający na tym, że ścieżka z danego skrzyżowania do punktu docelowego wybierana jest z istniejącej listy ścieżek [3]. W wersji zmodyfikowanej tej polityki, przed podjęciem decyzji wyboru kolejnego pasa ruchu, pod uwagę brana jest długość kolejek na pasach, które należą do najkrótszych ścieżek. Ponadto zaimplementowane są również polityki: agresywna i co-learning [155]. Pojazd będący na danej drodze wielopasmowej nie może zmienić pasa na inny.

Stan każdego pojazdu został opisany za pomocą czterech parametrów:

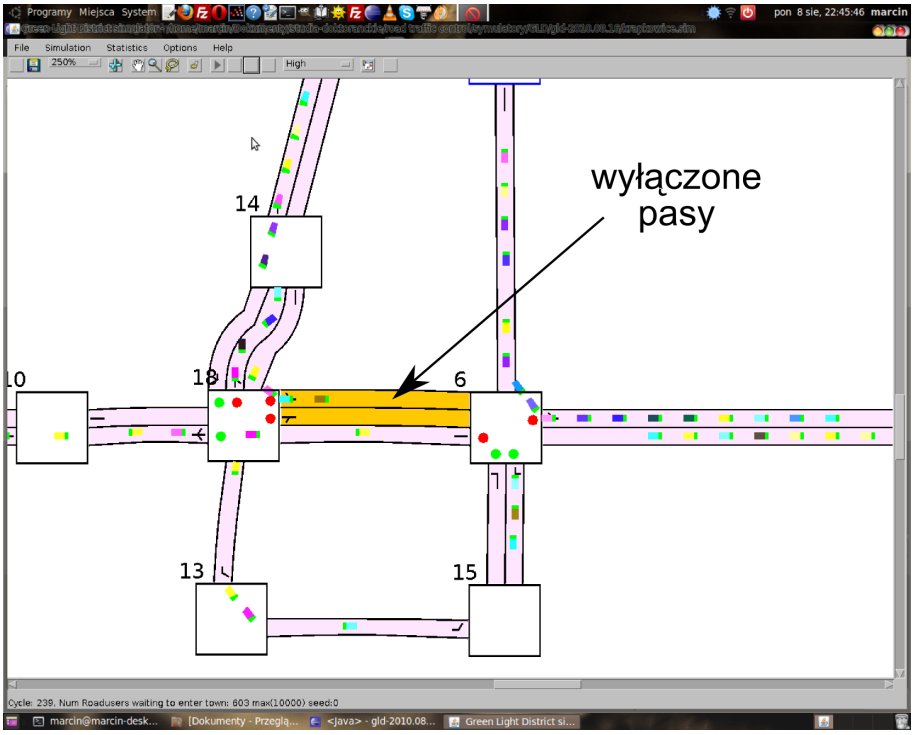
- numeru węzła (skrzyżowania), przed którym się znajduje *nr,*
- kierunku ruchu na skrzyżowaniu *kier,*
- miejsca zajmowanego w kolejce przed skrzyżowaniem *ms,*
- końcowego węzła wyjazdowego z sieci *wyj*.

W rzeczywistości wartości tych parametrów można osiągnąć przez stosowanie wideodetektorów lub pętli indykcyjnych. Zaawansowane systemy sterowania ruchem drogowym umożliwiają odbieranie od pojazdów danych dotyczących miejsca docelowego podróży [122,124,156,157].

7.1.4. Wypadki

W aplikacji przewidziano możliwość wystąpienia zdarzeń wyjątkowych (wypadków, remontów na drodze itp.) w trakcie trwania symulacji. Funkcjonalność ta zrealizowana jest w ten sposób, że pas ruchu, na którym wystąpiło zdarzenie wyjątkowe, blokowany jest przed wjazdem pojazdów. Zablokowany pas oznaczany jest żółtym kolorem (Rys. 17).

Wybór miejsca wystąpienia zdarzenia wyjątkowego następuje losowo. Pojawienie się zdarzenie jak i jego długość trwania następuje losowo z pewnymi prawdopodobieństwami. W trakcie blokady pojazdy znajdujące się na zablokowanym pasie mogą kontynuować podróż w stronę skrzyżowania, do którego zdążają. Natomiast pojazdy kierujące się na ten pas, muszą oczekiwać na jego odblokowanie.



Rys. 17: Oznaczenie zdarzenia wyjątkowego w aplikacji GLD

7.1.5. Sterowniki ruchu na skrzyżowaniach

Każde skrzyżowanie z sygnalizacją świetlną posiada pewną liczbę kolizyjnych lub bezkolizyjnych faz ruchu. W każdym kroku czasowym (cyklu symulacji zwanym dalej cyklem) sterownik sygnalizacji podejmuje decyzję dotyczącą wyboru jednej z możliwych faz ruchu dla skrzyżowania. Sterownik może użyć do tego celu informacji o liczbie pojazdów oczekujących na poszczególnych pasach przed skrzyżowaniem, o ich miejscu docelowym podróży, czasie oczekiwania na światło zielone (określony w cyklach aplikacji) oraz informacji o stanach innych sygnalizacji w badanej sieci. Sterowniki zatem mogą się między sobą wymieniać informacjami dotyczącymi liczby pasażerów, zapełnieniem danych odcinków itp. Wymiana informacji między sterownikami umożliwia określenie stanu sieci, w którym globalny czas oczekiwania przed skrzyżowaniami będzie najkrótszy.

W aplikacji zostało zaimplementowanych kilkanaście różnych algorytmów sterowania [3,155]:

- algorytm losowego wyboru fazy ruchu [155],
- algorytm przydzielenia światła zielonego dla pasów o największej liczbie pojazdów (Longest Queue Length) [155],

- algorytmy genetyczne,
- algorytmy uczenia ze wzmocnieniem (reinforcement learning) [158],
- algorytmy wykorzystujące sieci neuronowe itp.

Wybrane algorytmy do badań zostaną dokładniej omówione w rozdziale 8.

Aplikacja umożliwia implementowanie dodatkowych, własnych algorytmów sterowania, co wykorzystano dla celów niniejszej pracy.

7.1.6. Statystyki

W trakcie trwania symulacji obliczane i gromadzone są dane, które pomagają przy ocenie efektywności funkcjonowania badanych algorytmów. Gromadzone dane można podzielić na dwie grupy:

- lokalne – obliczane i zbierane dla każdego skrzyżowania indywidualnie,
- globalne – obliczane jako wynik pracy sterowników wszystkich skrzyżowań łącznie i zbierane.

Dla skrzyżowań z sygnalizacją świetlną obliczanymi parametrami efektywności są średni czas oczekiwania przed skrzyżowaniem oraz liczba pojazdów, które przejechały przez dane skrzyżowanie. Natomiast dla węzłów brzegowych obliczanymi parametrami efektywności są:

- liczba pojazdów, które przejechały przez dany węzeł,
- aktualna długość kolejki przed węzłem,
- średni czas podróży pojazdów przejeżdżających przez dany węzeł.

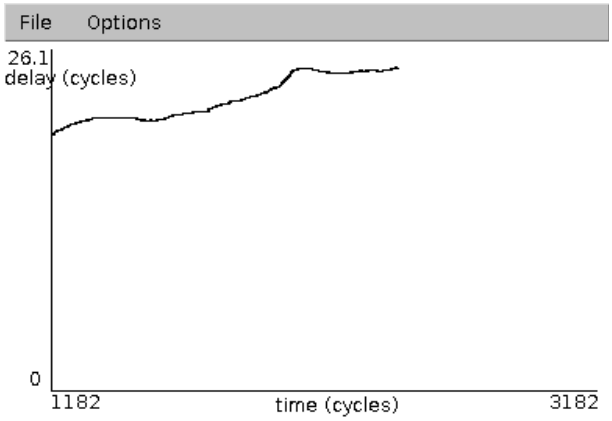
Do parametrów efektywności obliczanych globalnie należą [155]:

- całkowita długość kolejki pojazdów,
- średni czas oczekiwania przed skrzyżowaniami (ŚCOS),
- średni czas oczekiwania podczas podróży,
- całkowita liczba pojazdów, które przejechały przez sieć
- łączna liczba wypadków i zdarzeń wyjątkowych na drogach.

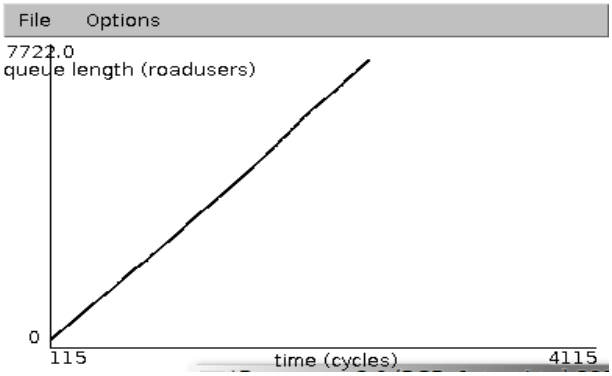
Średni czas oczekiwania pojazdu (ŚCOS) jest to średni czas pojazdu, jaki musi odczekać przed każdym skrzyżowaniem. Im większe zatłoczenie w sieci, tym większą wartość przyjmuje ten parametr [159].

Całkowita długość kolejki jest to liczba pojazdów oczekujących przed węzłami brzegowymi na wjazd do sieci. Pojazdy oczekują przed węzłami brzegowymi w przypadku, gdy pasy łączące dany węzeł brzegowy z innymi węzłami są pełne. Zwiększanie się tego parametru świadczy o tworzeniu się w sieci zatorów [159]. W pracy skupiono uwagę na liczbie pojazdów osiągających cel podróży w miejsce badania całkowitej długości kolejki pojazdów wjeżdżających do sieci.

W trakcie trwania symulacji gromadzone dane mogą być zobrazowane na wykresach. Na Rys. 18 i 19 przedstawiono wykresy średniego czasu oczekiwania i całkowitej długości kolejki w trakcie trwania symulacji.



Rys. 18: Wykres chwilowego średniego czasu oczekiwania w trakcie trwania symulacji



Rys. 19: Wykres całkowitej długości kolejki w trakcie trwania symulacji

Symulator GLD oblicza niektóre globalne parametry tylko dla pojazdów, które dotarły do celu. Przykładem jest parametr średni czas oczekiwania podczas podróży. Wiąże się to z tym, że w przypadku stałego zablokowania pewnych części sieci, czas podróży jak i opóźnienia pojazdów, które utkwiły w tych korkach nie są wliczane do statystyk. W efekcie algorytm, który nie radził sobie z rozładowaniem zatorów może osiągnąć lepsze wyniki statystyczne niż algorytm rozładowujący miejsca zakorkowane.

7.2. Wprowadzone Modyfikacje Do Aplikacji GLD

7.2.1. Generowanie ruchu o zadanym rozkładzie

Aplikację GLD rozszerzono o możliwość generowania ruchu o zadanym rozkładzie. W tym celu stworzono w projekcie aplikacji katalog gld.tg, który zawiera plik klasy bazowej generatorów `TrafficGenerator`, plik klasy zarządzającej generatorami `TGFactory` oraz pliki klas generatorów:

- `BirnbaumSaundersTG` – generator ruchu o rozkładzie Birnbauma-Saundersa,
- `Burr4PTG` – generator ruchu o rozkładzie Burra XII z czterema parametrami,
- `ErlangTG` – generator ruchu o rozkładzie Erlanga,
- `GammaTG` – generator ruchu o rozkładzie Gamma,
- `InverseGaussian3PTG` – generator ruchu o rozkładzie odwrotnym Gaussa z trzema parametrami,
- `Loglogistic3PTG` – generator ruchu o rozkładzie logarytmicznie logistycznym z trzema parametrami,
- `LognormalTG` – generator ruchu o rozkładzie logarytmicznie normalnym,
- `NormalTG` – generator ruchu o rozkładzie normalnym,
- `Pearson5TG` – generator ruchu o rozkładzie Pearson 5,
- `Pearson6TG` – generator ruchu o rozkładzie Pearson 6
- `WeibullTG` – generator ruchu o rozkładzie Weibull.

Klasy generatorów ruchu utworzono w oparciu o bibliotekę SSJ (Stochastic Simulation in Java) [160] w wersji 2.4.

Użytkownik ma możliwość wyboru generatora ruchu oraz ustawienia parametrów wybranego generatora z poziomu menu głównego aplikacji GLD. Wybrany generator ruchu obowiązuje dla wszystkich węzłów wjazdowych do badanej sieci. Można również ustawiać generatory indywidualnie dla każdego wjazdu do sieci.

Wygenerowana liczba jest odpowiednikiem odstępu czasu między ostatnio wygenerowanym pojazdem a pojazdem aktualnie wjeżdżającym do sieci. W jednym cyklu aplikacji do sieci w danym węźle wjeżdżają pojazdy, których suma odstępów

czasowych nie przekracza liczby równej długości najkrótszego przedziału sygnalizacji. W rzeczywistym programie sterowania ruchem na badanych skrzyżowaniach miasta Opola wartość długości tego czasu wynosiła 5 [s] [161].

7.2.2. Rozszerzenie biblioteki SSJ

Bibliotekę SSJ rozbudowano o trzy rozkłady: Burra XII z czterema parametrami (Burr 4P), odwrotny rozkład Gaussa z trzema parametrami (Inverse Gaussian 3P) oraz logarytmicznie logistyczny z trzema parametrami (Log-Logistic 3P). W tym celu stworzono cześć klas `Burr4PDist`, `Burr4PGen`, `InverseGaussian3PDist`, `InverseGaussian3PGen`, `Loglogistic3PDist` i `Loglogistic3PGen`. W klasach `Burr4PDist`, `InverseGaussian3PDist` i `Loglogistic3PDist` zaimplementowano m. in. następujące metody:

- `barF(double x)` – wartość dopełnienia dystrybuanty `1-cdf(double x)`,
- `cdf(double x)` – wartość dystrybuanty w punkcie x,
- `density(double x)` – wartość gęstości prawdopodobieństwa w punkcie x,
- `getMean()` – wartość oczekiwana rozkładu,
- `getVariance()` – wariancja rozkładu,
- `inverseF()` – odwrotność funkcji `cdf(double x)`.

Do obliczenia wartości oczekiwanej i wariancji rozkładu Burra XII z czterema parametrami (α, β, γ, k) opisanych wzorami (8) i (9) wykorzystano aproksymację funkcji $\Gamma(x)$ opisanej wzorem (10) zaproponowaną przez Corneliusa Lanczosa [162].

$$E(x) = \gamma + \beta \cdot \frac{\Gamma\left(1+\frac{1}{\alpha}\right)\cdot\Gamma\left(x-\frac{1}{\alpha}\right)}{\Gamma(x)} \tag{8}$$

$$V(x) = \beta^2 \cdot \frac{\Gamma\left(1+\frac{2}{\alpha}\right)\cdot\Gamma\left(x-\frac{2}{\alpha}\right)}{\Gamma(x)} - \left(\frac{\Gamma\left(1+\frac{1}{\alpha}\right)\cdot\Gamma\left(x-\frac{1}{\alpha}\right)}{\Gamma(x)}\right)^2 \tag{9}$$

gdzie:

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t}\, dt \tag{10}$$

Natomiast główną metodą w klasach `Burr4PGen`, `InverseGaussian3PGen` i `Loglogistic3PGen` jest `nextDouble()`.

W metodzie tej wykorzystano metodę `inverseF()` klasy `Burr4PDist` dla generatora liczb z rozkładu Burra XII (4P). Tym samym sposobem zaimplementowano odpowiednik tej metody z klasy `InverseGaussian3PDist` i `Loglogistic3PDist` dla generatorów liczb według rozkładu odwrotnego Gaussa z trzema parametrami i logarytmicznie logistycznego z trzema parametrami.

7.2.3. Zapisywanie generowanych statystyk

Generowane statystyki w trakcie wykonania serii symulacji zapisywane są do plików. W plikach tych rozszerzono nagłówek o informacje dotyczące użytego algorytmu sterowania, generatora ruchu jak również daty rozpoczęcia symulacji.

Wprowadzono również katalogowanie generowanych statystyk będących wynikiem przeprowadzania serii symulacji poprzez zapisywanie statystyk w trakcie symulacji do osobnych katalogów. Ułatwia to analizowanie wyników wykonanej serii symulacji dla dziesięciu ziaren generatora liczb losowych, której owocem jest dziesięć plików z danymi dla każdej z siedmiu statystyk. Wyniki każdej symulacji zapisywane są w osobnym katalogu, którego nazwa zawiera parametry symulacji takie jak:

- nazwa infrastruktury,
- nazwa algorytmu sterowania,
- datę rozpoczęcia symulacji z dokładnością do minut itp.

Wewnątrz tego katalogu znajdują się następujące podkatalogi, które segregują poszczególne statystyki:

- average_junction_waiting_time,
- average_trip_waiting_time,
- execution_time,
- number_of_accidents,
- total_arrived_roadusers,
- total_number_of_removed_cars,
- total_waiting_queue_length.

7.2.4. Pomiar czasu wykonania algorytmu sterowania

Na potrzeby tej pracy aplikacje GLD rozszerzono o statystykę czasu wykonania algorytmu sterowania. Wykorzystano w tym celu metodę `System.nanoTime()`, która zwraca czas zegara systemowego rzędu nanosekund. Czas wykonania algorytmu obliczono jako różnicę czasu po wykonaniu z czasem przed wykonaniem algorytmu sterowania. Pomiar czasu wykonania dokonywany jest w metodzie `switchSign()` obiektu typu `SignController` oraz w metodzie `moveLane()` obiektu typu `SimModel`. Przykładowy fragment kodu przedstawiono poniżej.

```
public void switchSigns()
{
   long startTime = System.nanoTime();
   TLDecision[][] decisions = tlcontroller.decideTLs();
   executionTime = System.nanoTime() - startTime;
   …
}
```

Dodatkowo opracowano klasę `ExecutionTimeTrackingView` bazującą na klasie `ExtendedTrackingView`. W klasie tej zaimplementowano mechanizmy zapisujące do plików otrzymane czasy wykonania jak również generowanie wykresów w trakcie wykonania symulacji.

7.2.5. Niezmienne progi generowania pojazdów

Każdy węzeł brzegowy, którym wjeżdżają pojazdy do badanej sieci drogowej ma trzy progi generowania pojazdów (`freq`), jeden dla każdego rodzaju pojazdów: osobowych (`CAR`), autobusów i ciężarowych (`BUS`), motocykli i rowerów (`BICYCLE`). Istnieje możliwość ustawienia tych parametrów. W aplikacji jednakże parametry te automatycznie były zmieniane podczas wyłączania pasa ruchu z powodu wydarzenia wyjątkowego lub podczas włączenia pasa. Wywoływana była wówczas metoda `addFrequencies()` obiektu typu `Validate`, która między innymi zamieniała progi pojazdów osobowych na wartość 0,25 a ciężarowych na 0,05. Celem uniknięcia automatycznych zmian dokonano modyfikacji linii kodu metody `addFrequencies()` obiektu typu `Validate`

```
freq = types[j] == RoaduserFactory.BUS ? 0.05f : 0.25f;
```

Powyższą linię kodu zamieniono na

```
freq = (freq >= 0 ? freq : 0);
```

8. PRZEGLĄD UŻYTYCH ALGORYTMÓW STEROWANIA

8.1. Most Cars

Algorytm Most Cars [4,155,163,164] przydziela dla każdego wjazdowego pasa ruchu *j* skrzyżowania *i* z sygnalizacją świetlną, na którym znajduje się co najmniej jeden pojazd, zysk q_{ij}=1 lub 0 dla pasów nie zajętych przez żaden pojazd, co opisuje wzór (11).

$$q_{ij} = \begin{cases} 0, \textit{jeżeli } lp_{ij} = 0 \\ 1, \textit{jeżeli } lp_{ij} > 0 \end{cases} \qquad (11)$$
$$i = 1, \ldots, n$$
$$j = 1, \ldots, lps_i$$

gdzie:

lp_{ij} – liczba pojazdów na danym pasie wjazdowym *j* przed skrzyżowaniem *i*,

lps_i – liczba pasów wjazdowych danego skrzyżowania *i*,

n – liczba skrzyżowań z sygnalizacją świetlną objętych sterowaniem,

q_{ij} – zysk pasa ruchu *j* skrzyżowania *i*, gdzie *q* tablica zysków dla całej sieci objętej sterowaniem.

Uzyskane w ten sposób zyski dla każdego pasa w sieci objętego sterowaniem przekazywane są do następnej warstwy sterowania, w której dla każdego skrzyżowania obliczany jest zbiór sum zysków. Zbiór stanowią sumy zysków dla każdej fazy ruchu z osobna. Obliczanie poszczególnych sum odbywa się w ten sposób, że sumowane są wartości zysków dla pasów wjazdowych, dla których w danej fazie ruchu przewidziany jest sygnał zielony. Ostatnim krokiem jest wybór fazy ruchu z największym zyskiem według zależności (12), co stanowi wyznaczenie decyzji sterującej.

$$Z_{ik}^{opt} = \arg\max_{Z_{ik}} \sum_{l \in Z_{ik}} q_{il} \qquad (12)$$
$$i = 1, \ldots, n$$
$$k = 1, \ldots, lf_i$$
$$l \in Z_{ik}$$

gdzie:

i – indeks skrzyżowania,

k – indeks fazy ruchu danego skrzyżowania,

l – indeks pasa ruchu, należącego do zbioru pasów Z_{ik},

lf_i – liczba możliwych faz ruchu skrzyżowania i,

n – liczba skrzyżowań z sygnalizacją świetlną objętych sterowaniem,

q_{il} – zysk pasa ruchu l skrzyżowania i,

Z_{ik} – zbiór pasów ruchu skrzyżowania i, które w fazie ruchu k mają sygnał zielony.

Procedury tej warstwy wykonywane są dla każdego z niżej opisanych i badanych algorytmów.

Atutem tego algorytmu jest prostota oraz zadowalające rezultaty sterowania ruchem z jego użyciem, co potwierdzają badania symulacyjne (Rozdz. 10.).

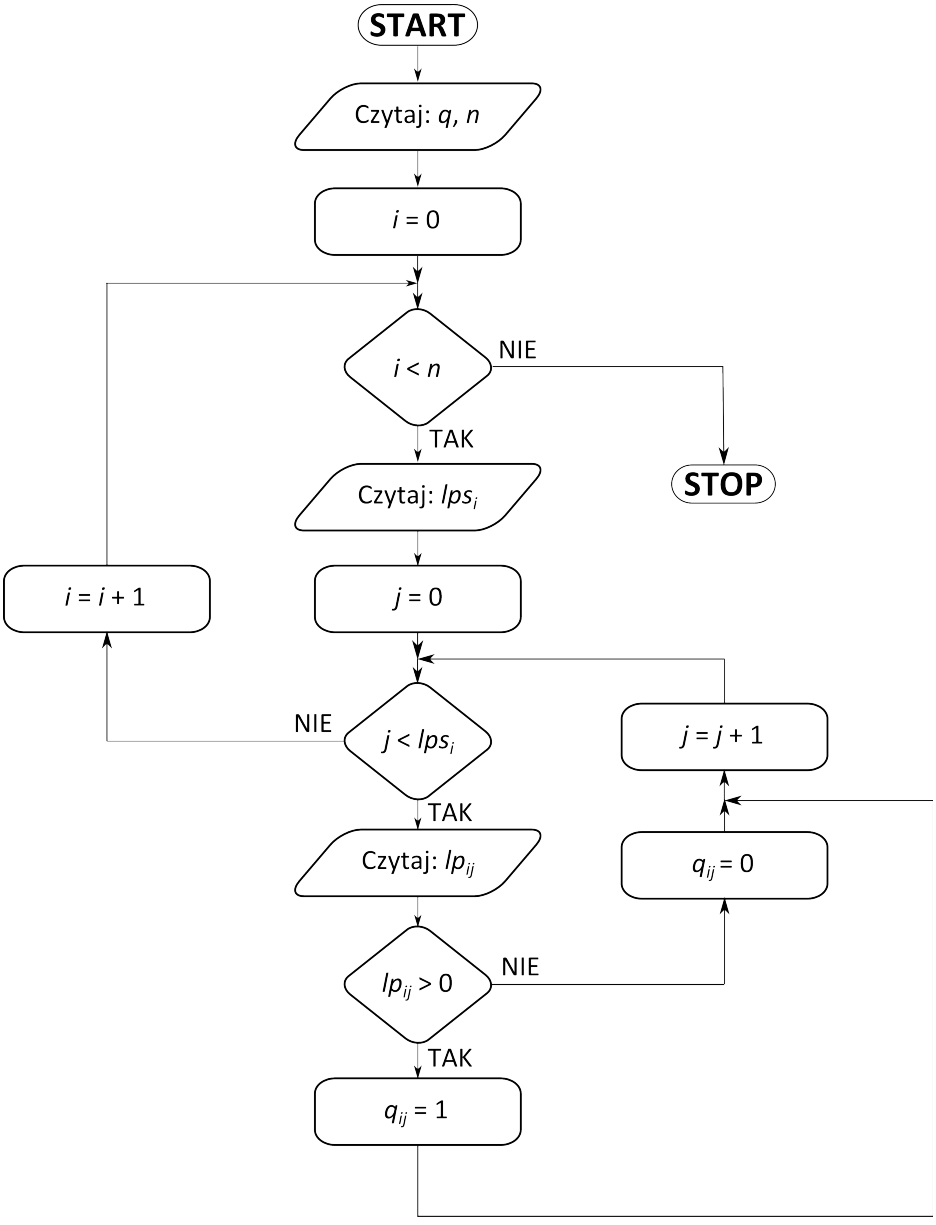
Sterowniki na skrzyżowaniach działające zgodnie z algorytmem Most Cars nie wymieniają się informacjami, co prowadzi do tego, że wybór najlepszej fazy ruchu dla skrzyżowania nie oznacza najlepszej fazy ruchu dla poszczególnych pojazdów. Pojazd, który zostanie zatrzymany sygnałem czerwonym może nie trafić na sygnał zielony na kolejnym skrzyżowaniu.

Algorytm nie bierze pod uwagę zapełnienia pasa na całej jego długości między skrzyżowaniami. Efektem tego może być wybór fazy ruchu, która nie uwzględnia sygnału zielonego dla pasa zajętego na całej długości. Pociąga to za sobą blokadę dla pojazdów mających sygnał zielony na skrzyżowaniu sąsiednim i chcących wjechać na pas zajęty na całej długości. Problem ten rozwiązano w algorytmie autorskim.

Algorytm Most Cars przydziela zysk równy 1 dla pasów zajętych przez jeden pojazd jak również przez więcej pojazdów. Własność ta jest w pewnych warunkach atutem a w innych wadą tego algorytmu. Atutem jest na skrzyżowaniach, których jedne wjazdy są częściej uczęszczane od innych. W przypadku pojawienia się pojazdów na takich wjazdach otrzymują one równe szanse na sygnał zielony. W wyniku takiego postępowania może jednak zostać wybrana faza ruchu, w której sygnał zielony zostanie przydzielony dla pasa ruchu z jednym pojazdem, natomiast kolejka pojazdów na innym pasie otrzyma sygnał czerwony. W przypadku przejechania tego potoku złożonego z jednego pojazdu i w tym samym czasie pojawienia się ponownie jednego pojazdu ta sama faza ruchu może zostać powtórzona. Tę wadę algorytmu wyeliminowano

w algorytmie autorskim przez wykorzystanie dopełnienia do 1 względnego zapełnienia pasa wyjazdowego, na który pojazdy przejadą z pasa *j*, tj. stosunku liczby bloków zajętych przez pojazdy do liczby bloków długości tego pasa ruchu.

Na Rys. 20 przedstawiono schemat blokowy algorytmu Most Cars. Algorytm przedstawiony na Rys. 20 stanowi podstawową część autorskiego algorytmu sterowania acykliczną sygnalizacją świetlną. Z algorytmu tego korzysta algorytm maksymalizacji według zależności (12).



Rys. 20: Schemat blokowy algorytmu Most Cars

8.2. Local Hill-Climbing

Algorytm Local Hill-Climbing (wspinaczkowy, najszybszego wzrostu) [4,114,155,163,164,165,166]. W każdym cyklu algorytm wylicza globalne sumy wartości zysków dla całej sieci. Wybrana zostaje konfiguracja faz ruchu skrzyżowań w sieci dającą największą sumę. Dla każdego skrzyżowania w sieci realizowana jest procedura według następujących kroków:

Krok 1: Wybierz spośród dostępnych dla tego skrzyżowania faz ruchu jedną fazę.

Krok 2: Dla danej fazy przydziel wartości zysków dla każdego pasa wjazdowego każdego skrzyżowania w sieci w następujący sposób: jeśli aktualnie dla danego pasa jest sygnał zielony i pierwszy stojący przed skrzyżowaniem lub zbliżający się do skrzyżowania pojazd może przejechać przez skrzyżowanie (odcinek za skrzyżowaniem nie jest zapełniony na całej swej długości), wówczas przypisz dla niego wartość zysku 1. W przeciwnym razie 0.

Krok 3: Oblicz dla tej zmiany globalny zysk.

Krok 4: Jeśli globalny zysk wyliczany jest po raz pierwszy w tym cyklu, zapamiętaj jako maksymalny, w przeciwnym razie jeśli dla tej aktualnej konfiguracji jest on większy od maksymalnego globalnego zysku, zapamiętaj jego wartość jak i aktualną fazę jako optymalne.

Krok 5: Jeśli nie sprawdzono wszystkich dostępnych faz dla danego skrzyżowania, przejdź do kroku pierwszego, w przeciwnym razie wybierz kolejne skrzyżowanie.

Algorytm Local Hill-Climbing wymaga dużo czasu na obliczenia ze względu na rozpatrywanie wszystkich faz ruchu dla każdego skrzyżowania.

8.3. TC-1 Bucket 2.0

TC-1 Bucket 2.0 (Traffic Controller 1 Bucket 2.0) jest to metoda *Q*-learning algorytmu uczenia się ze wzmocnieniem [3,4,84,158,163,164,167], w którym celem optymalizacji ustawień sygnalizacji w sieci obliczana jest suma wszystkich spodziewanych czasów oczekiwania wszystkich pojazdów biorąc pod uwagę możliwe fazy ruchu. Wybierane jest ustawienie faz ruchu w sieci, dla którego całkowity czas

oczekiwania na skrzyżowaniu jest najmniejszy. W tym celu uczona jest funkcja wartości *Q*, która estymuje dla każdego pojazdu długość czasu oczekiwania aż do dotarcia do miejsca docelowego, biorąc pod uwagę sygnał zielony i czerwony.

Korzystając z wektora parametrów opisu pojazdów $[nr, kier, ms, wyj]$, określono funkcję estymującą całkowity czas oczekiwania $Q([nr, kier, m, wyj], Sg)$, gdzie *Sg* – sygnał zielony lub czerwony, oraz funkcję estymującą średni czas oczekiwania $V([nr, kier, ms, wyj])$. Dla każdego skrzyżowania podejmowana jest decyzja na podstawie następującej funkcji:

$$Z_{ik}^{opt} = \arg\max_{Z_{ik}} \sum_{l \in Z_{ik}} \sum_{(nr, kier, ms, wyj) \in kol_l} \{Q([nr, kier, ms, wyj], cz) + \\ -Q([nr, kier, ms, wyj], z)\} \\ i = 1, \ldots, n \\ k = 1, \ldots, lf_i \\ l \in Z_{ik} \tag{13}$$

gdzie:

cz – sygnał czerwony,

z – sygnał zielony,

i – indeks skrzyżowania,

k – indeks fazy ruchu danego skrzyżowania,

kol_l – kolejka pojazdów na pasie l,

l – indeks pasa ruchu, należącego do zbioru pasów Z_{ik},

lf_i – liczba możliwych faz ruchu skrzyżowania *i*,

n – liczba skrzyżowań z sygnalizacją świetlną objętych sterowaniem,

Z_{ik} – zbiór pasów ruchu skrzyżowania *i*, które w fazie ruchu *k* mają sygnał zielony.

Funkcja *Q* obliczana jest w następujący sposób:

$$Q([nr, kier, ms, wyj], Sg) = \sum_{(nr', kier', ms')} P([nr, kier, ms, wyj], Sg, [nr', kier', ms']) \\ (R([nr, kier, ms], Sg, [nr', kier', ms']) + \gamma V([nr', kier', ms', wyj'])) \tag{14}$$

Występujące we wzorze (14) parametry *nr', kier', ms', wyj'* oznaczają parametry pojazdu w kolejnym stanie po wykonaniu akcji *Sg*. Natomiast *P* jest to prawdopodobieństwo przejścia ze stanu aktualnego $[nr, kier, ms, wyj]$, w jakim pojazd się znajduje do następnego stanu $[nr', kier', ms', wyj']$, przy wystąpieniu na skrzyżowaniu *nr* sygnału *Sg*. Realizacja funkcji prawdopodobieństwa *Pr* odbywa się

za pomocą tablicy przeglądowej. Wbrew pozorom tablica nie ma ogromnych rozmiarów z tego względu, że pojazd stojący w kolejce może się przesunąć do przodu lub nie przed tym samym skrzyżowaniem $(nr, kier)$, przy czym punkt docelowy *wyj* pozostaje przez całą podróż niezmienny. Kolejną funkcją występującą we wzorze (14) jest funkcja nagrody $R([nr, kier, ms], Sg, [nr', kier', ms'])$. Funkcja *R* przyjmuje wartość 1, jeżeli pojazd w trakcie cyklu stoi w miejscu ze względu na wystąpienie sygnału czerwonego lub zablokowania pasa, na który miałby wjechać po przejechaniu skrzyżowania. Natomiast w przypadku, gdy pojazd może poruszać się funkcja nagrody przyjmuje wartość 0. Parametr γ jest to współczynnik dyskontowy przyjmujący wartości $(0<\gamma<1)$, który redukuje wpływ funkcji *V* z przyszłego kroku.

Funkcję *V* opisano w następujący sposób:

$$V([nr, kier, ms, wyj]) = \sum_{Sg} P(Sg|[nr, kier, ms, wyj])Q([nr, kier, ms, wyj], Sg) \quad (15)$$

Występujące we wzorze (15) prawdopodobieństwo $P(Sg|[nr, kier, ms, wyj])$ oznacza prawdopodobieństwo wystąpienia sygnału czerwonego lub zielonego dla pojazdu opisanego wektorem $[nr, kier, ms, wyj]$.

W ten sposób jeśli prawdopodobieństwo oczekiwania jest duże (dla którego nagroda lub koszt jest 1), wówczas wartość *V* oraz wartości *Q* są duże. Funkcje *Q* i *V* inicjowane są wartościami 0,0.

Po każdym kroku symulacji obliczane jest prawdopodobieństwo przejścia stanów a później obliczane są wartości funkcji *V* i *Q* przez iterację za pomocą programowania dynamicznego z zastosowaniem w czasie rzeczywistym (real time dynamic programming) [168], które ma ustaloną liczbę iteracji. W badaniach wykorzystano tylko jedną iterację. Odpowiada to aktualizacji wartości *Q* i *V*, gdzie brany jest pod uwagę ruch pojazdów jedynie w trakcie trwania aktualnego kroku czasowego. Jest to szybki sposób częściowego przeliczenia pożądanych funkcji.

Algorytm TC-1 używa jedynie lokalnych informacji i oblicza czas oczekiwania pojazdów bez względu na sytuację na innych skrzyżowaniach.

Do algorytmu dodano mechanizm Bucket (wiaderko) [3,164], którego celem jest optymalizacja przepływu ruchu w sieciach o bardzo dużej gęstości. Mechanizm polega na tym, że sygnalizacje, z których ruch jest niemożliwy ze względu na zatłoczony pas docelowy za skrzyżowaniem, przekazuje część swojego zsumowanego zysku celem

stymulacji ruchu. Wartość zysku każdego pasa ruchu jest dodawana do „wiaderka". Ta wartość jest dopiero wykorzystywana do określenia optymalnej fazy ruchu. W przypadku, gdy jest aktywny sygnał zielony, ale pojazd nie może poruszyć się ze względu na zapełnienie pasa docelowego (za skrzyżowaniem), wówczas część zawartości wiaderka jest przelewana do wiaderka pasa docelowego. W czasie, gdy pojazd przejedzie przez skrzyżowanie zawartość wiaderka jest zmniejszana o 1/*n*, gdzie *n* oznacza liczbę pojazdów na danym pasie. Mechanizm ten umożliwia rozładowanie zatorów, w których bierze udział niewielka liczba pojazdów.

W każdym cyklu jest generowana liczba losowa w przedziale (0,1). Następnie ta liczba jest porównywana z wartością progową (próg losowości). Jeśli przekracza wartość progową zyski przypisywane są do pasów ruchu według algorytmu TC- 1 Bucket 2.0. W przeciwnym razie zyski przypisywane są losowo. Wartość progowa może wynosić np. 0,01.

8.4. RL SARSA 5

RL SARSA 5 jest to metoda SARSA (State-Action-Reward-State-Action – Stan-Akcja-Nagroda-Stan-Akcja) algorytmu uczenia się ze wzmocnieniem [4,84,163,164,167]. Różnica między metodą SARSA a metodą *Q*-learning opisaną w poprzednim rozdziale 8.3. polega na tym, że algorytm *Q*-learning nie ma z góry narzuconej polityki działania. W związku z tym sam musi znajdować akcję (sygnał zielony lub czerwony), która w danym stanie $[nr, kier, ms, wyj]$ da największą wartość funkcji *Q*. W tym celu obliczana jest wartość prawdopodobieństwa $P([nr, kier, ms, wyj], Sg, [nr', kier', ms'])$ oraz zgodnie ze wzorem (15) wartość funkcji $V([nr', kier', ms', wyj'])$. W metodzie SARSA po wybraniu akcji *Sg* w stanie $[nr, kier, ms, wyj]$ obserwowany jest stan $[nr', kier', ms', wyj]$ i wybierana jest na podstawie tej samej polityki akcja Sg'. Nie ma zatem potrzeby obliczania prawdopodobieństwa $P([nr, kier, ms, wyj], Sg, [nr', kier', ms'])$. Natomiast wartość funkcji $Q([nr', kier', ms', wyj'], Sg')$ odczytywana jest z tablicy wartości.

Każdorazowo na koniec każdego cyklu tablica wartości funkcji *Q* jest uaktualniana według następującej procedury:

$$Q(s, Sg) \leftarrow Q(s, Sg) + \alpha(r + \gamma Q(s', Sg') - Q(s, Sg)) \quad (16)$$

gdzie:

α - tempo uczenia się $0<\alpha<1$,

$s=[nr, kier, ms, wyj]$,

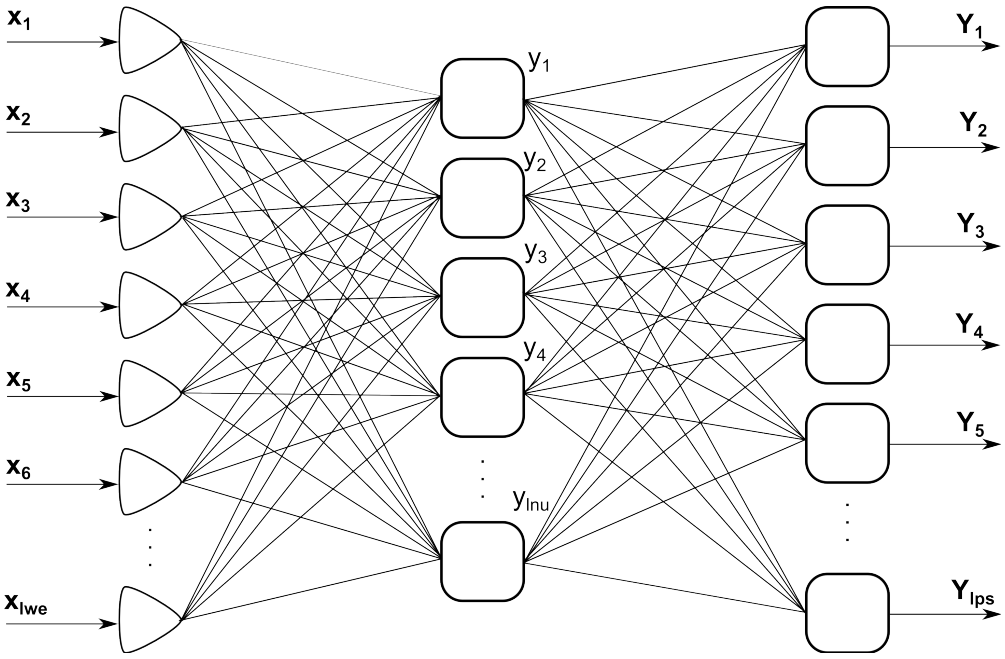
$s'=[nr', kier', ms', wyj']$,

$r=R([nr, kier, wyj], Sg, [nr', kier', wyj'])$.

W każdym cyklu jest generowana liczba losowa w przedziale (0,1). Następnie ta liczba jest porównywana z wartością progową (próg losowości) np. 0,01. Jeśli przekracza wartość progową zyski przypisywane są do pasów ruchu według algorytmu RL SARSA 5. W przeciwnym razie zyski przypisywane są losowo.

8.5. GenNeural

Algorytm GenNeural [120,155,164,169,170] stanowi nieliniowa, jednokierunkowa i dwuwarstwowa sieć neuronowa (Rys. 21) nadzorowana algorytmem genetycznym.



Rys. 21: Struktura sieci neuronowej użytej w algorytmie GenNeural

Każde skrzyżowanie jest wyposażone w sterownik oparty o sieć neuronową, której współczynniki wagowe jak i progi funkcji skalującej dopasowywane są algorytmem genetycznym. Liczba wejść *lwe* sieci określona jest według wzoru (17).

$$lwe=lps+\sum_{i\in Z_s} lpss_i \tag{17}$$

gdzie:

lps – liczba pasów wjazdowych danego skrzyżowania, dla którego tworzony jest sterownik,

$lpss_i$ – liczba pasów wjazdowych sąsiadującego skrzyżowania *i* z danym skrzyżowaniem, dla którego tworzony jest sterownik,

Z_s – zbiór indeksów skrzyżowań sąsiadujących z danym skrzyżowaniem.

Każde wejście x_k odpowiada jednemu pasowi wjazdowemu danego skrzyżowania lub skrzyżowania sąsiadującego z nim. Wartością wejściową jest iloraz liczby bloków (komórek) zajętych przez pojazdy do liczby bloków całego pasa (długości danego pasa).

Neuron *l* warstwy ukrytej wykonuje następującą operację:

$$y_l = \varphi\left(\sum_{k=1}^{lwe} \pi_{u_{kl}} x_k - \delta_{u_l}\right) \tag{18}$$

gdzie:

x_k – wartość na wejściu *k* sieci neuronowej,

$\pi_{u_{kl}}$ – wartość wagi wejścia *k* neuronu *l* warstwy ukrytej,

δ_{u_l} – wartość progowa na wyjściu neuronu *l* warstwy ukrytej,

φ – funkcja przejścia, której wartości określane są na podstawie zależności (19).

$$\varphi(e_l) = \begin{cases} 1 \, gdy \, e_l > 0 \\ 0 \, gdy \, e_l \leq 0 \end{cases} \tag{19}$$

gdzie:

$$e_l = \left(\sum_{k=1}^{lwe} \pi_{u_{kl}} x_k - \delta_{u_l}\right) \tag{20}$$

Istnieje możliwość określenia liczby neuronów warstwy ukrytej *lnu*. Natomiast warstwa wyjściowa składa się z neuronów w liczbie równej liczbie *lps* pasów wjazdowych z sygnalizacją świetlną danego skrzyżowania. Neurony warstwy wyjściowej wykonują podobną operację (21) jak neurony warstwy ukrytej zapisaną wzorem (18) oraz przetwarzają informacje według tej samej funkcji przejścia opisanej wzorem (19). Różnica polega na tym, że na wejścia neuronu wyjściowego podawane są wyjścia neuronów warstwy ukrytej (Rys. 21). Tak więc liczba wejść neuronu wyjściowego określona jest przez liczbę neuronów warstwy ukrytej *lnu*. Wartość na wyjściu sieci Y_j przypisanego do konkretnego pasa ruchu danego skrzyżowania jest wartością zysku dla

tego pasa, która w dalszym etapie zostanie użyta do określenia optymalnej fazy ruchu dla danego skrzyżowania.

$$Y_j = \varphi\left(\sum_{l=1}^{lnu} \pi_{wy_{jl}} y_l - \delta_{wy_j}\right) \tag{21}$$

Algorytm genetyczny nadzorujący powyższą sieć neuronową działa w oparciu o populację osobników, której liczebność jest uzależniona od liczby skrzyżowań z sygnalizacją świetlną w danej sieci drogowej. Z każdym skrzyżowaniem skojarzona jest grupa licząca 20 osobników. Każdy osobnik opisany jest przez pojedynczy chromosom. Chromosomy zbudowane są z wielu jednobajtowych genów, w których zapisane są informacje dotyczące wag i wartości progowych sieci neuronowej. Długość chromosomu *lg* jest uzależniona od liczby wjazdów danego skrzyżowania *lps*, liczby wjazdów skrzyżowań sąsiadujących $\sum_{i \in Z_s} lpss_i$ oraz liczby neuronów warstwy ukrytej *lnu* i można ją określić za pomocą wzoru (22):

$$lg = (lps + \sum_{i \in Z_s} lpss_i) \cdot lnu + lnu + lps \cdot lnu + lps \tag{22}$$

gdzie:

$(lps + \sum_{i \in Z_s} lpss_i) \cdot lnu$ – wyrażenie określa łączną liczbę współczynników wagowych neuronów warstwy ukrytej,

lnu – wyrażenie określa liczbę progów skalujących warstwy ukrytej,

$lps \cdot lnu$ – wyrażenie określa liczbę współczynników wagowych wszystkich neuronów warstwy wyjściowej,

lps – wyrażenie określa łączną liczbę progów skalujących warstwy wyjściowej.

Każdy osobnik z grupy działa aktywnie przez pewną określoną liczbę cykli np. 20. Po sprawdzeniu się wszystkich osobników danej grupy następuje ewolucja. W wyniku ewolucji powstaje nowa grupa osobników w tej samej liczbie. Owocem krzyżowania dwóch osobników jest para potomków, których chromosomy są połączeniem chromosomów rodziców. Decyzja dotycząca tego, od którego z rodziców potomek odziedziczy dany gen zależy od wartości wylosowanej liczby. Jeśli liczba przekroczy wartość progową, dziedziczony jest gen matki w przeciwnym razie ojca. Ewolucja następuje między osobnikami należącymi do jednej grupy. Następnie geny podawane

są mutacji. Losowy bit genu z pewnym prawdopodobieństwem jest negowany. W tym przypadku selekcja jest proporcjonalna. Istnieje możliwość selekcji rankingowej. Każdy osobnik ma przypisane przystosowanie *prz* według wzoru (23).

$$prz = \frac{lpp}{(lpp + lpo)} \tag{23}$$

gdzie:

lpo – liczba pojazdów oczekujących w trakcie życia aktualnych osobników,

lpp – liczba pojazdów poruszających się w trakcie życia aktualnych osobników.

Pierwsze pokolenie inicjowane jest liczbami losowymi. Po osiągnięciu pewnej określonej z góry liczby pokoleń (np. 100), kolejne pokolenie jest pierwszym pokoleniem inicjowanym losowo.

8.6. In-and-Outbound Lane Control

Autorski algorytm In-and-Outbound Lane Control [4,163,164] jest rozszerzeniem algorytmu Most Cars przedstawionego w rozdziale 8.1. Zauważono bowiem, że algorytm Most Cars daje zadowalające rezultaty sterowania ruchem oraz charakteryzuje się szybkością wykonania. Posiada jednak kilka wad, których można uniknąć po wprowadzeniu prostych modyfikacji.

W wersji pierwszej [4] algorytm autorski przypisywał zysk q_{ij}=1 dla każdego pasa ruchu *j* skrzyżowania *i*, na którym znajdował się co najmniej jeden pojazd, oraz zysk q_{ij}=2 dla pasa ruchu *j* skrzyżowania *i* zajętego przez pojazdy na całej jego długości, co wyraża zależność (24).

$$q_{ij} = \begin{cases} 0, & jeżeli\ lp_{ij} = 0 \\ 1, & jeżeli\ (lp_{ij} > 0) \wedge (lbp_{ij} < dp_{ij}) \\ 2, & jeżeli\ (lp_{ij} > 0) \wedge (lbp_{ij} = dp_{ij}) \end{cases} \quad \wedge - \text{koniunkcja} \quad i = 1, \ldots, n \quad j = 1, \ldots, lps_i \tag{24}$$

gdzie:

dp_{ij} – długość pasa *j* skrzyżowania *i*,

lbp_{ij} – liczba bloków pasa *j* skrzyżowania *i* zajętych przez pojazdy,

lp_{ij} – liczba pojazdów na danym pasie wjazdowym *j* przed skrzyżowaniem *i*.

lps_i – liczba pasów wjazdowych danego skrzyżowania *i*,

n – liczba skrzyżowań z sygnalizacją świetlną objętych sterowaniem,

q_{ij} – zysk pasa ruchu *j* skrzyżowania *i*, gdzie *q* jest tablicą zysków pasów całej sieci.

Naprawiono w ten sposób wadę algorytmu bazowego. Ta modyfikacja pozwoliła szybciej rozładowywać zatory powstające na odcinkach dróg o długości kilku pojazdów i będących kilkukrotnie krótszych od innych pasów. Zauważono jednak, że wjazdy częściej zajmowane na całej długości mogą być faworyzowane. Wadę tę usunięto w następnej wersji 1.1.

W wersji 1.1 [4,163,164] algorytmu zysk q_{ij}=1 pasa ruchu *j* skrzyżowania *i*, na którym czas oczekiwania bez ruchu tps_{ij} pojazdów na światło zielone wynosił co najmniej 3 cykle *(wtt*=3*),* jest mnożony przez współczynnik *f*=2. Wartości współczynników *f* jak i *wtt* mogą być dobierane w zależności od struktury sieci i warunków. Wartości tych współczynników w trakcie trwania okresu sterowania są takie same dla wszystkich pasów ruchu w sieci. Ponadto przez wartość współczynnika *f* mnożone są zyski pasów ruchu zajętych przez pojazdy na całej ich długości, co przedstawia zależność (25).

$$q_{ij}=\begin{cases} 0, & jeżeli\ (lp_{ij}=0)\wedge(rb>semafor) \\ 1, & jeżeli\ (lp_{ij}>0)\wedge(lbp_{ij}<dp_{ij})\wedge(tps_{ij}<wtt)\wedge(rb>semafor) \\ f, & jeżeli\ (lp_{ij}>0)\wedge\left\{(lbp_{ij}<dp_{ij})\ \dot{\vee}\ (tps_{ij}\geq wtt)\right\}\wedge(rb>semafor) \\ f\cdot f, & jeżeli\ (lp_{ij}>0)\wedge(lbp_{ij}=dp_{ij})\wedge(tps_{ij}\geq wtt)\wedge(rb>semafor) \\ rd(), & jeżeli\ (rb\leq semafor) \end{cases} \qquad (25)$$

$\wedge$ – koniunkcja, $\dot{\vee}$ – alternatywa rozłączna

$i=1,\ldots,n$

$j=1,\ldots,lps_i$

Autorski algorytm począwszy od wersji 1.1 przewiduje dla pewnego prawdopodobieństwa poniżej wartości progowej (progu losowości) $rb=0{,}02$ przypisywanie zysków w sposób losowy. Przypisana wartość funkcji rd() do zysku q_{ij} pasa *j* skrzyżowania *i* zawiera się w zbiorze [0,1] i generowana jest według rozkładu równomiernego. W każdym cyklu jest losowana liczba *semafor*, która jest porównywana z *rb.* Jeżeli $semafor>rb$, wówczas zyski przypisywane są zgodnie z algorytmem In-and-Outbound Lane Control. Mała wartość progu losowości ma na celu uniknięcie braku kontroli nad ruchem. W niektórych jednak sieciach o krótkich

odcinkach dróg między skrzyżowaniami, dużej liczbie skrzyżowań oraz dużym zatłoczeniu wprowadzenie niewielkiego wpływu losowego przydziału zysków pozwalało odblokowywać zatory [4].

W kolejnej wersji 1.2 algorytmu w miejsce wartości 1 dla pasa *j* skrzyżowania *i* zajętego przez co najmniej jeden pojazd przypisywana jest względna zajętość pasa tj. stosunek liczby bloków zajętych przez pojazdy lbp_{ij} do liczby bloków długości pasa $dp_{ij.}$ Zauważono, że lepsze rezultaty sterowania można osiągnąć, gdy dla pasa *j* skrzyżowania *i* zajętego przez pojazd lub pojazdy przypisywane jest dopełnienie do 1 ilorazu liczby zajętych bloków $lbp_{ij'}$ na pasie *j'* za skrzyżowaniem, na który pojazdy przejadą z pasa *j* do całkowitej liczby bloków $dp_{ij'}$ pasa *j'* za skrzyżowaniem, co wyraża wzór (26).

$$q_{ij}{}' = 1 - \frac{lbp_{ij'}}{dp_{ij'}} \tag{26}$$

W wersji 1.2 opisanej zależnością (27) utrzymano modyfikacje dotyczące współczynników wprowadzone w poprzedniej wersji. Zmianie uległa tylko wartość bazowa q_{ij}', która podlega mnożeniu przez współczynnik *f*. Usunięto w ten sposób mankament algorytmu Most Cars, który traktował wjazdy do skrzyżowania bez względu na rozmiar zapełnienia i nie uwzględniał pasów wyjazdowych.

$$q_{ij} = \begin{cases} 0, & jeżeli\ (lp_{ij} = 0) \wedge (rb > semafor) \\ 1 - \dfrac{lbp_{ij'}}{dp_{ij'}}, & jeżeli\ (lp_{ij} > 0) \wedge (lbp_{ij} < dp_{ij}) \wedge (tps_{ij} < wtt) \wedge (rb > semafor) \\ f \cdot \left(1 - \dfrac{lbp_{ij'}}{dp_{ij'}}\right), & jeżeli\ (lp_{ij} > 0) \wedge \left\{(lbp_{ij} < dp_{ij}) \dot{\vee} (tps_{ij} \geq wtt)\right\} \wedge (rb > semafor) \\ f \cdot f \cdot \left(1 - \dfrac{lbp_{ij'}}{dp_{ij'}}\right), & jeżeli\ (lp_{ij} > 0) \wedge (lbp_{ij} = dp_{ij}) \wedge (tps_{ij} \geq wtt) \wedge (rb > semafor) \\ rd(), & jeżeli\ (rb \leq semafor) \end{cases} \tag{27}$$

$\wedge$ – koniunkcja, $\dot{\vee}$ – alternatywa rozłączna

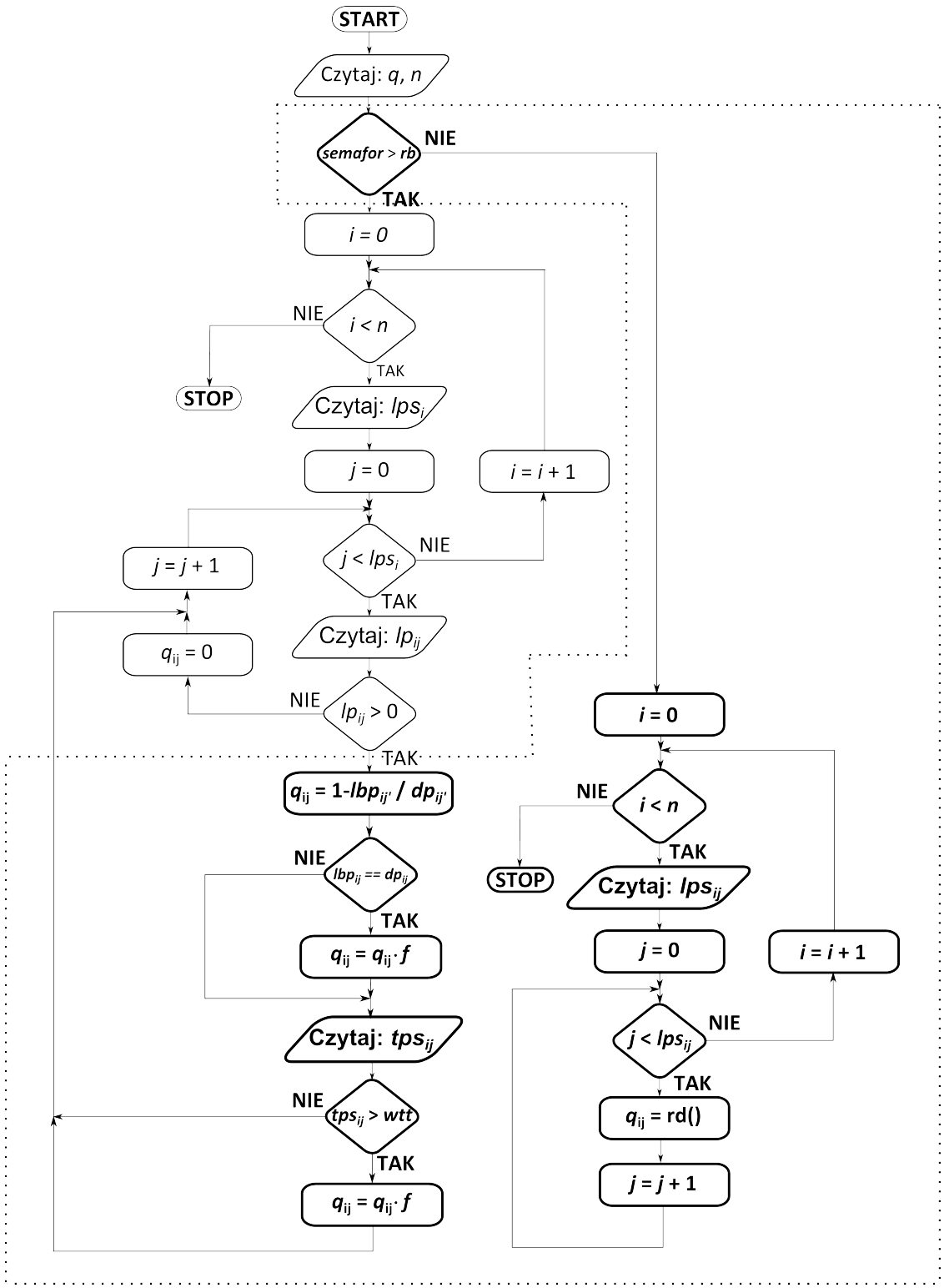
$i = 1, \ldots, n$

$j = 1, \ldots, lps_i$

Obliczone w opisany sposób zyski dla pasów ruchu na wjazdach skrzyżowań z sygnalizacją świetlną w kolejnym etapie wykorzystane są do wyboru odpowiedniej fazy ruchu identycznie jak w algorytmie Most Cars zgodnie z zależnością (12).

Schemat blokowy autorskiego algorytmu w wersji 1.2 przedstawia Rys. 22.

Na schemacie z Rys. 22 wyróżniono część algorytmu, którą dodano w porównaniu do algorytmu Most Cars.



Rys. 22: Schemat blokowy algorytmu In-and-Outbound Lane Control, wersja 1.2.

9. BADANIE RUCHU

W ramach badania ruchu pojazdów przeprowadzono pomiar natężenia ruchu w jednym kierunku oraz pomiar odstępów czasu między pojazdami w wybranych punktach miasta Krapkowic (Rys. 23) i Opola (Rys. 24).

9.1. Pomiar Natężenia Ruchu

9.1.1. Opis przeprowadzonych badań

Pomiar natężenia ruchu wykonano jednocześnie dla wszystkich punktów.

Dla punktów miasta Krapkowice wyniki pomiarów są rezultatem badań własnych. Przeprowadzono je jednocześnie dla wszystkich punktów pomiarowych w okresie między godz. 15:00 a godz. 16:00. Wykorzystano w pracy następujące punkty pomiarowe określone na Rys. 23 str. 93 [171]:

A – ul. 3 Maja parking marketu Biedronka, strumień w kierunku Krapkowic,

B – ul. Ks. Koziołka 33, strumień w kierunku centrum miasta Krapkowic,

C – ul. Żeromskiego 8, strumień w kierunku skrzyżowania z ul. Ks. Koziołka,

D – ul. 1 Maja 22, strumień w kierunku dzielnicy Otmęt,

E – ul. Opolska 12, strumień do skrzyżowania z ul. 1 Maja.

Natomiast dla wybranych punktów sieci miasta Opola wykorzystano wyniki pomiarów uzyskane dzięki uprzejmości Miejskiego Zarządu Dróg w Opolu, które zostały przeprowadzone w 2005 r. Na potrzeby niniejszej pracy wykorzystano wyniki pomiarów wykonane między godz. 15:00 a godz. 16:00. Wykorzystano w pracy następujące punkty pomiarowe określone na Rys. 24 str. 93 [171]:

A – ul. Wrocławska 58-60, strumień w kierunku skrzyżowania z ul. Nysy Łużyckiej,

B – ul. Nysy Łużyckiej 8, strumień w kierunku skrzyżowania z ul. Wrocławską,

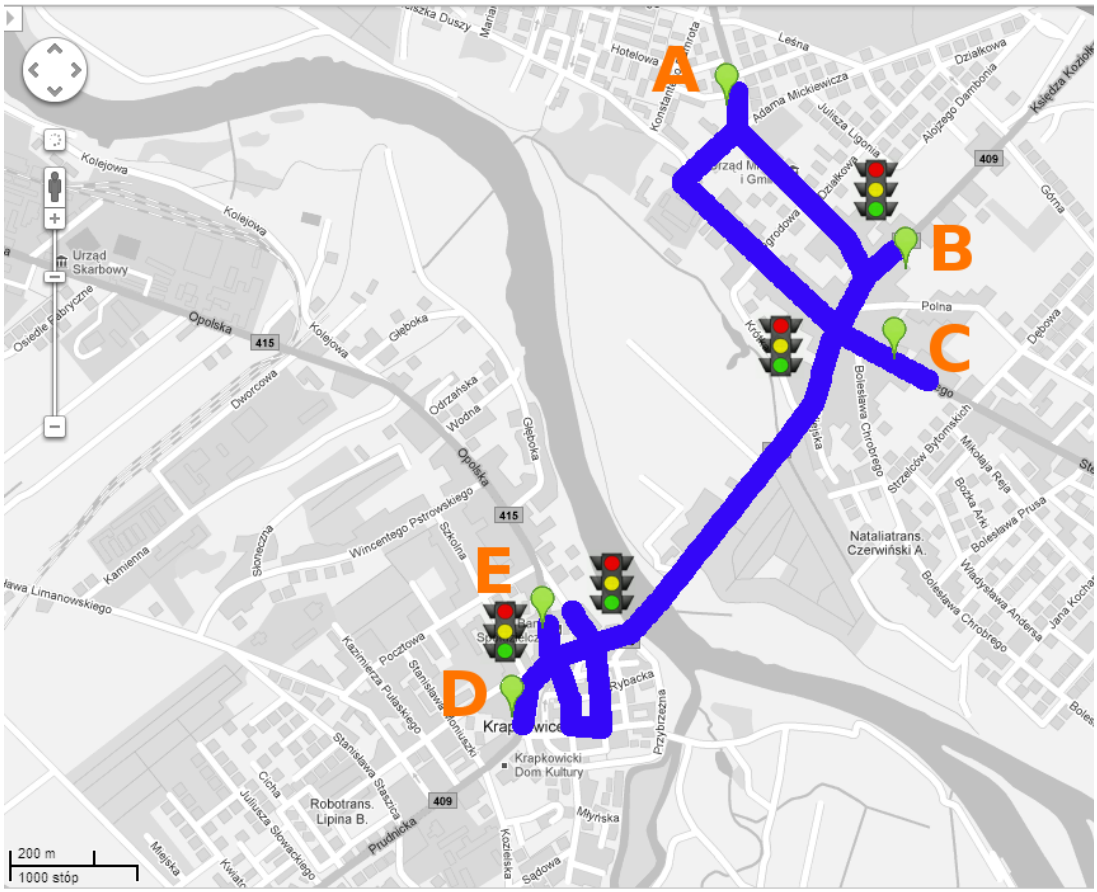
C – ul. Piastowska 12, strumień w kierunku dzielnicy Zaodrze,

D – ul. Krapkowicka 8A, strumień w kierunku ul. Spychalskiego,

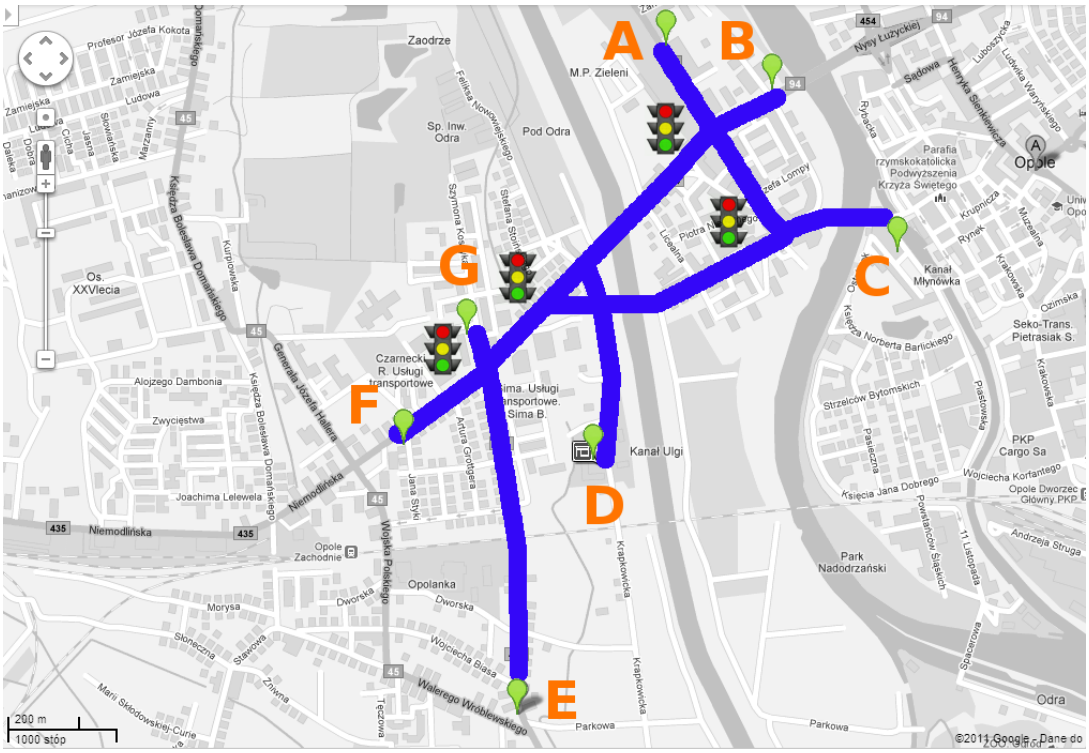
E – ul. Prószkowska 68, strumień w kierunku ul. Niemodlińskiej,

F – ul. Niemodlińska 57, strumień w kierunku centrum miasta,

G – ul. Koszyka 1, strumień w kierunku ul. Niemodlińskiej.



Rys. 23: Mapa punktów pomiarowych we fragmencie sieci miasta Krapkowice



Rys. 24: Mapa punktów pomiarowych we fragmencie sieci miasta Opola

9.1.2. Wyniki badań natężenia ruchu

Wyniki badań własnych liczby pojazdów przejeżdżających przez przekrój drogi w wybranych miejscach miasta Krapkowice zestawiono w Tab. 1. Wartości względne liczby pojazdów wykorzystano w badaniach symulacyjnych w rozdziale 10.4. Natomiast wyniki pomiarów w wybranych punktach miasta Opola przedstawiono w Tab. 2. Rozróżniono pojazdy osobowe i ciężarowe. Następnie obliczono wartość względną liczby pojazdów osobowych i ciężarowych względem maksymalnej wartości pojazdów osobowych ze wszystkich punktów pomiarowych od A do G. Wartości względne wykorzystano w badaniach symulacyjnych opisanych w rozdziale 10.5.

Liczba pojazdów	**Punkty pomiarowe**				
	A	**B**	**C**	**D**	**E**
Łącznie	341	471	138	581	404
Osobowych	325	431	130	**535**	358
Ciężarowych	16	40	8	46	46
Maksymalna osobowych	535				
Osobowych wzgl. maksymalnej	0,61	0,81	0,24	1	0,67
Ciężarowych wzgl. maksymalnej	0,03	0,07	0,01	0,09	0,09

Tab. 1: Wyniki badania natężenia ruchu w wybranych punktach miasta Krapkowice

Liczba pojazdów	**Punkt pomiarowy**						
	A	**B**	**C**	**D**	**E**	**F**	**G**
Łącznie	721	1685	804	204	343	1155	30
Osobowych	618	**1490**	753	193	301	1055	28
Ciężarowych	203	195	51	11	42	100	2
Maksymalna osobowych	1490						
Osobowych wzgl. maksymalnej	0,41	1	0,51	0,13	0,2	0,71	0,02
Ciężarowych wzgl. maksymalnej	0,07	0,13	0,03	0,01	0,03	0,07	0

Tab. 2: Wyniki badania natężenia ruchu w wybranych punktach miasta Opola

9.2. Pomiar Odstępów Czasu Między Pojazdami

9.2.1. Opis przeprowadzonych badań

W przypadku opisu procesu zgłoszeń za pomocą odstępów czasu między pojazdami należy zaznaczyć, że nie są to fizyczne odstępy odpowiadające przestrzennym lukom między kolejnymi pojazdami, ale odstępy czasu mierzone pomiędzy momentami pojawienia się frontów kolejnych pojazdów w przyjętym przekroju [17,172,173]. Badania przeprowadzono w dwóch punktach w Krapkowicach przy ul. Ks. Koziołka 33 i 1 Maja 22 oraz w Opolu przy ul. Wrocławskiej 30 i ul. Nysy Łużyckiej 8. Pomiarów dokonano w godzinach szczytu od $15^{\underline{00}}$ do $16^{\underline{00}}$. Badano strumienie:

- na ul. Księdza Koziołka 33 w Krapkowicach-Otmęcie skierowanym z Gogolina do centrum Krapkowic,
- na ul. 1 Maja 22 skierowanym do dzielnicy Otmęt,
- na ulicy Wrocławskiej 30 skierowanym do skrzyżowania z ul. Nysy Łużyckiej i Niemodlińskiej,
- na ul. Nysy Łużyckiej 8 skierowanym do dzielnicy Zaodrze.

Do badań wykorzystano skrypt w aplikacji Scilab [174], w którym mierzono i zapisywano odstępy czasowe między kolejnymi kliknięciami klawisza na komputerze przenośnym. Do pomiarów czasu wykorzystano funkcje :

- `tic()` - uruchamia licznik odliczający z dokładnością do milisekundy,
- `toc()` - zwraca wartość licznika w chwili wywołania.

Otrzymane wyniki podzielono na klasy według reguły Freedmana-Diaconisa [175]. Następnie sporządzono histogramy odstępów czasowych między pojazdami i estymowano przy pomocy aplikacji EasyFit 5.5 Professional [176] parametry rozkładów. W Tab. 3 przedstawiono część rozkładów poddanych estymacji parametrów i testom zgodności, które uzyskały w testach najniższe statystyki.

Rozkład	Wzór funkcji gęstości prawdopodobieństwa
Birnbauma-Saundersa (Fatigue Life)	$f(x)=\frac{\sqrt{\frac{x}{\beta}}+\sqrt{\frac{\beta}{x}}}{2\alpha(x)}\cdot\phi\left(\frac{1}{\alpha}\left(\sqrt{\frac{x}{\beta}}-\sqrt{\frac{\beta}{x}}\right)\right), \phi(x)=\frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}}$
Burr	$f(x)=\frac{\alpha k\left(\frac{x}{\beta}\right)^{\alpha-1}}{\beta\left(1+\left(\frac{x}{\beta}\right)^{\alpha}\right)^{k+1}}$
Burr (4P)	$f(x)=\frac{\alpha k\left(\frac{x-\gamma}{\beta}\right)^{\alpha-1}}{\beta\left(1+\left(\frac{x-\gamma}{\beta}\right)^{\alpha}\right)^{k+1}}$
Dagum	$f(x)=\frac{\alpha k\left(\frac{x}{\beta}\right)^{\alpha k-1}}{\beta\left(1+\left(\frac{x}{\beta}\right)^{\alpha}\right)^{k+1}}$
Gamma (3P)	$f(x)=\frac{(x-\gamma)^{\alpha-1}}{\beta^{\alpha}\Gamma(\alpha)}e^{-\frac{x-\gamma}{\beta}}, \gamma\le x<+\infty$
Gaussa odwrotny (3P) (Inv. Gaussian 3P)	$f(x)=\sqrt{\frac{\lambda}{2\pi(x-\gamma)^3}}e^{-\frac{\lambda(x-\gamma-\mu)^2}{2\mu^2(x-\gamma)}}$
Kumaraswamy	$f(x)=\frac{\alpha_1\alpha_2 z^{\alpha_1-1}\left(1-z^{\alpha_1}\right)^{\alpha_2}}{b-a}, z\equiv\frac{x-a}{b-a}$
Logarytmicznie logistyczny (3P) (Log-Logistic 3P)	$f(x)=\frac{\alpha}{\beta}\left(\frac{x-\gamma}{\beta}\right)^{\alpha-1}\left(1+\left(\frac{x-\gamma}{\beta}\right)^{\alpha}\right)^{-2}$
Logarytmicznie normalny (Lognormal)	$f(x)=\frac{e^{-\frac{1}{2}\left(\frac{lnx-\mu}{\sigma}\right)^2}}{x\sigma\sqrt{2\pi}}$
Lognormal (3P)	$f(x)=\frac{1}{(x-\gamma)\sigma\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{\ln(x-\gamma)-\mu}{\sigma}\right)^2}$
Pearson 5	$f(x)=\frac{e^{-\frac{\beta}{x}}}{\beta\Gamma(\alpha)\left(\frac{x}{\beta}\right)^{\alpha+1}}$

Rozkład	Wzór funkcji gęstości prawdopodobieństwa
Pearson 5 (3P)	$f(x)=\frac{e^{-\frac{\beta}{x-\gamma}}}{\beta\,\Gamma(\alpha)\left(\frac{x-\gamma}{\beta}\right)^{\alpha+1}}$
Pearson 6	$f(x)=\frac{\left(\frac{x}{\beta}\right)^{\alpha_1-1}}{\beta\,\mathrm{B}(\alpha_1,\alpha_2)\left(1+\frac{x}{\beta}\right)^{\alpha_1+\alpha_2}}$ $\mathrm{B}(\alpha_1,\alpha_2)=\int_0^1 t^{\alpha_1-1}(1-t)^{\alpha_2-1}dt$
Pearson 6 (4P)	$f(x)=\frac{\left(\frac{x-\gamma}{\beta}\right)^{\alpha_1-1}}{\beta\,\mathrm{B}(\alpha_1,\alpha_2)\left(1+\frac{x-\gamma}{\beta}\right)^{\alpha_1+\alpha_2}}$
Uogólniony gamma (Gen. Gamma)	$f(x)=\frac{k\,x^{k\alpha-1}}{\beta^{k\alpha}\Gamma(\alpha)}e^{-\left(\frac{x}{\beta}\right)^k}$
Uogólniony gamma (4P) (Gen. Gamma 4P)	$f(x)=\frac{k\,(x-\gamma)^{k\alpha-1}}{\beta^{k\alpha}\Gamma(\alpha)}e^{-\left(\frac{x-\gamma}{\beta}\right)^k}$
Uogólniony Pareto (Gen. Pareto)	$f(x)=\begin{cases}\frac{1}{\sigma}\left(1+k\frac{(x-\mu)}{\sigma}\right)^{-1-\frac{1}{k}}, & k\neq 0\\ \frac{1}{\sigma}e^{\left(\frac{-x-\mu}{\sigma}\right)}, & k=0\end{cases}$
Weibull (3P)	$f(x)=\frac{\alpha}{\beta}\left(\frac{\alpha-\gamma}{\beta}\right)^{\alpha-1}e^{-\left(\frac{x-\gamma}{\beta}\right)^\alpha}$

Tab. 3: Rozkłady dopasowywane do uzyskanych wyników pomiarów

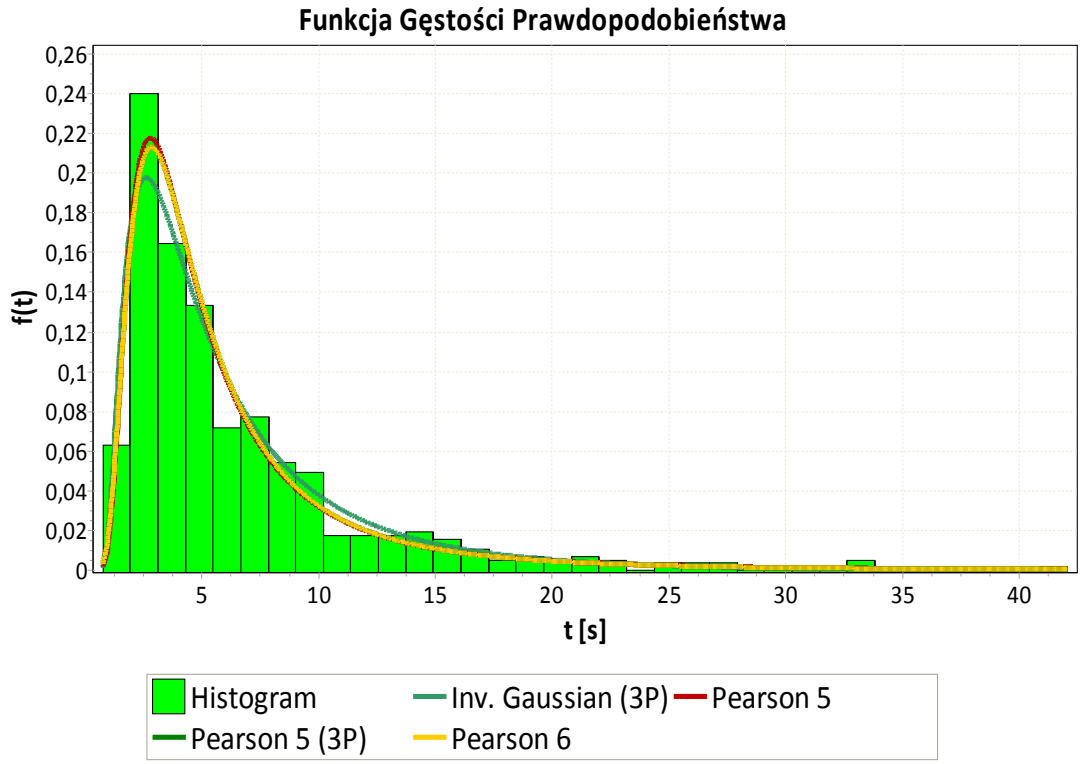
9.2.2. Wyniki badań odstępów czasu między pojazdami

9.2.2.1. Krapkowice ul. 1 Maja 22

W wyniku przeprowadzonych badań otrzymano 570 odstępów czasowych w zakresie od 0,75 [s] do 42,078 [s]. Wartość oczekiwana pomiarów wynosi 6,3238 [s], odchylenie standardowe 5,3556 [s] i skośność 2,4182.

Wyniki pomiarów podzielono na 35 klas o szerokości 1,2 [s] i przedstawiono na Rys. 25 za pomocą histogramu. Na tym samym Rys. 25 umieszczono również wykres

funkcji gęstości prawdopodobieństwa rozkładu Pearson 5, Pearson 6, Pearson 5 (3P). Rozkłady te zajęły najwyższe kolejne miejsca w rankingu pod względem najniższej statystyki otrzymanej w teście zgodności metodą Andersona-Darlinga jak również wysokie lokaty w rankingach otrzymanych w wyniku przeprowadzenia testów zgodności metodami Kołmogorowa-Smirnowa i χ^2. Dodano również rozkład odwrotny Gaussa z trzema parametrami (Inv. Gaussian 3P), który wybrano do symulacji obok rozkładu Pearson 5. Na Rys. 25 pominięto rozkłady Johnson SB i uogólniony rozkład Pareto (Gen. Pareto), które uzyskały wysokie lokaty w teście Kołmogorowa-Smirnowa, natomiast bardzo niskie w teście Andersona-Darlinga. W teście χ^2 nie otrzymano statystyk dla tych rozkładów.



Rys. 25: Histogram wyników pomiarów wykonanych na ul. 1 Maja 22 w Krapkowicach

Skrócony ranking rozkładów dla badanej próby przedstawiono w Tab. 4. Natomiast parametry rozkładów o najniższych statystykach uzyskane w procesie estymacji przedstawiono w Tab. 5.

Rozkład	Metoda testowania hipotez					
	Kołmogorowa Smirnowa		Andersona Darlinga		χ^2	
	Statystyka	Miejsce	Statystyka	Miejsce	Statystyka	Miejsce
Pearson 5	0,04437	3	1,15260	1	12,986	2
Pearson 6	0,04764	4	1,29400	2	15,436	6
Pearson 5 (3P)	0,04766	5	1,29620	3	15,435	5
Pearson 6 (4P)	0,04768	6	1,29670	4	15,436	7
Inv. Gaussian (3P)	0,05427	11	1,48980	8	21,772	11

Tab. 4: Wyniki przeprowadzenia testów zgodności dla wybranych rozkładów (ul. 1 Maja 22, Krapkowice)

Rozkład	Parametry
Pearson 5	α=2,4524; β=9,616
Pearson 6	α=2,5911; β=10,605; γ=-0,12232
Pearson 5 (3P)	α_1=40,906; α_2=2,5932; β=0,2548
Pearson 6 (4P)	α_1=232,28; α_2=2,5915; β=0,04551; γ=-0,09986
Inv. Gaussian (3P)	λ=7,9447; μ=5,9859; γ=0,33789

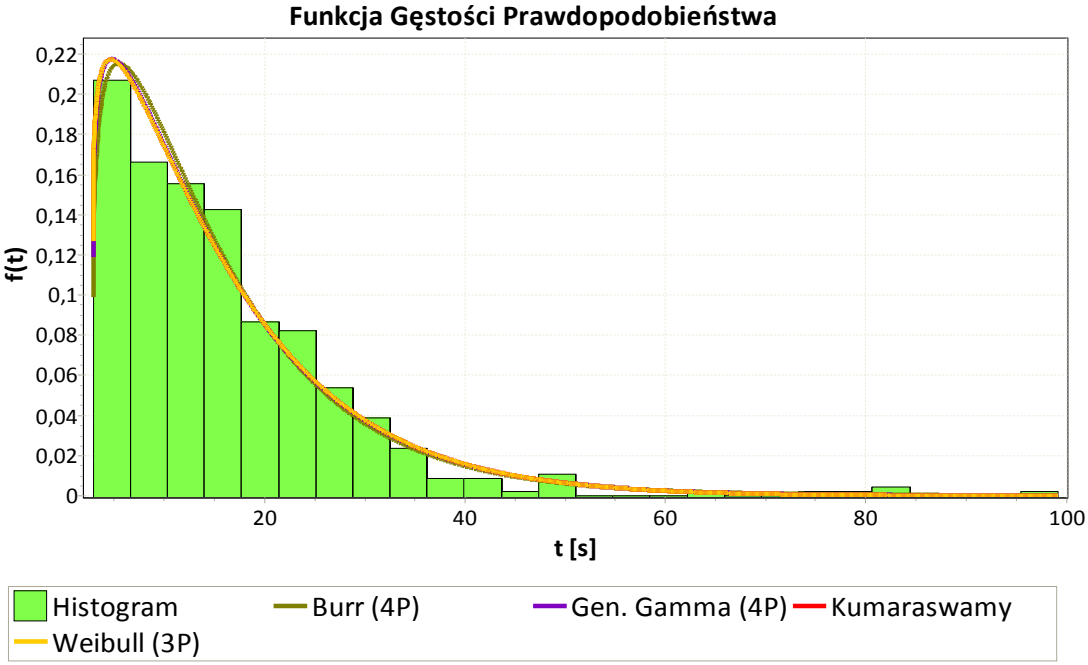
Tab. 5: Parametry wybranych rozkładów odstępów między pojazdami uzyskanych na ul. 1 Maja 22 w Krapkowicach

9.2.2.2. Krapkowice ul. Ks. Koziołka 33

Jako wynik przeprowadzonych badań otrzymano 463 odstępy czasowe w przedziale od 2,906 [s] do 99,188 [s]. Wartość średnia pomiarów wynosi 15,921 [s], odchylenie standardowe 12,115 [s] i skośność 2,5338.

Otrzymane odstępy czasowe podzielono na 26 klas o szerokości 3,502 [s] i zobrazowano na Rys. 26 w postaci histogramu.

Dodatkowo na Rys. 26 przedstawiono wykresy funkcji gęstości prawdopodobieństwa rozkładów: Burra z czterema parametrami (Burr 4P), który ma najniższą statystykę w teście zgodności Kołmogorowa-Smirnowa, rozkładu Kumaraswamy, którego statystyka w teście Andersona-Darlinga okazuje się najniższa, uogólniony Gamma z czterema parametrami (Gen. Gamma 3P) oraz Weibulla z trzema parametrami (Weibull 3P), którego statystyka w teście Andersona-Darlinga ma drugie miejsce natomiast w pozostałych testach trzecie.



Rys. 26: Histogram wyników pomiarów wykonanych na ul. Ks. Koziołka 33 w Krapkowicach

Wymienione wyżej rozkłady oraz rozkład Birnbauma-Saundersa (Fatigue Life) zestawiono w Tab. 6 w postaci rankingu. Pominięto piąty w kolejności według rankingu po przeprowadzeniu testów zgodności metodą Kołmogorowa-Smirnowa uogólniony rozkład Pareto (Gen. Pareto), który w testach pozostałymi metodami osiągnął niskie lokaty. Natomiast w Tab. 7 przedstawiono estymowane parametry tych rozkładów.

Rozkład	**Metoda testowania hipotez**					
	Kołmogorowa Smirnowa		**Andersona Darlinga**		χ^2	
	Statystyka	Miejsce	Statystyka	Miejsce	Statystyka	Miejsce
Burr (4P)	0,03168	1	0,759	3	17,162	15
Kumaraswamy	0,03479	2	0,748	1	11,102	2
Weibull (3P)	0,03479	3	0,749	2	11,102	3
Gen. Gamma (4P)	0,03488	4	0,761	4	12,192	5
Fatigue Life	0,03546	6	1,106	7	17,092	14

Tab. 6: Wyniki przeprowadzenia testów zgodności dla wybranych rozkładów (ul. Ks. Koziołka 33, Krapkowice)

Rozkład	Parametry
Burr (4P)	k=11,152; α=1,1904; β=97,384; γ=2,8875
Kumaraswamy	α_1=1,1148; α_2=4,9989·10^6; a=2,8993; b=1,3834·10^7
Weibull (3P)	α=1,1148; λ=0,0738; δ=2,8993
Gen. Gamma (4P)	k=1,0876; α=1,0443; β=12,905; γ=2,8972
Fatigue Life	α=0,73296; β=12,548

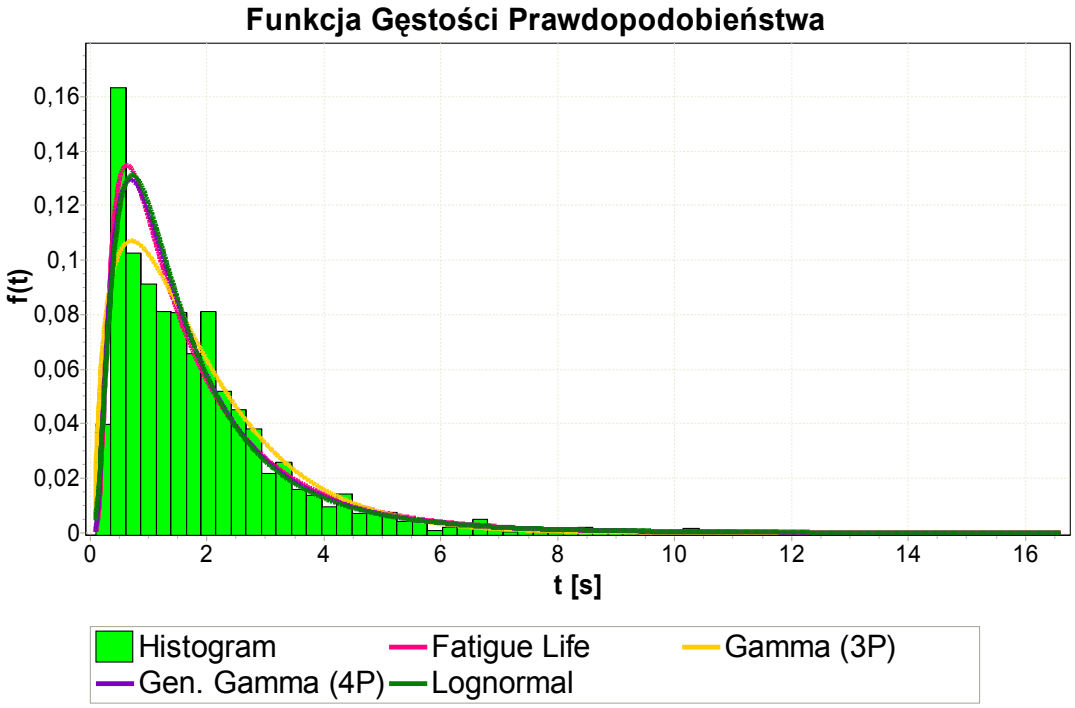
Tab. 7: Parametry wybranych rozkładów odstępów między pojazdami na ul. Ks. Koziołka 33 w Krapkowicach

9.2.2.3. Opole ul. Nysy Łużyckiej 8

Przeprowadzone badania zaowocowały uzyskaniem 1960 odstępów czasowych, które zawierały się w przedziale od 0,094 [s] do 16,61 [s]. Wartość średnia pomiarów wyniosła 1,8394 [s], odchylenie standardowe 2,5525 [s], skośność 2,7047.

Otrzymane wyniki pomiarów podzielono na 64 klasy o szerokości 0,25 [s] i zobrazowano za pomocą histogramu (Rys. 27).

Rys. 27 przedstawia również funkcje gęstości prawdopodobieństwa: rozkładów Birnbauma-Saundersa (Fatigue Life) oraz uogólnionego gamma z czterema parametrami (Gen. Gamma 4P), które w teście zgodności Andersona-Darlinga uzyskały w rankingu miejsca odpowiednio pierwsze i trzecie, jak również rozkładu gamma



Rys. 27: Histogram wyników pomiarów wykonanych na ul. Nysy Łużyckiej 8 w Opolu

z trzema parametrami (Gamma 3P), który w rankingu powstałym po przeprowadzeniu testu zgodności metodą Kołmogorowa-Smirnowa zajął trzecie miejsce. Dodano również funkcję gęstości prawdopodobieństwa rozkładu logarytmicznie normalnego (Lognormal), który jest opisywany w literaturze [17,172].

Na Rys. 27 nie uwzględniono funkcji gęstości prawdopodobieństwa rozkładu uogólnionego Pareto (Gen. Pareto), który uzyskał najniższą statystykę w teście Kołmogorowa-Smirnowa, jednak w teście Andersona-Darlinga uzyskał 40. miejsce. Natomiast w teście χ^2 nie uzyskano statystyki dla tego rozkładu. Statystyki przeprowadzonych testów zgodności innych rozkładów przedstawiono w Tab 8.

Poniżej w Tab. 9 zaprezentowano zestawienie parametrów wybranych rozkładów po procesie estymacji. Wyboru dokonano na podstawie rankingu rozkładów uzyskanego jako wynik testów zgodności (Tab. 8). W zestawieniu rozkładów uwzględniono również rozkład logarytmicznie normalny.

Rozkład	**Metoda testowania hipotez**					
	Kołmogorowa Smirnowa		**Andersona Darlinga**		χ^2	
	Statystyka	Miejsce	Statystyka	Miejsce	Statystyka	Miejsce
Gen. Pareto	0,02876	1	85,02400	40	niedostępne	
Fatigue Life	0,03750	2	4,81840	1	59,61	3
Gamma (3P)	0,04029	3	6,88780	7	60,09	4
Gen. Gamma (4P)	0,04233	4	5,61880	3	61,14	5
Lognormal	0,04637	10	7,14630	9	71,002	13

Tab. 8: Wyniki przeprowadzenia testów zgodności dla wybranych rozkładów (ul. Nysy Łużyckiej 8, Opole)

Rozkład	**Parametry**
Gen. Pareto	k=-0,01577; σ=1,568; μ=0,29581
Fatigue Life	α=0,84522; β=1,3551
Gamma (3P)	α=1,5473; β=1,1289; γ=0,09281
Gen. Gamma (4P)	k=2,3252; α=1,3604; β=1,2669
Lognormal	σ=0,79171; μ=0,30386

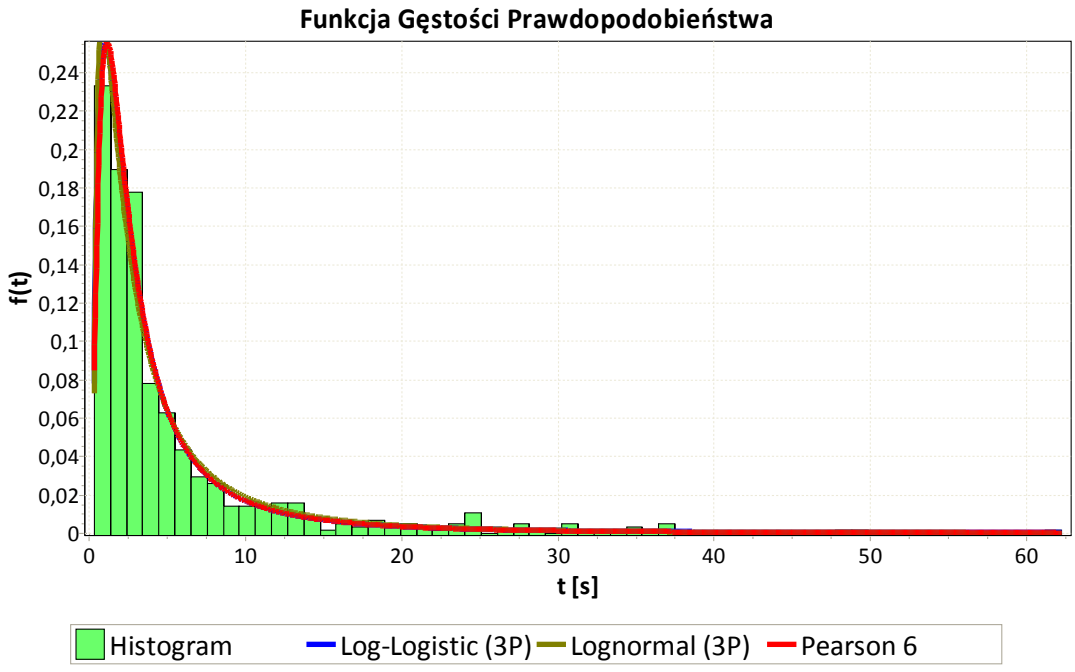
Tab. 9: Parametry wybranych rozkładów odstępów między pojazdami na ul. Nysy Łużyckiej 8 w Opolu

9.2.2.4. Opole ul. Wrocławska 30

W wyniku przeprowadzonych badań otrzymano 575 odstępów czasowych, które zawierały się w przedziale od 0,328 [s] do 62,235 [s]. Wartość średnia pomiarów wynosi 5,388 [s], odchylenie standardowe 7,357 [s], skośność 3,2358.

Otrzymane wyniki pomiarów podzielono na 60 klas o szerokości 1,03 [s] i zobrazowano za pomocą histogramu (Rys. 28).

Na Rys. 28 przedstawiono również wykres funkcji gęstości prawdopodobieństwa rozkładu Pearson 6, dla którego to rozkładu uzyskano najmniejszą statystykę podczas testowania hipotezy metodą Kołmogorowa-Smirnowa oraz logarytmicznie logistycznego z trzema parametrami (Log-Logistic 3P), dla którego otrzymano najmniejszą statystykę metodą Andersona-Darlinga oraz metodą χ^2. Ponadto na rysunku uwzględniono funkcję gęstości prawdopodobieństwa rozkładu logarytmicznie normalnego z trzema parametrami (Lognormal 3P). Rozkład ten jest wykorzystywany do opisu rozkładu odstępów czasowych między pojazdami w strumieniu [17,172]. W testach zgodności dla tego rozkładu hipoteza zerowa nie została odrzucona. Statystyki przeprowadzonych testów zgodności innych rozkładów przedstawiono w Tab 10.



Rys. 28: Histogram wyników pomiarów wykonanych na ul. Wrocławskiej 33 w Opolu

Rozkład	Metoda testowania hipotez					
	Kołmogorowa Smirnowa		Andersona Darlinga		χ^2	
	Statystyka	Miejsce	Statystyka	Miejsce	Statystyka	Miejsce
Pearson 6	0,02347	1	0,63620	4	18,22	15
Log-Logistic (3P)	0,02601	2	0,56034	1	11,74	1
Burr	0,02617	3	0,79529	10	18,09	14
Dagum	0,02673	4	0,68863	5	15,52	9
Lognormal (3P)	0,03578	9	0,79440	9	14,050	5

Tab. 10: Wyniki przeprowadzenia testów zgodności dla wybranych rozkładów (ul. Wrocławska 30, Opole)

Poniżej w Tab. 11 zaprezentowano zestawienie parametrów wybranych rozkładów po procesie estymacji. Wyboru dokonano na podstawie rankingu rozkładów uzyskanego jako wynik testów zgodności. W zestawieniu rozkładów uwzględniono również rozkłady: logarytmicznie normalny z trzema parametrami oraz uogólniony rozkład gamma w wersji czteroparametrycznej.

Rozkład	Parametry
Pearson 6	α_1=4,2458; α_2=1,6075; β=0,91638
Log-Logistic (3P)	α=1,3986; β=2,4918; γ=0,31165
Burr	k=0,65882; α=1,999; β=2,0438
Dagum	k=2,3252; α=1,3604; β=1,2669
Lognormal (3P)	σ=1,1904; μ=0,95393; γ=0,23645

Tab. 11: Parametry wybranych rozkładów odstępów między pojazdami na ul. Wrocławskiej 30 w Opolu

9.3. Wnioski

Badania odstępów czasu między pojazdami przeprowadzono w czterech różniących się od siebie miejscach. Cechą charakterystyczną punktu pomiarowego w Krapkowicach przy ul. 1 Maja 22 jest ruch pojazdów do centrum miasta z dróg dojazdowych z Kędzierzyna Koźla, Prudnika i Opola. Jest to również droga dojazdowa do dzielnicy Otmęt, w której znajdują się siedziby wielu firm produkcyjno-handlowych. Inne charakterystyki ruchu uzyskano w punkcie pomiarowym przy ul. Ks. Koziołka 33 w Krapkowicach-Otmęcie. Pomiarów dokonywano podczas nieaktywnej sygnalizacji

świetlnej na skrzyżowaniu ul. Ks. Koziołka z ul. 3 Maja. Ruch w tym miejscu był falowy. Pojazdy przejeżdżały w grupach. W pobliżu punktu pomiarowego znajdują się dwa markety. Ul. Ks. Koziołka jest ponadto drogą dojazdową z autostrady A4 do centrum miasta Krapkowic ze zjazdu nr 248 (Krapkowice-Gogolin). Jest również drogą łączącą Gogolin i Krapkowice.

Punkt pomiarowy przy ul. Nysy Łużyckiej 8 w Opolu charakteryzuje się drogą trzypasmową w kierunku do skrzyżowania z ul. Wrocławską. Jest to najkrótsza droga prowadząca z os. ZWM, miasteczka akademickiego Uniwersytetu Opolskiego i centrum miasta Opola w kierunku dzielnicy Zaodrze. Dojazd ul. Wrocławskiej do skrzyżowania z ul. Nysy Łużyckiej i ul. Niemodlińskiej charakteryzuje się kilkusetmetrową prostą bez skrzyżowań, dojazdów i zjazdów. Jest to również droga z obwodnicy północnej Opola oraz droga z położonego na obrzeżach miasta centrum handlowego do centrum miasta. W punkcie pomiarowym droga ma dwa pasy w kierunku do skrzyżowania.

Do badań symulacyjnych (Rozdz. 10.) celem generowania ruchu na wjazdach do sieci Krapkowic użyto dwóch generatorów liczb losowych wybranych na podstawie badań przeprowadzonych na ul. 1 Maja 22 (Tab. 12) oraz dwa generatory liczb losowych wybrane na podstawie badań przeprowadzonych na ul. Ks. Koziołka 33 (Tab. 13).

Rozkład	Parametry
Inv. Gaussian (3P)	λ=7,9447; μ=5,9859; γ=0,33789
Pearson 5	α=2,4524; β=9,616

Tab. 12: Rozkłady wybrane do badań symulacyjnych na podstawie pomiarów odstępów czasowych wykonanych na ul. 1 Maja 22

Z badań przeprowadzonych na ul. Ks. Koziołka 33 wybrano w miejsce rozkładu Kumaraswamy rozkład Weibulla z trzema parametrami (Weibull 3P), który nie jest rozkładem obustronnie ograniczonym.

Rozkład	Parametry
Burr (4P)	k=11,152; α=1,1904; β=97,384; γ=2,8875
Weibull (3P)	α=1,1148; λ=0,0738; δ=2,8993

Tab. 13: Rozkłady wybrane do badań symulacyjnych na podstawie pomiarów odstępów czasowych wykonanych na ul. Ks. Koziołka 33

Natomiast dla sieci drogowej miasta Opola wybrano dwa generatory liczb losowych na podstawie badań przeprowadzonych na ul. Nysy Łużyckiej 8 (Tab. 14) oraz do kolejnych badań symulacyjnych dla tej sieci wybrano dwa generatory liczb losowych na podstawie badań przeprowadzonych na ul. Wrocławskiej 30 (Tab. 15).

Rozkład	Parametry
Fatigue Life	α=0,84522; β=1,3551
Lognormal	σ=0,79171; μ=0,30386

Tab. 14: Rozkłady wybrane do badań symulacyjnych na podstawie pomiarów odstępów czasowych wykonanych na ul. Nysy Łużyckiej 8

Rozkład	Parametry
Log-Logistic (3P)	α=1,3986; β=2,4918; γ=0,31165
Pearson 6	α_1=4,2458; α_2=1,6075; β=0,91638

Tab. 15: Rozkłady wybrane do badań symulacyjnych na podstawie pomiarów odstępów czasowych wykonanych na ul. Wrocławskiej 30

Należy nadmienić, że ruch pojazdów na drogach miast można ogólnie opisać jako proces stochastyczny niestacjonarny. Dlatego weryfikacja hipotez statystycznych odnośnie do typu rozkładu może być obarczona dużym błędem, a także nie można się spodziewać powtarzalności serii pomiarowych. Otrzymane zatem wyniki dotyczące typu rozkładu należy traktować jako przybliżone, orientacyjne. Z drugiej strony, późniejsze przykłady symulacyjne w zakresie sterowania ruchem drogowym wskazują, że działanie algorytmów sterowania nie jest istotnie zależne od typu rozkładu rozważanej zmiennej bazowej, tj. czasu odstępu pomiędzy pojazdami.

10. BADANIA SYMULACYJNE

10.1. Opis Badań

10.1.1. Parametry ogólne

Badania symulacyjne przeprowadzono dla następujących parametrów:

- politykę jazdy pojazdów przeprowadzono według algorytmu najkrótszej ścieżki („Normal shortest path" [3]),
- włączono możliwość wystąpienia zdarzeń wyjątkowych (Accidents = true),
- włączono możliwość wyznaczenia nowej trasy w razie wystąpienia zdarzeń wyjątkowych (opcja Rerouting = true),
- usuwanie pojazdów oczekujących powyżej określonego progu czasu oczekiwania przed węzłem (Removal stuck cars = false),
- w symulacji uwzględniono jedynie bezkolizyjne fazy ruchu (Sign contoller switches trafficlight safely = true).

Wartości progów dla węzłów brzegowych (edge node) ustawiono zgodnie z wartościami względnymi liczby pojazdów otrzymanymi na podstawie badań przedstawionych w Tab 1 (str. 94) dla infrastruktury miasta Krapkowice oraz w Tab. 2 (str. 94) dla infrastruktury miasta Opola.

Dla danych ustawień wybranego algorytmu i generatora ruchu wykonano serię symulacji liczącą 50000 cykli dla każdego z trzech ziaren generatora liczb losowych. Przyjęto następujące wartości ziarna generatora liczb losowych: 1000, 8000, 13000.

Badania symulacyjne przeprowadzono na komputerze o następujących parametrach:

- procesor: Intel Core 2 Duo E8400 3,0 GHz,
- pamięć RAM: DDR2 2GB (800MHz),
- płyta główna: Asus P5K-E,
- system operacyjny: Ubuntu 10.04 desktop amd64,
- środowisko JAVA SUN 1.6

10.1.2. Parametry algorytmów

Przeprowadzono badania symulacyjne dla algorytmów sterowania opisanych w rozdziale 8. z parametrami przedstawionymi w Tab. 16.

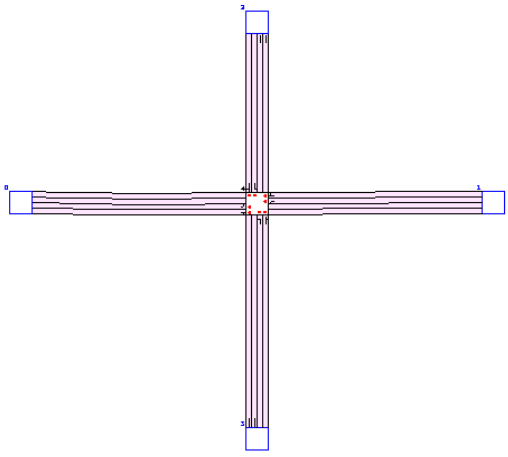
Algorytm sterowania	Parametry
Most Cars	Brak
Local Hill-Climbing	Brak
TC-1 Bucket 2.0	γ = 0,9; próg losowości = 0,01;
RL SARSA 5	γ = 0,95; α = 0,7; próg losowości = 0,01;
GenNeural	liczba kroków życia osobnika = 20; liczba osobników w populacji = 20; próg dziedziczenia = 0,1; prawdopodobieństwo mutacji = 0,01;
In-and-Outbound Lane Control	*wtt* = {2; 3; 4}; *f* = {2; 3; 4; 5; 6}; próg losowości *rb* = {0,0; 0,02};

Tab. 16: Parametry algorytmów sterowania użytych w badaniach symulacyjnych

10.2. Odosobnione Skrzyżowanie

10.2.1. Badana infrastruktura

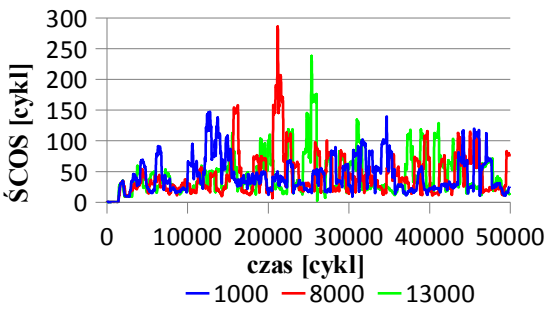
Zamodelowano skrzyżowanie jak na Rys. 29. Odległość każdego węzła wjazdowego od skrzyżowania wynosi 100m. Każdy wjazd ma dwa pasy ruchu. Również wyjazdy ze skrzyżowania liczą dwa pasy. Progi generowania pojazdów dla każdego z czterech wjazdów ustawiono na 0,8.



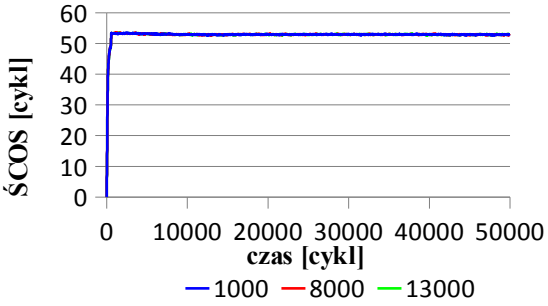
Rys. 29: Infrastruktura skrzyżowania odosobnionego

10.2.2. Wyniki badań dla generatora ruchu według rozkładu Fatigue Life

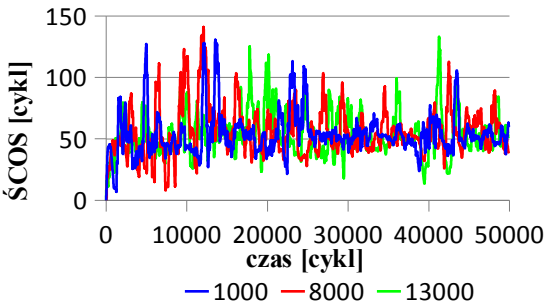
Średnie czasy oczekiwania przed skrzyżowaniem przedstawione w Tab. 17 sugerują największą efektywność sterowania przy zastosowaniu algorytmu Local Hill-Climbing. Okazuje się jednak, że chwilowe ŚCOS uzyskane podczas sterowania przy pomocy tego algorytmu przekraczają nawet 200 cykli (Rys. 30), co nie ma miejsca w przypadku algorytmu autorskiego (Rys. 31). Kolejnym mankamentem algorytmu Local Hill-Climbing jest mała liczba pojazdów osiągających cel podróży w trakcie trwania symulacji (Tab. 18). Osiągnięta liczba pojazdów dojeżdżających do celu 73499 przy sterowaniu tym algorytmem stanowi 74% liczby pojazdów osiągających cel przy zastosowaniu mniej złożonych obliczeniowo algorytmów jak Most Cars czy algorytmu autorskiego. Czas wykonania algorytmu Most Cars jest dla tej infrastruktury ośmiokrotnie krótszy od czasu wykonania algorytmu Local Hill-Climbing (Tab. 19). Pozostałe algorytmy TC-1 Bucket 2.0, RL SARSA 5 i GenNeural osiągały podobne średnie czasy oczekiwania oraz liczby pojazdów osiągających cel podróży do algorytmów Most Cars oraz autorskiego. Podczas sterowania przy użyciu algorytmu GenNeural uzyskano chwilowe ŚCOS przekraczające wartości 100 cykli (Rys. 32).



Rys. 30: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem Local Hill-Climbing



Rys. 31: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=4 f=4 rb=0,0)



Rys. 32: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem GenNeural

Podsumowując najefektywniejszymi algorytmami w tych warunkach wydają się być algorytmy Most Cars, autorski oraz RL SARSA 5.

Algorytm	Średni czas oczekiwania przed skrzyżowaniami ŚCOS [cykl]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	51,89	51,90	51,90
Local Hill-Climbing	42,69	43,62	43,01
TC-1 Bucket 2.0	51,66	51,64	51,66
RL SARSA 5	51,91	51,88	51,90
GenNeural	53,16	55,42	53,76
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,0	52,27	52,25	52,35
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	52,27	52,25	52,35
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	52,27	52,25	52,35
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	52,17	52,12	52,18
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	52,16	52,12	52,17
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	52,05	52,05	52,04
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,02	52,06	52,05	52,08

Tab. 17: Średnie czasy oczekiwania przed skrzyżowaniami uzyskane dla generatora ruchu według rozkładu Fatigue Life

Algorytm	Liczba pojazdów, które dotarły do celu [pojazd]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	99898	99895	99895
Local Hill-Climbing	73499	73284	73463
TC-1 Bucket 2.0	99898	99892	99893
RL SARSA 5	99893	99889	99891
GenNeural	95308	91010	93798
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,0	99891	99892	99891
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	99891	99892	99891
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	99891	99892	99891
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	99889	99888	99890
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	99890	99888	99892
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	99891	99889	99893
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,02	99891	99889	99891

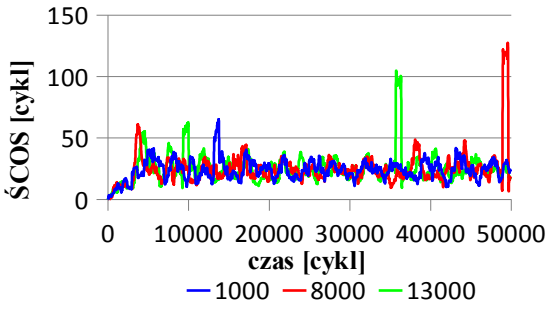
Tab. 18: Liczba pojazdów, które dotarły do celu podczas trwania symulacji dla generatora ruchu według rozkładu Fatigue Life

Algorytm	Czas wykonania algorytmu [ns]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	11017	10939	11013
Local Hill-Climbing	89628	90923	91651
TC-1 Bucket 2.0	416667	417163	430123
RL SARSA 5	19170	18928	18671
GenNeural	44603	42908	43001
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,0	16010	16154	16200
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	16308	16460	15730
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	16433	16493	16149
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	16853	17212	17453
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	15956	15301	15913
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	16134	16092	16122
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,02	16121	15448	15716

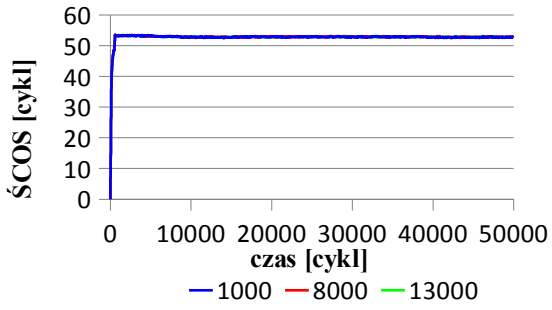
Tab. 19: Czasy wykonania algorytmów dla generatora ruchu według rozkładu Fatigue Life

10.2.3. Wyniki badań dla generatora ruchu według rozkładu Lognormal

Podobnie jak w warunkach ruchu stworzonych przez generator Fatigue Life również i w tych warunkach najkrótsze ŚCOS osiągnięto podczas sterowania przy pomocy algorytmu Local Hill-Climbing (Tab. 20). Podobnie jak w poprzednich warunkach chwilowe ŚCOS przekraczały wartość 100 cykli (Rys. 33). Potwierdza to fakt, że algorytm przez dłuższy czas faworyzował wybrane wloty skrzyżowania kosztem innych wlotów, na których pojazdy były wstrzymywane. Potwierdzeniem tej tezy jest mała liczba pojazdów osiągających cel podróży (Tab. 21). Zjawisko takie nie występowało



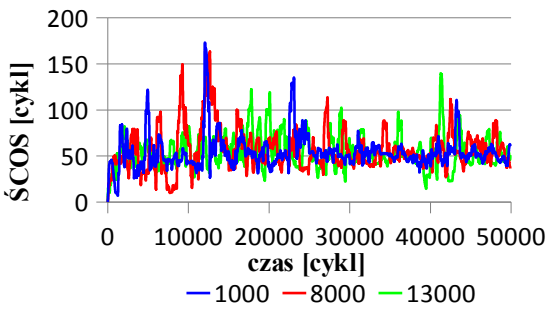
Rys. 33: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem Local Hill-Climbing



Rys. 34: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=4 f=4 rb=0,0)

podczas sterowania za pomocą algorytmu Most Cars, TC-1 Bucket 2.0, RL SARSA 5 oraz autorskiego (Rys. 34).

Problemy ze sterowaniem wykazywał również algorytm GenNeural, który chociaż ŚCOS oraz liczby pojazdów osiągających cel podróży pozwalał uzyskać na zadowalającym poziomie, to jednak chwilowe ŚCOS wielokrotnie przekraczały wartość 100 cykli (Rys. 35).



Rys. 35: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem GenNeural

Zadowalające wyniki osiągnięto podczas sterowania algorytmem RL SARSA 5. Algorytm ten niewiele ustępował algorytmowi autorskiemu pod względem czasu wykonania (Tab. 22). W porównaniu jednak do algorytmu Most Cars czas wykonania jest już prawie dwukrotnie dłuższy.

Algorytm	**Średni czas oczekiwania przed skrzyżowaniami ŚCOS [cykl]**		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	51,87	51,85	51,86
Local Hill-Climbing	23,39	24,36	24,67
TC-1 Bucket 2.0	51,64	51,61	51,61
RL SARSA 5	51,86	51,86	51,85
GenNeural	53,41	56,11	53,94
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	52,12	52,16	52,17
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	52,12	52,16	52,17
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	52,12	52,16	52,17
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	52,05	52,04	52,07
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	52,04	52,09	52,06
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	51,99	51,99	52,00
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,02	51,99	52,01	52,00

Tab. 20: Średnie czasy oczekiwania przed skrzyżowaniami uzyskane dla generatora ruchu według rozkładu Lognormal

Algorytm	Liczba pojazdów, które dotarły do celu [pojazd]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	99896	99894	99896
Local Hill-Climbing	74000	74060	74044
TC-1 Bucket 2.0	99895	99891	99893
RL SARSA 5	99890	99890	99892
GenNeural	94883	89684	92979
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	99891	99891	99894
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	99891	99891	99892
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	99891	99891	99894
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	99892	99892	99893
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	99891	99890	99893
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	99890	99892	99893
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,02	99890	99892	99893

Tab. 21: Liczba pojazdów, które dotarły do celu podczas trwania symulacji dla generatora ruchu według rozkładu Lognormal

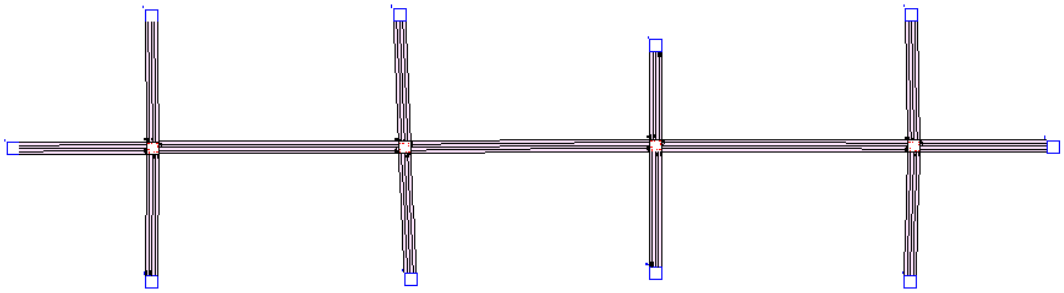
Algorytm	Czas wykonania algorytmu [ns]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	10594	10387	10617
Local Hill-Climbing	90399	88989	90672
TC-1 Bucket 2.0	417557	416094	409648
RL SARSA 5	19097	18914	18062
GenNeural	45722	43520	44613
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	16384	16161	16111
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	16098	15868	15965
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	16530	15669	16347
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	16166	16486	16454
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	16523	16495	15874
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	16598	16187	16122
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,02	16288	16536	15726

Tab. 22: Czasy wykonania algorytmów dla generatora ruchu według rozkładu Lognormal

10.3. Arteria

10.3.1. Badana infrastruktura

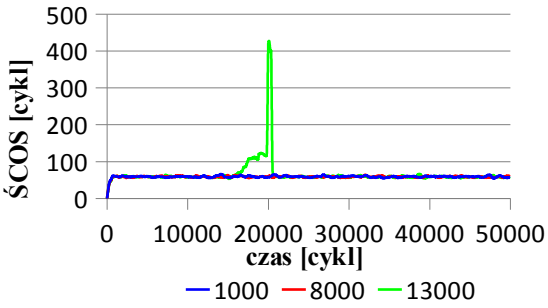
W badaniach wykorzystano infrastrukturę jak na Rys. 36. Odległość między skrzyżowaniami wynosi 200m. Wjazdy poziome oddalone są od najbliższego skrzyżowania o 100m, a pionowe od 77m do 101m. Zrezygnowano z pełnej symetrii sieci, która w rzeczywistych sieciach nie występuje. Progi generowania pojazdów dla wjazdów pionowych ustawiono na 0,6, natomiast dla wjazdów poziomych 0,8.



Rys. 36: Infrastruktura arterii

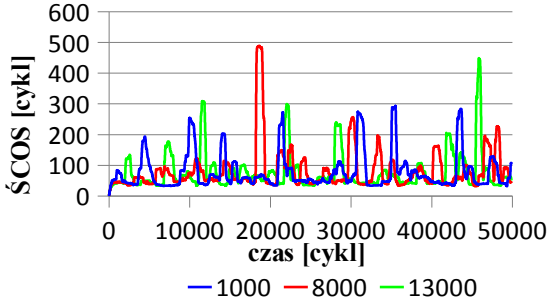
10.3.2. Wyniki badań dla generatora ruchu według rozkładu Fatigue Life

Dla danej infrastruktury w warunkach ruchu stworzonych przez generator ruchu według rozkładu Fatigue Life najkrótsze ŚCOS (Tab. 23) oraz największe liczby pojazdów osiągających cel podróży (Tab. 24) uzyskano podczas sterowania za pomocą algorytmu TC-1 Bucket 2.0. Algorytm ten pozwalał uzyskać również dobre wartości chwilowych ŚCOS poza jednym wypadkiem wystąpienia zatoru (Rys. 37).



Rys. 37: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem TC-1 Bucket 2.0

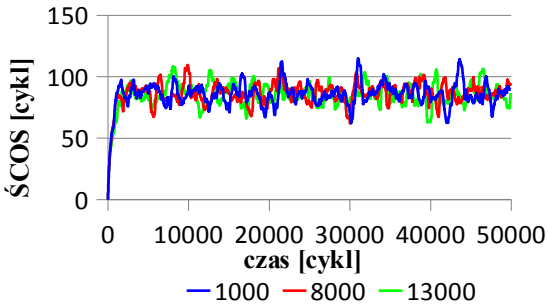
Drugim algorytmem pod względem efektywności okazał się algorytm autorski. Jednak liczba pojazdów



Rys. 38: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=2 f=2 rb=0,0)

osiągających cel podróży uzyskana za pomocą sterowania algorytmem autorskim stanowi ok. 80% liczby pojazdów przy zastosowaniu algorytmu TC-1 Bucket 2.0. Jedynym mankamentem tego algorytmu jest jego czas wykonania, który w tym przypadku ponad 42 razy dłuższy niż czas wykonania algorytmu Most Cars oraz niespełna 30 razy dłuższy od algorytmu aturskiego (Tab. 25).

Chociaż autorski algorytm pozwalał uzyskać ŚCOS poniżej wartości 80 cykli dla parametrów *wtt*=2 i *f*=2, to chwilowe wartości wielokrotnie przekraczały wartość 150 cykli (Rys. 38). Lepsze wartości chwilowe uzyskano dla parametrów *wtt=4* i *f*=4 (Rys. 39). W tych warunkach podobne rezultaty osiągnęły algorytmy Most Cars oraz RL SARSA 5. Słabo wykazały się natomiast algorytmy Local Hill-Climbing oraz GenNeural.



Rys. 39: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=4 f=4 rb=0,02)

Algorytm	Średni czas oczekiwania przed skrzyżowaniami ŚCOS [cykl]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	146,34	146,77	145,02
Local Hill-Climbing	64,43	108,51	94,18
TC-1 Bucket 2.0	58,42	58,31	64,96
RL SARSA 5	138,88	137,19	139,29
GenNeural	375,22	158,11	224,59
In-and-Outbound Lane Control *wtt*=2 *f*=2 *rb*=0,0	76,98	80,83	76,03
In-and-Outbound Lane Control *wtt*=2 *f*=2 *rb*=0,02	82,57	80,65	81,58
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,0	91,74	92,33	92,87
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	88,70	87,64	88,48
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	89,50	89,26	89,39
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	85,12	84,86	84,88
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,02	85,18	85,61	84,79

Tab. 23: Średnie czasy oczekiwania przed skrzyżowaniami uzyskane dla generatora ruchu według rozkładu Fatigue Life

Algorytm	Liczba pojazdów, które dotarły do celu [pojazd]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	125314	124996	125409
Local Hill-Climbing	99583	99734	100018
TC-1 Bucket 2.0	182714	182622	176482
RL SARSA 5	124120	124885	123984
GenNeural	72634	91981	82370
In-and-Outbound Lane Control *wtt*=2 *f*=2 *rb*=0,0	145185	145500	144990
In-and-Outbound Lane Control *wtt*=2 *f*=2 *rb*=0,02	143892	143998	143910
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,0	141127	140458	140402
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	142221	143063	142656
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	141893	142227	142130
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	141191	141742	141686
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,02	141561	141333	142028

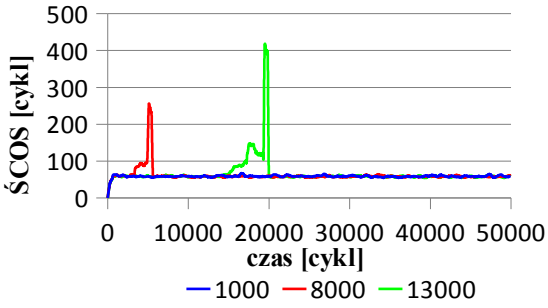
Tab. 24: Liczba pojazdów, które dotarły do celu podczas trwania symulacji dla generatora ruchu według rozkładu Fatigue Life

Algorytm	Czas wykonania algorytmu [ns]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	60341	60053	59223
Local Hill-Climbing	1683304	1729458	1508949
TC-1 Bucket 2.0	2578004	2568267	2628568
RL SARSA 5	120206	120408	120254
GenNeural	219372	190470	217100
In-and-Outbound Lane Control *wtt*=2 *f*=2 *rb*=0,0	90765	93001	88947
In-and-Outbound Lane Control *wtt*=2 *f*=2 *rb*=0,02	89119	85052	82586
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,0	90072	89343	89925
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	90696	88846	88448
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	90140	91136	90988
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	93190	89077	88635
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,02	88018	89406	92236

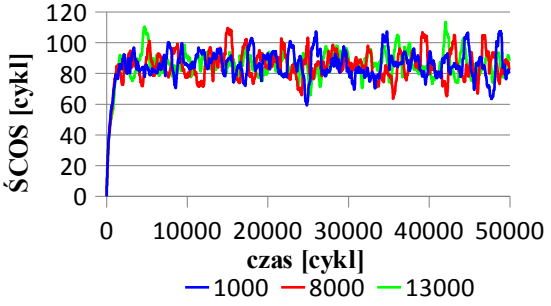
Tab. 25: Czasy wykonania algorytmów dla generatora ruchu według rozkładu Fatigue Life

10.3.3. Wyniki badań dla generatora ruchu według rozkładu Lognormal

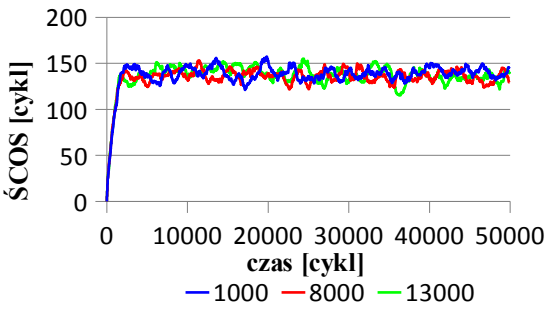
W warunkach ruchu stworzonych przez generator ruchu o rozkładzie Lognormal najlepsze efekty sterowania osiągnięto za pomocą algorytmu TC-1 Bucket 2.0. W trakcie sterowania tym algorytmem wystąpiły dwa zatory, które zostały rozładowane (Rys. 40). Wyniki przeprowadzonych symulacji wykazały najkrótszy ŚCOS w przypadku sterowania za pomocą algorytmu Local Hill-Climbing równy 15,06 cykla (Tab. 26). Jednak liczba pojazdów osiągających cel podróży w tym przypadku stanowi 55% liczby pojazdów docierających do celu w przypadku zastosowania algorytmu TC-1 Bucket 2.0 i 70% przy zastosowaniu algorytmu autorskiego (Tab. 27). Potwierdza to fakt faworyzowania przez algorytm Local Hill-Climbing pewnych kierunków ruchu, co nie było z góry zamierzone. Drugim z kolei pod względem efektywności okazał się algorytm autorski wyprzedzając algorytm Most Cars. Chwilowe ŚCOS nie wiele przekraczały 100 cykli. Nie wiele gorsze rezultaty w porównaniu do algorytmu Most Cars osiągnięto podczas sterowanie za pomocą algorytmu RL SARSA 5. Cechą charakterystyczną sterowania za pomocą tego algorytmu jest stały ŚCOS na poziomie 150 cykli (Rys. 42). Algorytm ten odbiega jednak od algorytmu autorskiego i Most Cars pod względem czasu wykonania (Tab. 28). Czas wykonania jest też mankamentem algorytmu TC-1 Bucket 2.0, który w porównaniu do autorskiego w tej sieci potrzebował trzydzieści razy więcej czasu.



Rys. 40: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem TC-1 Bucket 2.0



Rys. 41: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=4 f=4 rb=0,02)



Rys. 42: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem RL SARSA 5

Algorytm	Średni czas oczekiwania przed skrzyżowaniami ŚCOS [cykl]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	132,82	132,09	131,48
Local Hill-Climbing	15,06	73,65	81,84
TC-1 Bucket 2.0	57,93	60,40	65,17
RL SARSA 5	136,73	133,84	135,75
GenNeural	404,51	157,37	229,34
In-and-Outbound Lane Control *wtt*=2 *f*=2 *rb*=0,0	74,73	74,13	71,19
In-and-Outbound Lane Control *wtt*=2 *f*=2 *rb*=0,02	76,23	76,08	76,71
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,0	80,98	82,22	82,17
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	83,33	83,89	83,66
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	82,87	82,44	82,59
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	83,12	83,87	82,95
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	84,00	84,70	84,15

Tab. 26: Średnie czasy oczekiwania przed skrzyżowaniami uzyskane dla generatora ruchu według rozkładu Lognormal

Algorytm	Liczba pojazdów, które dotarły do celu [pojazd]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	127229	126983	127332
Local Hill-Climbing	100211	100215	100455
TC-1 Bucket 2.0	182895	179299	175563
RL SARSA 5	119909	121097	120104
GenNeural	68680	89734	77698
In-and-Outbound Lane Control *wtt*=2 *f*=2 *rb*=0,0	142762	142420	142878
In-and-Outbound Lane Control *wtt*=2 *f*=2 *rb*=0,02	142208	142361	142261
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,0	142029	141271	141578
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	140526	139926	140889
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	139411	139949	139978
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	140249	139376	139553
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	137524	137318	137662

Tab. 27: Liczba pojazdów, które dotarły do celu podczas trwania symulacji dla generatora ruchu według rozkładu Lognormal

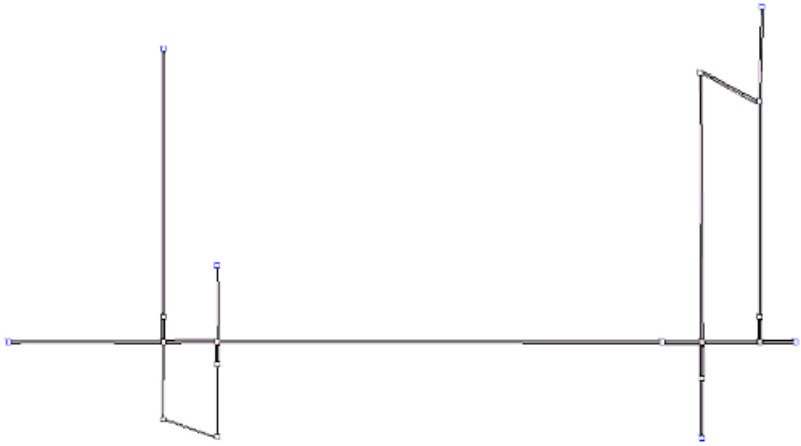
Algorytm	Czas wykonania algorytmu [ns]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	58030	58006	57140
Local Hill-Climbing	1531981	1627593	1745326
TC-1 Bucket 2.0	2601593	2662000	2594122
RL SARSA 5	118985	119442	118784
GenNeural	216952	187405	208188
In-and-Outbound Lane Control *wtt*=2 *f*=2 *rb*=0,0	88407	85893	82061
In-and-Outbound Lane Control *wtt*=2 *f*=2 *rb*=0,02	82426	83791	84005
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,0	89048	85879	85001
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	86513	85571	85168
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	85287	85631	85346
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	85697	86336	85548
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	89033	91830	91893

Tab. 28: Czasy wykonania algorytmów dla generatora ruchu według rozkładu Lognormal

10.4. Fragment Sieci Miasta Krapkowice

10.4.1. Badana infrastruktura

W aplikacji GLD zamodelowano fragment sieci drogowej miasta Krapkowice, który zobrazowano na Rys. 23 (str. 93). Wymiary odcinków dróg sieci przeliczono na podstawie rzeczywistych map miasta Krapkowice według przeliczników zawartych w rozdziale 7.1.2. W wyniku otrzymano infrastrukturę przedstawioną na Rys. 43.

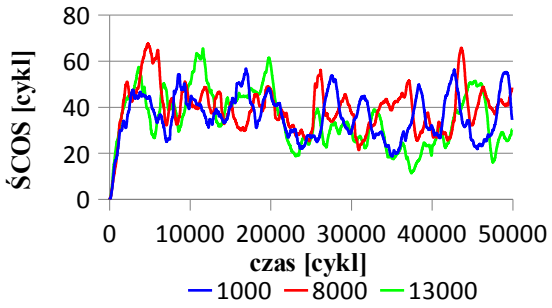


Rys. 43: Badana infrastruktura fragmentu sieci drogowej miasta Krapkowice

Progi generowania pojazdów ustawiono zgodnie z wartościami względnymi przedstawionymi w Tab. 1 na str. 94.

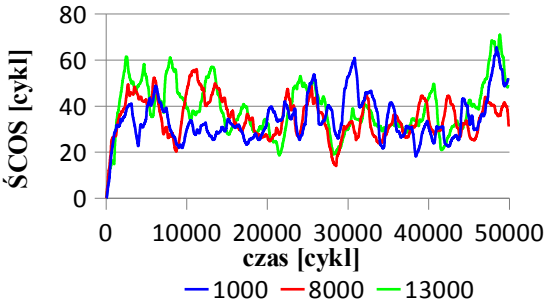
10.4.2. Wyniki badań dla generatora ruchu według rozkładu Burr (4P)

W warunkach ruchu drogowego, jakie stworzył generator ruchu Burr (4P) najlepiej ze sterowaniem ruchu radził sobie najprostszy algorytm Most Cars. Przy pomocy tego algorytmu uzyskano nie tylko krótkie średnie czasy oczekiwania w porównaniu do pozostałych algorytmów (Tab. 29), ale również największą ilość pojazdów, które w trakcie trwania symulacji osiągnęły cel swojej podróży (Tab. 30). Autorski algorytm, który jest pochodnym algorytmu Most Cars osiągał minimalnie krótsze średnie czasy oczekiwania pojazdów przed skrzyżowaniami, jednak liczba pojazdów osiągających cel podróży była mniejsza nawet o 10000 pojazdów. Stanowi to ok. 12% wszystkich pojazdów osiągających cel przy zastosowaniu autorskiego algorytmu.



Rys. 44: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem Most Cars

Pod względem chwilowych wartości średnich czasów oczekiwania oba algorytmy osiągały podobne rezultaty (Rys. 44 i Rys. 45). Pod względem czasu wykonania algorytm Most Cars przewyższa w niewielkim stopniu autorski algorytm (Tab. 31). Rezultaty osiągnięte przez pozostałe algorytmy odbiegają znacznie od rezultatów algorytmów wymienionych powyżej.



Rys. 45: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=2, f=5 rb=0,0)

Okazuję się również, że w tych warunkach wartość parametru *rb* miała niewielki wpływ na efektywność algorytmu autorskiego dla parametrów *wtt*=2 i *f*=5 lub *f*=6.

Algorytm	Średni czas oczekiwania przed skrzyżowaniami [cykl]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	35,69	38,82	34,31
Local Hill-Climbing	160,68	191,57	153,85
TC-1 Bucket 2.0	133,45	105,49	158,91
RL SARSA 5	61,97	60,27	57,85
GenNeural	239,13	208,39	243,99
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	35,08	38,59	38,91
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	34,51	38,73	33,88
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	38,61	37,24	33,81
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	33,76	34,38	38,22
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	36,93	35,16	37,17
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	32,99	36,33	32,11
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	40,98	38,14	35,12

Tab. 29: Średnie czasy oczekiwania przed skrzyżowaniami uzyskane dla generatora ruchu według rozkładu Burr (4P)

Algorytm	Liczba pojazdów, które dotarły do celu [pojazd]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	91268	87910	91418
Local Hill-Climbing	54554	53756	61062
TC-1 Bucket 2.0	69780	73959	58890
RL SARSA 5	70251	71053	73533
GenNeural	38067	42568	41340
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	80114	76825	74754
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	80470	80607	81214
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	80260	79919	84266
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	82399	81486	80708
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	80022	80953	80240
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	82548	81292	81773
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	80199	79142	81843

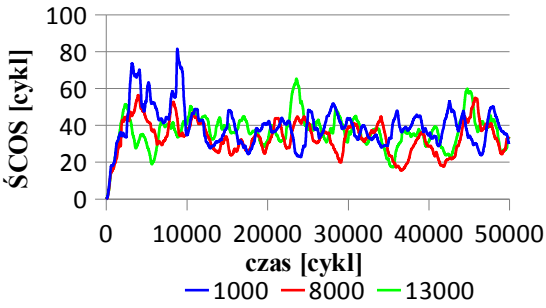
Tab. 30: Liczba pojazdów, które dotarły do celu podczas trwania symulacji dla generatora ruchu według rozkładu Burr (4P)

Algorytm	Czas wykonania algorytmu [ns]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	47809	48479	44288
Local Hill-Climbing	842088	866821	816258
TC-1 Bucket 2.0	2484928	1728699	2983069
RL SARSA 5	74019	71758	70705
GenNeural	171188	177221	169492
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	50356	52620	50251
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	52853	54019	50289
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	53225	50518	49216
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	53241	53663	52482
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	53791	51667	52512
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	51416	52565	49535
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	55110	51862	50380

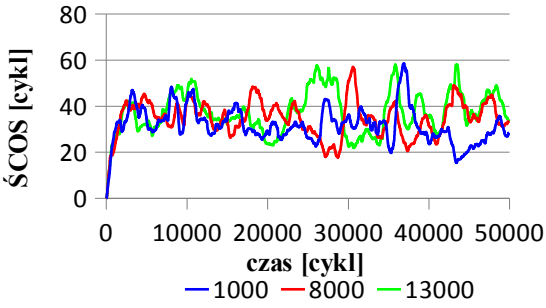
Tab. 31: Czasy wykonania algorytmów dla generatora ruchu według rozkładu Burr (4P)

10.4.3. Wyniki badań dla generatora ruchu według rozkładu inv. Gaussian (3P)

Podobnie jak w warunkach ruchu stworzonych za pomocą generatora ruchu według rozkładu Burr (4P) w warunkach, kiedy ruch generowany był według rozkładu odwrotnego Gaussa z trzema parametrami (inv. Gaussian 3P) najlepsze rezultaty osiągnięto stosując algorytm Most Cars. Również w tym przypadku liczba pojazdów osiągających cel podróży przekraczała 90000 pojazdów (Tab. 34), przy średnim czasie oczekiwania przed skrzyżowaniami równym 33,03 cykla (Tab. 33). Chociaż autorski algorytm osiągał niewiele krótsze średnie czasy oczekiwania przed skrzyżowaniami, to



Rys. 46: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem Most Cars



Rys. 47: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=2, f=6 rb=0,02)

jednak liczba pojazdów osiągających cel podróży wynosiła maksymalnie 82885 pojazdów, co stanowi znaczną różnicę w porównaniu do algorytmu Most Cars. Czasy wykonania algorytmu Most Cars stanowiły ok. 92% czasu wykonania algorytmu autorskiego (Tab. 34). Atutem autorskiego algorytmu w tych warunkach były chwilowe średnie czasy oczekiwania pojazdów przed skrzyżowaniami nie przekraczające wartości 60 cykli (Rys. 46), czego w przypadku algorytmu Most Cars nie udało się osiągnąć (Rys. 47). Duże chwilowe ŚCOS wskazują na dłuższe przetrzymywanie pojazdów przed skrzyżowaniem z pewnych pasów wjazdowych, a następnie przydzielenie sygnału zielonego dla tego pasa i rozładowanie kolejki. Spośród bardziej złożonych czasowo algorytmów najlepsze rezultaty osiągnięto podczas sterowania ruchem za pomocą algorytmu RL SARSA 5. Jednak w porównaniu do algorytmu Most Cars oraz algorytmu autorskiego efektywność tego algorytmu nie jest zachęcająca do jego stosowania w tych warunkach. Algorytmy Local Hill-Climbing oraz GenNeural wyjątkowo nie radziły sobie w tej sieci i w tych warunkach ze sterowaniem ruchu.

Algorytm	**Średni czas oczekiwania przed skrzyżowaniami [cykl]**		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	39,10	33,03	36,23
Local Hill-Climbing	243,97	174,94	200,23
TC-1 Bucket 2.0	134,09	125,66	185,53
RL SARSA 5	55,80	57,21	60,36
GenNeural	270,75	196,67	248,55
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	38,86	41,93	36,89
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	32,54	34,86	37,10
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	35,12	39,27	37,45
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	35,59	35,50	35,19
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	38,13	36,62	36,39
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	35,97	35,32	34,99
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	31,15	34,16	36,82

Tab. 32: Średnie czasy oczekiwania przed skrzyżowaniami uzyskane dla generatora ruchu według rozkładu inv. Gaussian (3P)

Algorytm	Liczba pojazdów, które dotarły do celu [pojazd]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	88674	90942	90467
Local Hill-Climbing	48305	51202	54598
TC-1 Bucket 2.0	69583	70045	55792
RL SARSA 5	71742	71678	71988
GenNeural	35387	42153	43663
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	79927	75926	76504
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	83898	80461	79351
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	81136	78265	79521
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	81576	79878	79429
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	80513	79596	82346
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	82236	80349	83019
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	82885	81392	80135

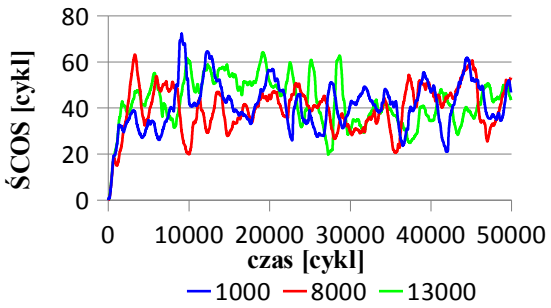
Tab. 33: Liczba pojazdów, które dotarły do celu podczas trwania symulacji dla generatora ruchu według rozkładu inv. Gaussian (3P)

Algorytm	Czas wykonania algorytmu [ns]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	46783	46209	45592
Local Hill-Climbing	922141	877781	869309
TC-1 Bucket 2.0	2085914	2417329	3218187
RL SARSA 5	79268	75524	74748
GenNeural	174544	167784	169219
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	51406	51256	50775
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	51297	51068	53011
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	52742	55168	53701
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	51379	49956	52863
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	54344	51417	47912
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	54535	53714	52411
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	52805	53897	55792

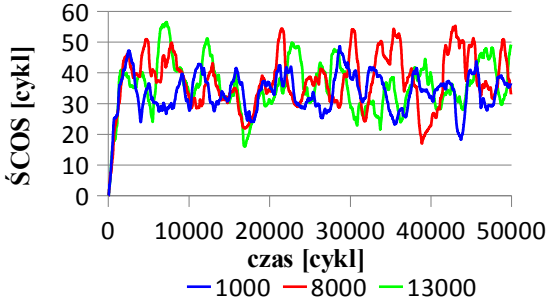
Tab. 34: Czasy wykonania algorytmów dla generatora ruchu według rozkładu inv. Gaussian (3P)

10.4.4. Wyniki badań dla generatora ruchu według rozkładu Pearson 5

We fragmencie sieci drogowej Krapkowic w warunkach ruchu stworzonych przez generator ruchu o rozkładzie Pearson 5 najkrótsze średnie czasy oczekiwania osiągnięto przy zastosowaniu algorytmu autorskiego (Tab. 35). Dla ziarna o wartości 13000 przy sterowaniu ruchem za pomocą algorytmów Most Cars oraz autorskiego uzyskano liczbę pojazdów osiągających cel podróży równą odpowiednio 84251 i 83842 (Tab. 36). Wartości średnich czasów oczekiwania przed skrzyżowaniami wynosiły odpowiednio 42,29 i 32,05 cykla co przemawia na korzyść algorytmu autorskiego jak również chwilowe średnie czasy oczekiwania przed skrzyżowaniami (Rys. 48, Rys. 49 i Rys. 50). Pod względem czasu wykonania oba algorytmy osiągały podobne rezultaty (Tab. 37).

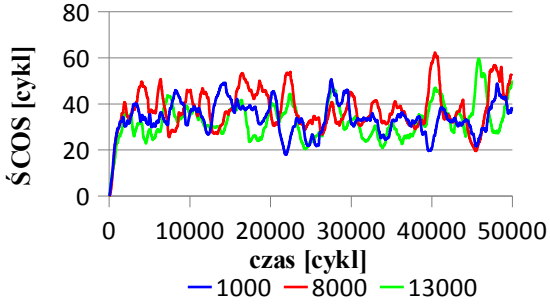


Rys. 48: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem Most Cars



Rys. 49: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=2, f=6 rb=0,0)

Spośród algorytmów bardziej złożonych czasowo najefektywniejszy okazuje się algorytm RL SARSA 5. Jednak również i w tych warunkach rezultaty osiągnięte w trakcie sterowania ruchem za pomocą tego algorytmu odbiegają znacznie od rezultatów osiąganych przy pomocy algorytmu Most Cars i algorytmu autorskiego. Pozostałe algorytmy jak Local Hill-Climbing, GenNeural oraz TC-1 Bucket 2.0 nie osiągnęły zadowalających rezultatów. Zauważono nieznaczący wpływ wartości parametru *rb* algorytmu autorskiego na jego efektywność zarówno dla średnich jak i chwilowych wartości ŚCOS oraz liczby pojazdów osiągająca cel podróży.



Rys. 50: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=2, f=6 rb=0,02)

Algorytm	Średni czas oczekiwania przed skrzyżowaniami [cykl]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	40,63	39,19	42,29
Local Hill-Climbing	157,14	203,17	242,26
TC-1 Bucket 2.0	140,56	134,49	136,58
RL SARSA 5	62,85	59,42	60,24
GenNeural	248,81	160,61	259,36
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	38,89	39,08	36,55
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	35,62	37,60	34,80
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	33,60	39,40	37,44
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	33,80	36,71	35,50
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	35,30	36,59	35,89
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	33,06	36,87	35,19
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	33,18	37,22	32,05

Tab. 35: Średnie czasy oczekiwania przed skrzyżowaniami uzyskane dla generatora ruchu według rozkładu Pearson 5

Algorytm	Liczba pojazdów, które dotarły do celu [pojazd]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	87372	87342	84251
Local Hill-Climbing	54871	55215	51440
TC-1 Bucket 2.0	66769	66516	60481
RL SARSA 5	70812	72900	71528
GenNeural	35129	47308	43685
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	82413	79558	77732
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	79614	80222	81032
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	82247	80838	77150
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	81914	81263	79657
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	81167	81489	79470
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	83511	81021	82497
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	82877	80304	83842

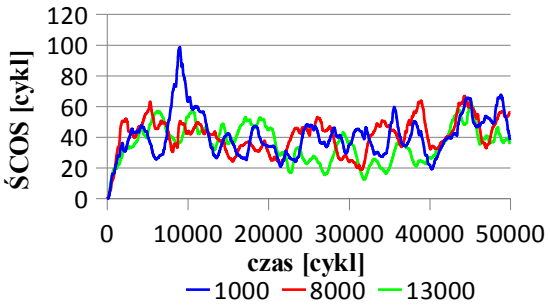
Tab. 36: Liczba pojazdów, które dotarły do celu podczas trwania symulacji dla generatora ruchu według rozkładu Pearson 5

Algorytm	Czas wykonania algorytmu [ns]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	51740	50612	52142
Local Hill-Climbing	781675	789399	803870
TC-1 Bucket 2.0	2250669	2617253	2835846
RL SARSA 5	87804	77760	80842
GenNeural	156204	157907	155571
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	52401	52385	49123
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	51502	54476	50292
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	51962	52786	51196
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	51115	51880	50620
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	52784	51399	50842
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	53202	53449	51177
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	52888	52853	48839

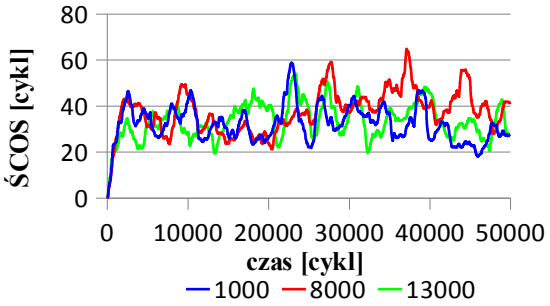
Tab. 37: Czasy wykonania algorytmów dla generatora ruchu według rozkładu Pearson 5

10.4.5. Wyniki badań dla generatora ruchu według rozkładu Weibull (3P)

Warunki ruchu stworzone za pomocą generatora ruchu o rozkładzie Weibull (3P) potwierdzają, że dla fragmentu sieci miasta Krapkowice najefektywniejszym algorytmem jest algorytm Most Cars biorąc pod uwagę średni czas oczekiwania pojazdów przed skrzyżowaniem (Tab. 38) jak również liczbę pojazdów osiągających cel podróży (Tab. 39), której wartość była większa o kilka tysięcy od drugiego w kolejności pod względem efektywności algorytmu autorskiego. Atutem przemawiającym na korzyść autorskiego algorytmu są średnie wartości oraz chwilowe średnie czasy



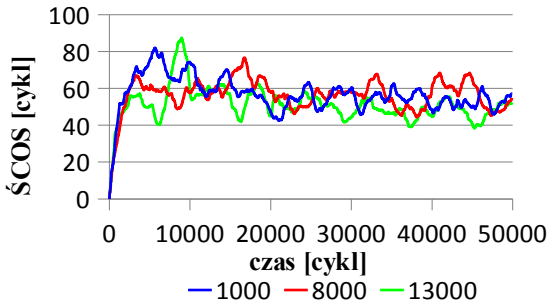
Rys. 51: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem Most Cars



Rys. 52: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=2, f=4 rb=0,0)

oczekiwania, które również i w tych warunkach nie przekraczały wartości 60 cykli (Rys. 51 i Rys. 52). Pod względem czasu wykonania algorytmy Most Cars oraz autorski wypadły podobnie (Tab. 40).

Podobnie jak w innych warunkach dla tej sieci trzecie miejsce zajął algorytm RL SARSA 5. Jednak osiągi tego algorytmu dają wiele do życzenia w porównaniu do wyżej wymienionych algorytmów (Rys. 53). Pozostałe algorytmy jak Local Hill-Climbing, GenNeural oraz TC-1 Bucket 2.0 nie wykazały satysfakcjonujących efektów.



Rys. 53: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem RL SARSA 5

Algorytm	**Średni czas oczekiwania przed skrzyżowaniami [cykl]**		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	40,62	40,55	35,50
Local Hill-Climbing	153,71	188,65	161,44
TC-1 Bucket 2.0	115,04	122,12	148,33
RL SARSA 5	56,27	55,92	50,68
GenNeural	267,22	150,10	241,18
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	37,78	37,81	39,48
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	31,69	37,05	33,11
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	33,94	37,47	35,64
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	34,29	33,39	35,93
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	33,43	33,98	32,66
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	36,17	38,08	33,15
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	35,48	35,38	38,65

Tab. 38: Średnie czasy oczekiwania przed skrzyżowaniami uzyskane dla generatora ruchu według rozkładu Weibull (3P)

Algorytm	Liczba pojazdów, które dotarły do celu [pojazd]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	86449	87202	90472
Local Hill-Climbing	59681	56720	57045
TC-1 Bucket 2.0	73549	69441	66902
RL SARSA 5	72100	71590	73310
GenNeural	34595	48138	43412
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	80951	80508	77885
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	82229	79639	81864
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	80832	79553	81041
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	81271	82697	82402
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	83403	80279	81557
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	81963	79732	82820
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	82111	81603	78832

Tab. 39: Liczba pojazdów, które dotarły do celu podczas trwania symulacji dla generatora ruchu według rozkładu Weibull (3P)

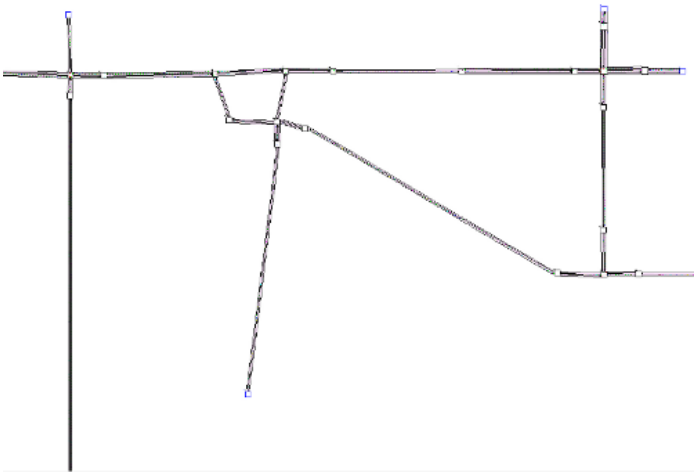
Algorytm	Czas wykonania algorytmu [ns]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	47869	48829	46507
Local Hill-Climbing	782795	829941	825950
TC-1 Bucket 2.0	2016455	1946410	2694558
RL SARSA 5	80225	81987	78548
GenNeural	161342	148998	176659
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	53997	49821	51145
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	53144	57342	55466
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	55397	54372	53586
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	52322	47783	48262
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	50742	52020	49985
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	56988	57245	54297
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	52745	52145	51850

Tab. 40: Czasy wykonania algorytmów dla generatora ruchu według rozkładu Weibull (3P)

10.5. Fragment Sieci Drogowej Miasta Opola

10.5.1. Badana infrastruktura

W aplikacji GLD zamodelowano fragment sieci drogowej miasta Opola, który zobrazowano na Rys. 24 (str. 93).



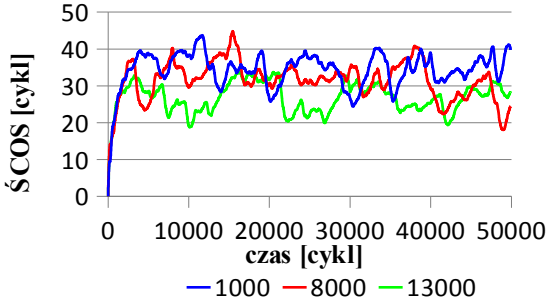
Rys. 54: Badana infrastruktura fragmentu sieci drogowej miasta Opola

Wymiary odcinków dróg sieci przeliczono na podstawie rzeczywistych map miasta Opola według przeliczników zawartych w rozdziale 7.1.2.

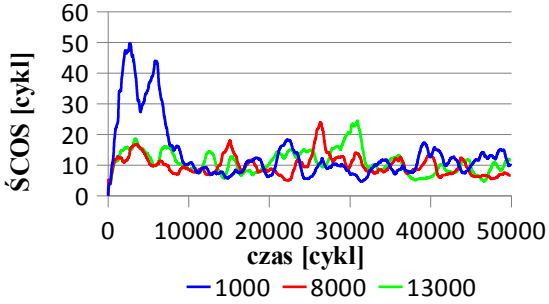
W wyniku otrzymano infrastrukturę przedstawioną na Rys. 54. Progi generowania pojazdów dla poszczególnych wjazdów ustawiono zgodnie z wartościami względnymi w Tab. 2 na str. 94.

10.5.2. Wyniki badań dla generatora ruchu według rozkładu Fatigue Life

Przeprowadzone badania wykazały, że najkrótsze średnie czasy oczekiwania w tych warunkach osiągnięto za pomocą sterowania autorskim algorytmem In-and-Outbound Lane Control. Z pozostałych testowanych algorytmów jak pokazuje Tab. 41 (str. 131) jedynie algorytm RL SARSA 5 osiągnął najbliższe temu średnie czasy oczekiwania z całego przebiegu symulacji. Jednak wartość najkrótszego osiągniętego średniego czasu oczekiwania przed skrzyżowaniem przez autorski algorytm do wartości tego parametru osiągniętego przez algorytm



Rys. 55: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem RL SARSA 5



Rys. 56: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=2 f=6 rb=0,02)

RL SARSA 5 w tych samych warunkach jest trzykrotnie mniejsza. Rys. 55 pokazuje, że algorytm RL SARSA 5 w symulacji nie potrafił zmniejszyć średniego czasu oczekiwania, przyjmującego wartości od początku do końca powyżej 20 cykli. Udało się to osiągnąć przy pomocy algorytmu autorskiego (Rys. 56). Kilkukrotnie większe wartości średniego czasu oczekiwania przed skrzyżowaniem osiągał algorytm TC-1 Bucket 2.0, który podobnie jak autorski algorytm sprawdza zajętość pasa na całej długości. Drugim kryterium porównania algorytmów jest liczba pojazdów, które dotarły do celu podczas trwania symulacji. Według tego kryterium również najlepsze (największe) wartości osiągał autorski algorytm jak pokazuje Tab. 42 Drugim algorytmem według tego kryterium był algorytm Most Cars. W podanych warunkach drogowych słabe rezultaty osiągnął algorytm Local Hill-Climbing, którego czas wykonania był ponad czterdziestokrotnie większy od czasu wykonania autorskiego algorytmu (Tab. 43). W wyniku zastosowania algorytmu GenNeural liczba pojazdów osiągających cel podróży była niemalże dwukrotnie mniejsza od najlepszych osiągów autorskiego algorytmu. Również ŚCOS rzędu 80 cykli dają wiele do życzenia.

Algorytm	**Średni czas oczekiwania przed skrzyżowaniami ŚCOS [cykl]**		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	69,34	45,17	54,24
Local Hill-Climbing	138,51	92,43	109,35
TC-1 Bucket 2.0	88,52	85,74	58,17
RL SARSA 5	33,82	31,01	26,27
GenNeural	97,72	78,89	86,16
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	18,12	10,12	10,42
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	15,50	9,20	8,85
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	15,75	9,26	9,69
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	21,16	9,34	9,70
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	13,24	9,45	10,62
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	16,98	11,28	10,95
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	24,45	11,63	13,28

Tab. 41: Średnie czasy oczekiwania przed skrzyżowaniami uzyskane dla generatora ruchu według rozkładu Fatigue Life

Algorytm	Liczba pojazdów, które dotarły do celu [pojazd]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	105865	113341	107677
Local Hill-Climbing	68382	60856	65256
TC-1 Bucket 2.0	84977	79228	86749
RL SARSA 5	98602	97268	84746
GenNeural	64001	60072	60773
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	123618	119869	119357
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	120918	120695	119664
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	122500	117805	120537
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	122901	118558	118387
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	122750	119186	118676
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	118567	115540	117030
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	122819	116803	115696

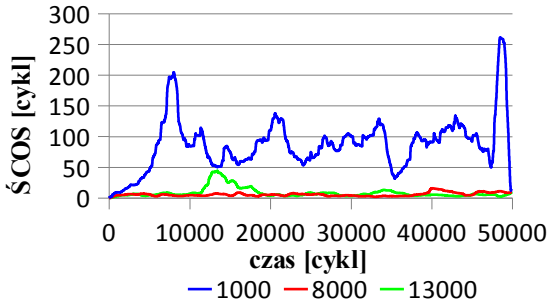
Tab. 42: Liczba pojazdów, które dotarły do celu podczas trwania symulacji dla generatora ruchu według rozkładu Fatigue Life

Algorytm	Czas wykonania algorytmu [ns]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	67891	56950	59558
Local Hill-Climbing	2280565	2159692	2119479
TC-1 Bucket 2.0	1240568	1112351	1009466
RL SARSA 5	89916	84896	71998
GenNeural	148741	121572	132064
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	62447	52485	52777
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	56345	52421	51774
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	59512	54302	52432
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	66466	54959	52474
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	57421	57099	54529
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	58294	53897	54306
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	66530	54515	54466

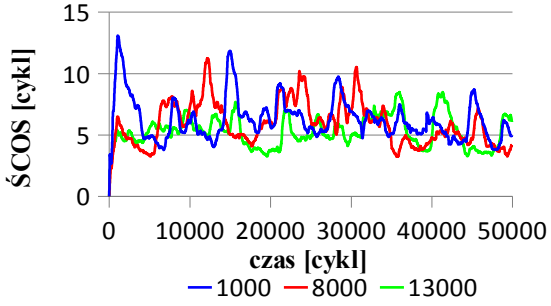
Tab. 43: Czasy wykonania algorytmów dla generatora ruchu według rozkładu Fatigue Life

10.5.3. Wyniki badań dla generatora ruchu według rozkładu Log-Logistic (3P)

W warunkach ruchu stworzonych za pomocą generatora ruchu o rozkładzie logarytmicznie logistycznym z trzema parametrami. Najlepsze osiągi w warunkach, gdy generator zainicjowano ziarnem równym 8000 uzyskał algorytm TC-1 Bucket 2.0 co obrazują Tab. 44 i Tab. 45. Chociaż średni czas oczekiwania pojazdów przed skrzyżowaniem minimalnie lepszy uzyskano za pomocą autorskiego algorytmu, to jednak przy zastosowaniu algorytmu TC-1 Bucket 2.0 liczba pojazdów osiągających cel podróży jest większa o ponad 12000. Na taki wynik może mieć wpływ niewłaściwy dobór fazy ruchu w pewnym momencie, co pociąga za sobą w czasie zablokowanie pewnego strumienia pojazdów. Znaczna różnica jest natomiast w działaniu obu algorytmów dla ziarna równego 1000, co można zobaczyć na Rys. 57 i Rys. 58.



Rys. 57: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem TC-1 Bucket 2.0



Rys. 58: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=2 f=4 rb=0,0)

Dużą liczbę pojazdów osiągających cel uzyskano również przy zastosowaniu algorytmu Most Cars. Średnie czasy oczekiwania przed skrzyżowaniem są jednak kilkukrotnie większe w porównaniu do algorytmu TC-1 Bucket 2.0 i algorytmu autorskiego. Może to świadczyć o tym, że ruch odbywał się falami z dłuższymi przestojami przed skrzyżowaniami. Krótkie średnie czasy oczekiwania przed skrzyżowaniami osiągnął również algorytm RL SARSA 5. Jednak liczba pojazdów osiągających cel jest dużo mniejsza w porównaniu do algorytmów wymienionych powyżej. Może świadczyć to o tym, że pojazdy przed skrzyżowaniem oczekują średnio kilkanaście cykli i zdarza się to przy praktycznie każdorazowym dojechaniu do skrzyżowania. Algorytm ten osiąga długie czasy wykonania względem autorskiego (Tab. 46). Zupełnie nie radziły sobie algorytmy Local Hill-Climbing i GenNeural.

Algorytm	Średni czas oczekiwania przed skrzyżowaniami ŚCOS [cykl]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	48,20	51,31	38,22
Local Hill-Climbing	205,44	89,90	92,71
TC-1 Bucket 2.0	86,48	5,40	8,12
RL SARSA 5	14,50	10,21	10,47
GenNeural	109,62	76,46	95,98
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	5,75	5,40	4,73
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	10,63	5,86	5,46
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	8,99	6,02	5,97
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	18,38	5,32	5,50
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	7,59	5,89	5,79
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	19,27	6,47	6,13
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	15,73	7,01	6,04

Tab. 44: Średnie czasy oczekiwania przed skrzyżowaniami uzyskane dla generatora ruchu według rozkładu Log - Logistic (3P)

Algorytm	Liczba pojazdów, które dotarły do celu [pojazd]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	96696	93164	99392
Local Hill-Climbing	44642	57498	64842
TC-1 Bucket 2.0	72792	111026	111565
RL SARSA 5	83088	74840	69963
GenNeural	55995	50869	48950
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	99203	99724	99366
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	103446	97143	97819
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	103432	97891	97414
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	109089	99841	99860
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	99674	97945	98553
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	107536	95346	95625
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	102521	93905	95036

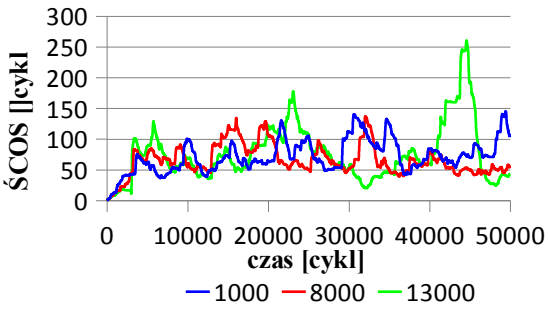
Tab. 45: Liczba pojazdów, które dotarły do celu podczas trwania symulacji dla generatora ruchu według rozkładu Log - Logistic (3P)

Algorytm	Czas wykonania algorytmu [ns]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	58272	55962	46810
Local Hill-Climbing	2415764	2154493	2077147
TC-1 Bucket 2.0	1019342	465049	499396
RL SARSA 5	56190	50658	47022
GenNeural	144597	111177	124549
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	42484	40563	39883
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	45630	40940	41063
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	48486	45572	43354
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	56250	40046	40126
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	42239	41307	40852
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	59352	44063	43555
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	50977	41388	40455

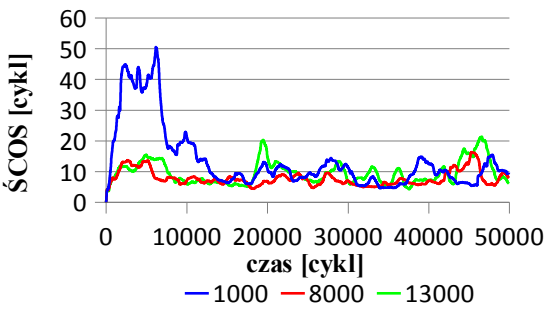
Tab. 46: Czasy wykonania algorytmów dla generatora ruchu według rozkładu Log - Logistic (3P)

10.5.4. Wyniki badań dla generatora ruchu według rozkładu Lognormal

Warunki stworzone w sieci przez zastosowanie generatora ruchu o rozkładzie logarytmicznie normalnym najlepiej odpowiadały algorytmowi autorskiemu, przy pomocy którego średnie czasy oczekiwania pojazdów przed skrzyżowaniami przyjmowały wartości poniżej dziesięciu cykli (Tab. 47). Również przy pomocy tego algorytmu najwięcej pojazdów osiągnęło cel swojej podróży w czasie trwania symulacji (Tab. 48). Dobre wyniki liczby pojazdów osiągających cel podróży uzyskano podczas sterowania ruchem za pomocą algorytmu Most Cars oraz algorytmu



Rys. 59: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem Most Cars



Rys. 60: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=2, f=4 rb=0,0)

RL SARSA 5. Algorytm RL SARSA 5 umożliwił uzyskanie średnich czasów oczekiwania pojazdów na skrzyżowaniu na poziomie 25 cykli. Pod względem tego parametru algorytm Most Cars pozostaje w tyle z czasami w okolicach 70 cykli. Może to świadczyć o braku zdolności tego algorytmu do umożliwienia w sieci płynności ruchu, co charakteryzuje się długimi oczekiwaniami całych kolumn pojazdów (Rys. 59). Ruch taki wydaje się być dalekim od ruchu, jaki występuje przy zastosowaniu zielonej fali. Początkowy wzrost średniego czasu oczekiwania potwierdza również fakt wystąpienia zdarzenia wyjątkowego na pasie drogi. Efektem jest zablokowanie całego pasa na pewien czas. W tym przypadku algorytm Most Cars w przeciwieństwie do algorytmu autorskiego (Rys. 60) nie potrafił w późniejszym czasie zmniejszyć wpływu tego zaburzenia na ruch w sieci. W tych warunkach słabo wypadł algorytm TC-1 Bucket 2.0. Bardzo słabo wypadły algorytmy Local Hill-Climbing oraz GenNeural.

Pod względem czasu wykonania algorytmu (Tab. 49) najmniejsze wartości osiągnięto dla algorytmu autorskiego oraz algorytmu Most Cars.

Algorytm	**Średni czas oczekiwania przed skrzyżowaniami [cykl]**		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	71,67	65,41	75,53
Local Hill-Climbing	120,83	135,79	92,40
TC-1 Bucket 2.0	71,12	54,43	52,73
RL SARSA 5	29,50	25,46	26,48
GenNeural	91,33	87,20	97,26
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	31,03	14,93	12,88
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	13,14	7,27	9,06
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	20,77	8,89	9,39
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	19,18	9,49	9,43
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	28,04	12,67	9,29
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	27,14	11,60	11,62
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	35,64	15,65	10,01

Tab. 47: Średnie czasy oczekiwania przed skrzyżowaniami uzyskane dla generatora ruchu według rozkładu Lognormal

Algorytm	Liczba pojazdów, które dotarły do celu [pojazd]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	104600	105117	95464
Local Hill-Climbing	66908	67710	71719
TC-1 Bucket 2.0	85519	85880	87084
RL SARSA 5	99393	82897	83442
GenNeural	63796	55696	58729
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	124037	117702	118679
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	121569	119474	117817
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	123288	117065	117865
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	122235	118067	117247
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	129937	122200	117219
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	118031	113376	114351
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	123280	114784	114568

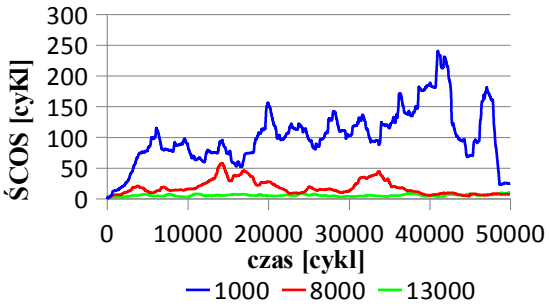
Tab. 48: Liczba pojazdów, które dotarły do celu podczas trwania symulacji dla generatora ruchu według rozkładu Lognormal

Algorytm	Czas wykonania algorytmu [ns]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	74443	67592	67483
Local Hill-Climbing	2350624	2196262	2058664
TC-1 Bucket 2.0	1077156	1006571	992404
RL SARSA 5	83553	70572	71741
GenNeural	142269	122798	126793
In-and-Outbound Lane Control *wtt*=2 *f*=3 *rb*=0,02	70994	55673	52293
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	53742	50085	49828
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	62832	52206	53343
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,02	60152	53355	54108
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	79885	57482	53565
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,0	61469	53393	51928
In-and-Outbound Lane Control *wtt*=3 *f*=4 *rb*=0,02	75481	52166	50481

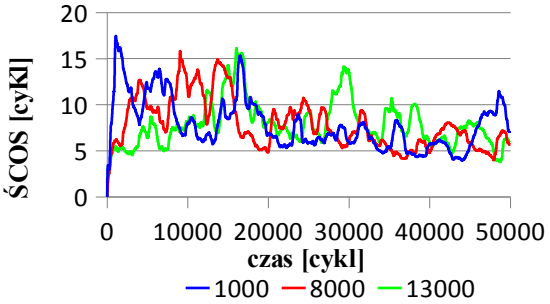
Tab. 49: Czasy wykonania algorytmów dla generatora ruchu według rozkładu Lognormal

10.5.5. Wyniki badań dla generatora ruchu według rozkładu Pearson 6

W kolejnych warunkach ruchu drogowego w sieci stworzonych za pomocą generatora ruchu o rozkładzie Pearson 6 ogólnie dla wszystkich wartości ziarna generatora ruchu najlepiej radził sobie ze sterowaniem ruchu algorytm autorski. Uzyskano przy pomocy sterowania tym algorytmem najkrótsze średnie czasy oczekiwania (Tab. 50). Kolejną korzyścią z zastosowania w tych warunkach autorskiego algorytmu były liczby pojazdów osiągających cel podróży w trakcie trwania symulacji przekraczające 110000 pojazdów (Tab. 51). Dla ziarna o wartości 13000 najlepsze rezultaty osiągnięto sterując ruchem przy pomocy algorytmu TC-1 Bucket 2.0. Jednak algorytm ten dla pozostałych wartości ziaren nie był już taki efektywny jak obrazuje Rys. 61. Chwilowe ŚCOS obliczane w danym cyklu w najwęższym zakresie zmian osiągnięto dla autorskiego algorytmu z parametrami *wtt*=4 *f*=4 *rb*=0,02 (Rys. 62). Duże liczby pojazdów osiągających cel umożliwiło zastosowanie sterowania algorytmem Most Cars. Algorytm ten jednak nie potrafił osiągnąć ŚCOS na poziomie kilkunastu lub mniej cykli. Niskie średnie czasy oczekiwania pojazdów uzyskano przy zastosowaniu algorytmu RL SARSA 5. Jednak liczba pojazdów osiągających cel była nawet o ok. 30000 mniejsza w okresie trwania symulacji w porównaniu do rezultatów algorytmu TC-1 Bucket 2.0 i o ok. 25000 względem osiągów algorytmu autorskiego. W tych warunkach algorytm Most Cars oraz autorski mogą poszczycić się krótkimi czasami wykonania (Tab. 52). Czasy wykonania poniżej 45µs osiągnięto również przy zastosowaniu algorytmu RL SARSA 5 dla ziarna równego 13000. Algorytmy Local Hill-Climbing oraz GenNeural w tej sieci wypadły nie zadowalająco.



Rys. 61: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem TC-1 Bucket 2.0



Rys. 62: Wykres chwilowych ŚCOS uzyskanych podczas sterowania algorytmem autorskim (wtt=4, f=4 rb=0,02)

Algorytm	Średni czas oczekiwania przed skrzyżowaniami [cykl]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	44,34	35,53	42,45
Local Hill-Climbing	154,98	83,22	87,79
TC-1 Bucket 2.0	101,48	17,57	5,18
RL SARSA 5	16,32	10,53	9,46
GenNeural	85,63	81,23	103,64
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	11,34	5,91	5,61
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	7,71	5,32	5,95
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	19,54	5,80	5,26
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	15,87	5,52	5,14
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	7,63	5,43	5,81
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	8,58	6,50	7,61
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,02	7,28	7,36	7,38

Tab. 50: Średnie czasy oczekiwania przed skrzyżowaniami uzyskane dla generatora ruchu według rozkładu Pearson 6

Algorytm	Liczba pojazdów, które dotarły do celu [pojazd]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	100094	98598	97102
Local Hill-Climbing	46440	60552	58862
TC-1 Bucket 2.0	74388	120921	114388
RL SARSA 5	85689	71328	71862
GenNeural	59224	46007	47807
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	103906	98112	98672
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	101068	98053	98089
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	106298	97854	99427
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	112193	98070	99177
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	102297	99026	97386
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	94062	94406	93231
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,02	96251	93394	93133

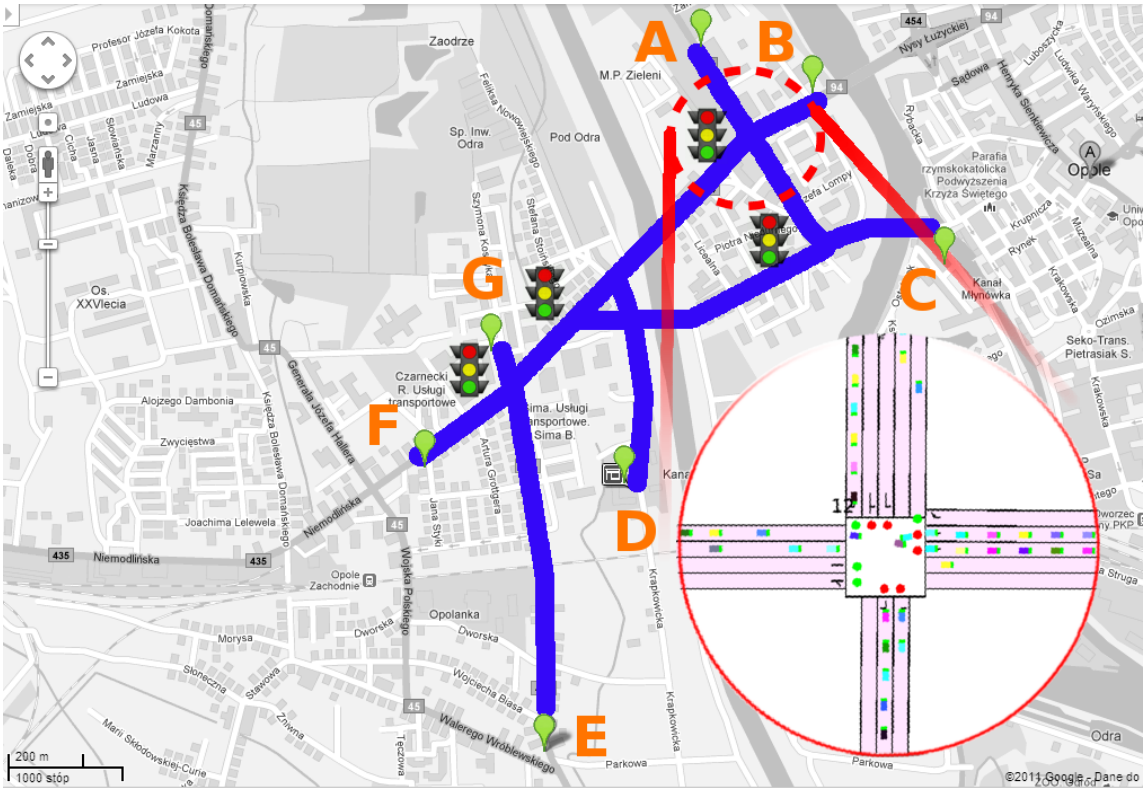
Tab. 51: Liczba pojazdów, które dotarły do celu podczas trwania symulacji dla generatora ruchu według rozkładu Pearson 6

Algorytm	Czas wykonania algorytmu [ns]		
	ziarno 1000	ziarno 8000	ziarno 13000
Most Cars	57805	52568	50929
Local Hill-Climbing	4684697	4168067	4237546
TC-1 Bucket 2.0	1039375	605391	470741
RL SARSA 5	58373	44861	45081
GenNeural	139652	108958	135445
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,0	48076	44043	42200
In-and-Outbound Lane Control *wtt*=2 *f*=4 *rb*=0,02	44990	41925	42349
In-and-Outbound Lane Control *wtt*=2 *f*=5 *rb*=0,0	56364	43393	41176
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,0	55578	42255	41937
In-and-Outbound Lane Control *wtt*=2 *f*=6 *rb*=0,02	44655	40710	40995
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,0	41322	39393	40152
In-and-Outbound Lane Control *wtt*=4 *f*=4 *rb*=0,02	42904	42276	39899

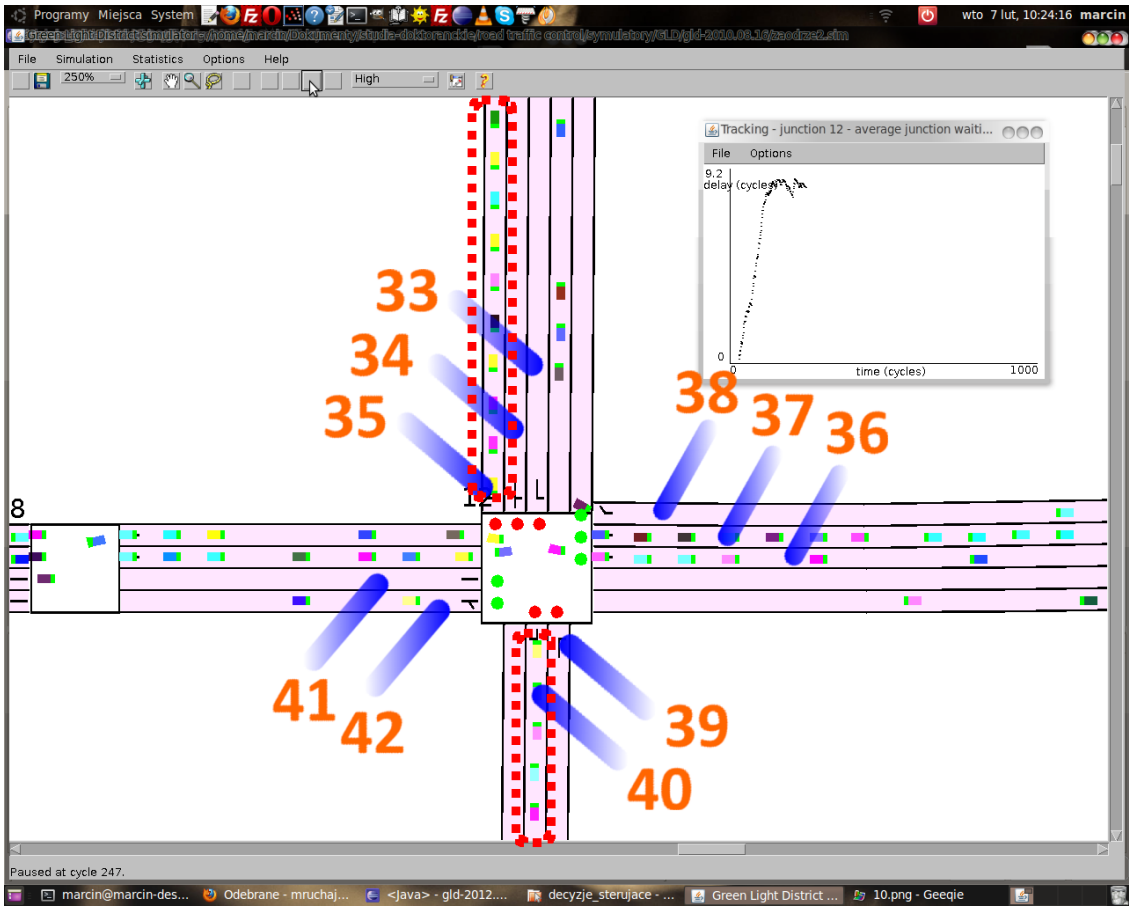
Tab. 52: Czasy wykonania algorytmów dla generatora ruchu według rozkładu Pearson 6

10.6. Wizualizacja Sterowania

Proces sterowania w wykorzystanym środowisku symulacyjnym dzieli się na dwa etapy: obliczenie zysków *q* algorytmami przedstawionymi w Rozdz. 8. oraz wybór odpowiedniej fazy ruchu. W drugim etapie, po wyborze fazy ruchu na podstawie obliczonych zysków, następuje ustawienie sygnalizatorów. Celem obrazowego przedstawienia procesu sterowania wybrano skrzyżowanie ul Wrocławskiej z ul. Nysy Łużyckiej i ul. Niemodlińską, należącego do infrastruktury fragmentu sieci miasta Opola (Rys. 63 [171]). Symulację przeprowadzono dla generatora ruchu Fatigue Life. Na podstawie części symulacji prześledzono poniżej zmiany faz ruchu przebiegające w trakcie następujących po sobie 40 cykli symulacyjnych. Jako pierwszy próbie został poddany algorytm Most Cars, którego działanie śledzono 20 cykli. Następnie włączono autorski algorytm z parametrami *wtt*=2, *f*=5, rb=0,0, którego działanie również śledzono przez kolejne 20 cykli symulacyjnych.



Rys. 63: Skrzyżowanie ul. Wrocławskiej z ul. Nysy Łużyckiej i z ul. Niemodlińską w Opolu



Rys. 64: Wizualizacja sterowania za pomocą algorytmu Most Cars w cyklu nr 247

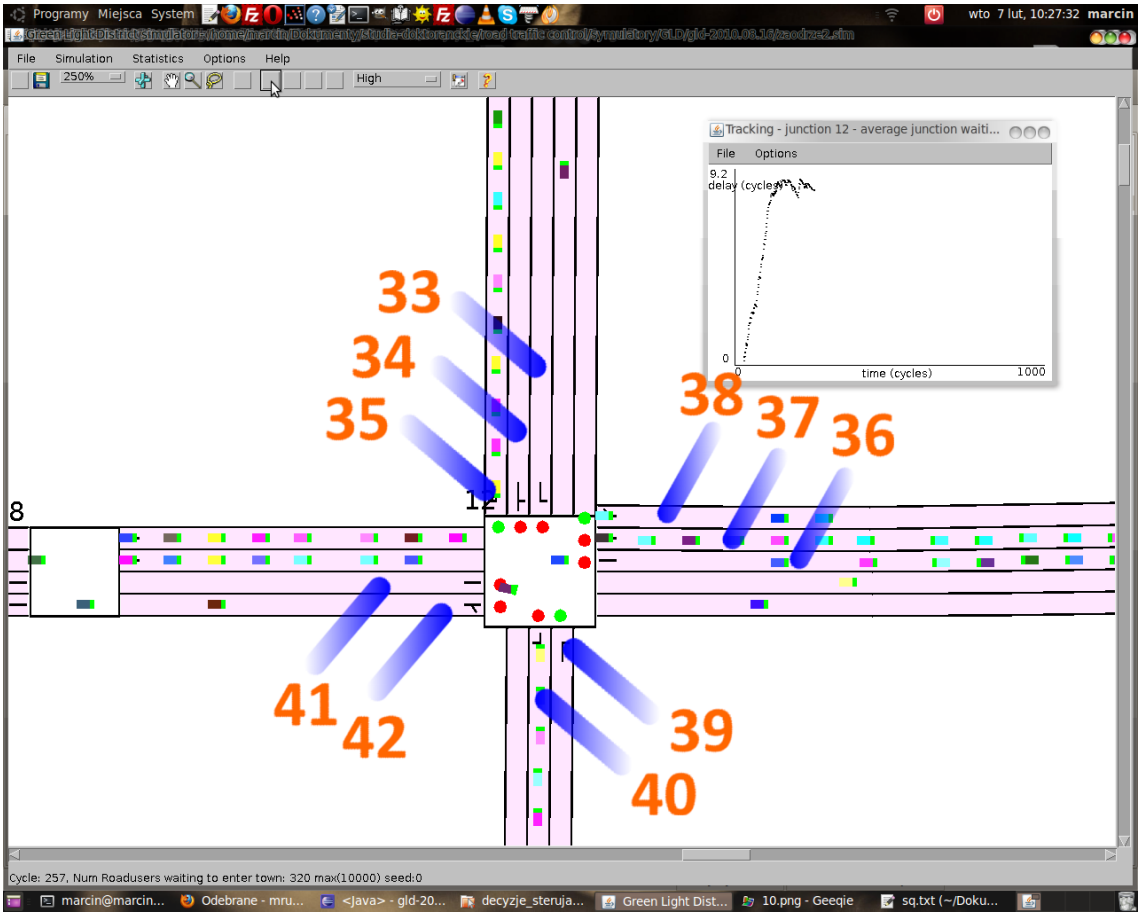
W trakcie sterowania za pomocą algorytmu Most Cars zauważono, że począwszy od cyklu nr 247 pojazdy oczekują na pasach wjazdowych nr 35 i nr 40 do skrzyżowania (oznaczono czerwoną ramką na Rys. 64), nie otrzymując sygnału zielonego.

Algorytm Most Cars nie zmienił fazy ruchu przez siedem kolejnych cykli począwszy od cyklu nr 247, co pokazuje Tab. 53. Pomimo tego, że pasy ruchu nr 35 oraz 40 w tym czasie otrzymywały zysk równy 1, algorytm nie przydzielił fazy ruchu, która dawałaby możliwość przejazdu przez skrzyżowanie z tych pasów.

Nr cyklu	Zysk pasa ruchu										Pasy z sygnałem zielonym
	33	34	35	36	37	38	39	40	41	42	
247	0	0	1	1	1	0	0	1	0	0	36,37,38,41,42
248	0	0	1	1	1	0	0	1	0	1	36,37,38,41,42
249	0	0	1	1	1	0	0	1	0	0	36,37,38,41,42
250	0	0	1	1	1	0	0	1	0	1	36,37,38,41,42
251	0	0	1	1	1	0	0	1	0	0	36,37,38,41,42
252	0	0	1	1	1	0	0	1	0	0	36,37,38,41,42
253	0	0	1	0	1	0	0	1	0	0	36,37,38,41,42
254	0	0	1	1	1	0	0	1	0	0	36,37,38,39
255	0	0	1	1	1	0	0	1	0	0	36,37,38,41,42
256	0	0	1	1	1	1	0	1	1	0	36,37,38,41,42
257	0	0	1	0	1	1	0	1	0	0	35,38,39
258	0	0	1	0	1	0	0	1	0	0	36,37,38,39
259	0	0	1	0	1	0	0	1	0	0	36,37,38,39
260	0	0	1	1	1	1	0	1	0	0	36,37,38,39
261	0	0	1	1	1	1	0	1	0	0	36,37,38,41,42
262	0	0	1	1	1	1	0	1	0	1	36,37,38,41,42
263	0	0	1	1	1	0	0	1	0	0	36,37,38,41,42
264	0	0	1	1	1	0	0	1	0	0	36,37,38,39
265	0	0	1	1	1	0	0	1	0	1	36,37,38,41,42
266	0	0	1	1	1	0	0	1	0	0	36,37,38,41,42

Tab. 53: Wizualizacja sterowania za pomocą algorytmu Most Cars, cykle 247-266

W cyklu nr 254 nastąpiła zmiana fazy ruchu na okres trwania cyklu, jednak pojazdy oczekujące na pasach ruchu nr 35 i nr 40 nadal nie otrzymały sygnału zielonego. W kolejnym cyklu algorytm włączył poprzednią fazę ruchu na dwa cykle. Dopiero w cyklu nr 257 nastąpiło włączenie fazy ruchu uwzględniającej sygnał zielony dla pasa ruchu nr 35 (Rys. 65).

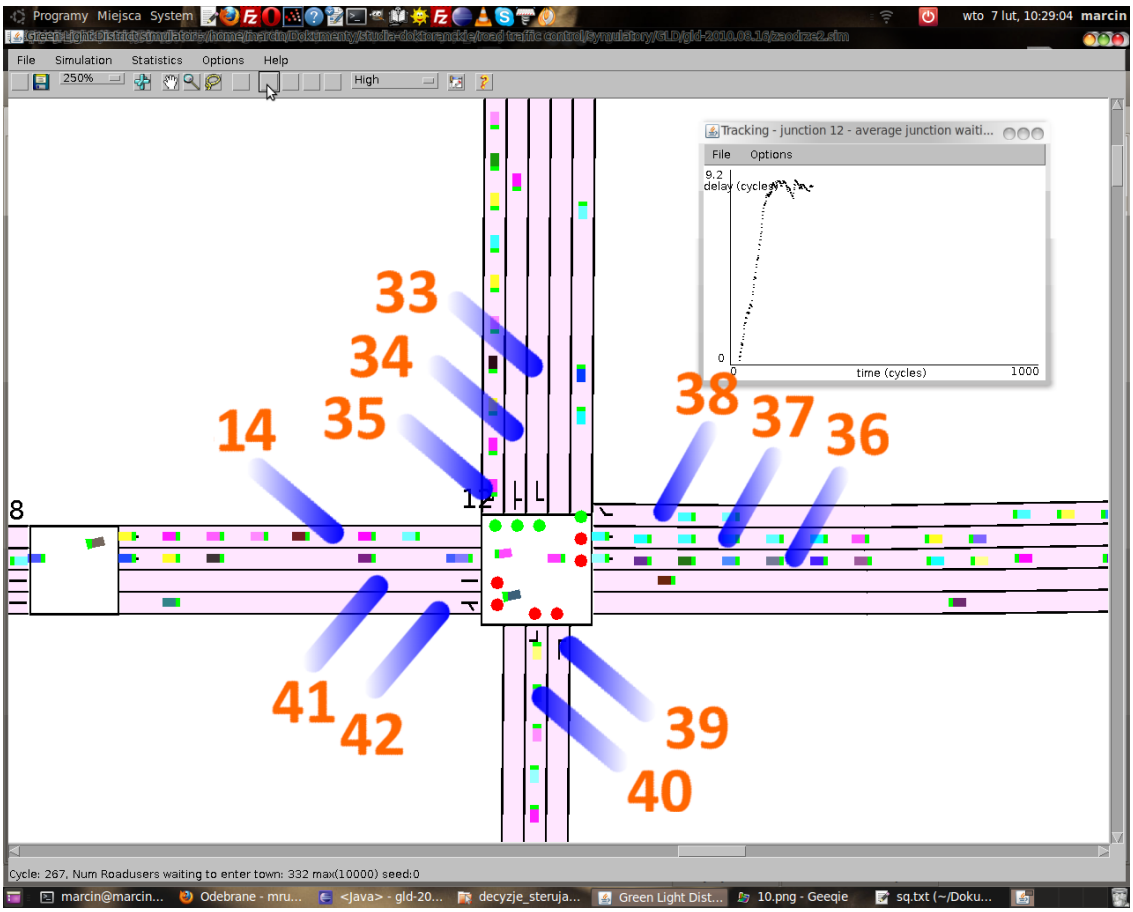


Rys. 65: Wizualizacja sterowania za pomocą algorytmu Most Cars w cyklu nr 257

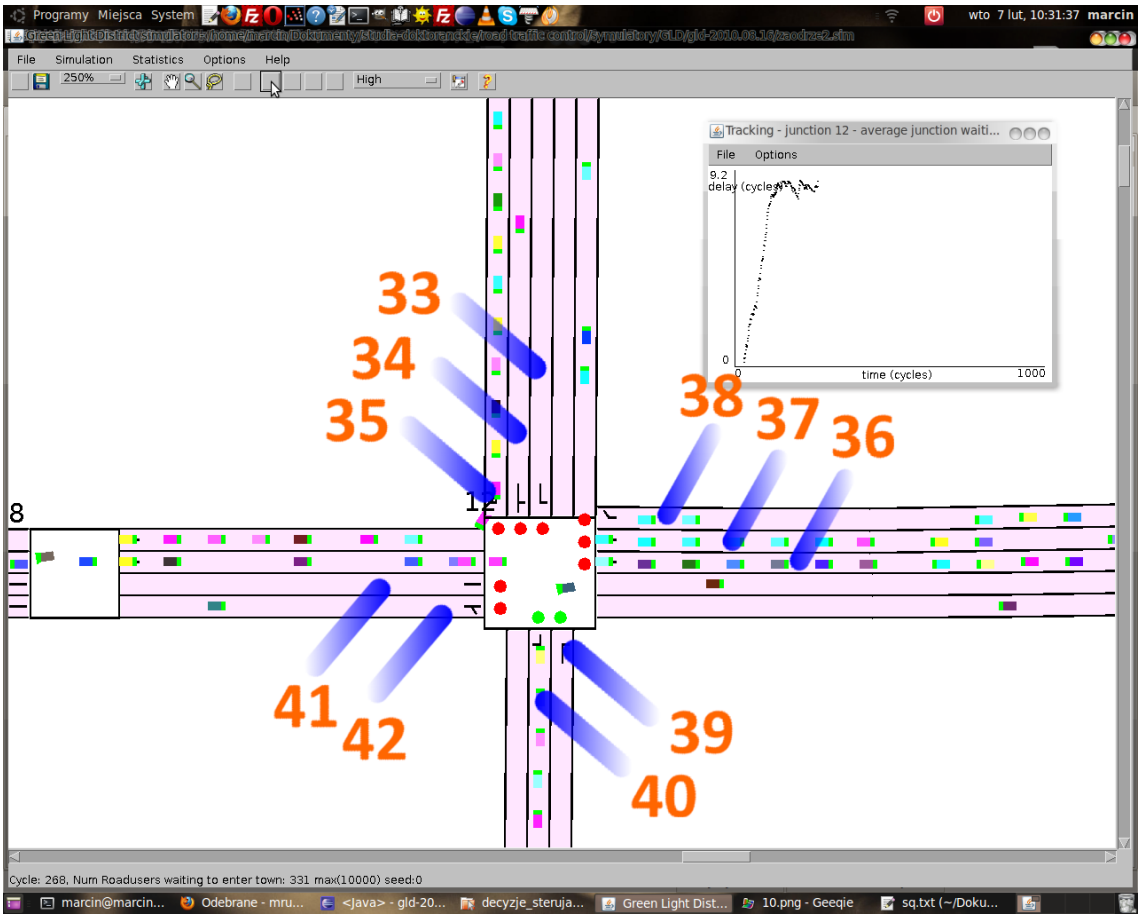
W kolejnych krokach sterowania za pomocą algorytmu Most Cars włącznie z cyklem nr 266 nie został przydzielony sygnał zielony dla pojazdów znajdujących się na pasach ruchu nr 35 i nr 40. W tym momencie włączono algorytm autorski. W pierwszym cyklu sterowania algorytmem autorskim został przydzielony sygnał zielony dla pasa ruchu nr 35, a w następnym cyklu dla pasa ruchu nr 40 (Rys. 66 i Rys. 67). W ciągu czasu równego dziesięciu cyklom pojazdy oczekujące na każdym z jedenastu pasów wjazdowych otrzymały co najmniej trzykrotnie sygnał zielony (Tab. 54). Pas ruchu nr 40 otrzymał w dwóch pierwszych cyklach sterowania autorskim algorytmem zyski równe 5, co świadczy o wolnym miejscu na pasie wyjazdowym ze skrzyżowania nr 14 i o czasie oczekiwania pojazdów przekraczającym wartość *wtt*.

Nr cyklu	Zysk pasa ruchu										Pasy z sygnałem zielonym
	33	34	35	36	37	38	39	40	41	42	
267	0	1	3,24	0,18	0,65	1	0	5	0	1	33,34,35,38
268	0	1	0,18	0,18	0,18	1	0	5	0	1	39,40
269	0	1	0,18	0,18	0,88	1	0	1	0	1	36,37,38,41,42
270	0	1	0,18	0,18	0,88	1	0,92	1	0	0,92	36,37,38,41,42
271	0	1	1,47	0,18	0,29	1	0,92	1	0	0,92	33,34,35,38
272	0	1	0,88	0,29	0,18	1	1	5	0	1	39,40
273	0	1	0,29	0,41	0,06	1	1	1	0	1	36,37,38,39
274	0	1	2,65	0,53	0,88	1	1	1	0	5	35,38,41,42
275	0	5	0,65	0,65	1,47	1	1	1	0	0	33,34,35,38
276	0	0	0,76	0,29	1,47	1	1	5	0	0	39,40

Tab. 54: Wizualizacja sterowania za pomocą autorskiego algorytmu, cykle 267-276



Rys. 66: Wizualizacja sterowania za pomocą autorskiego algorytmu w cyklu nr 267

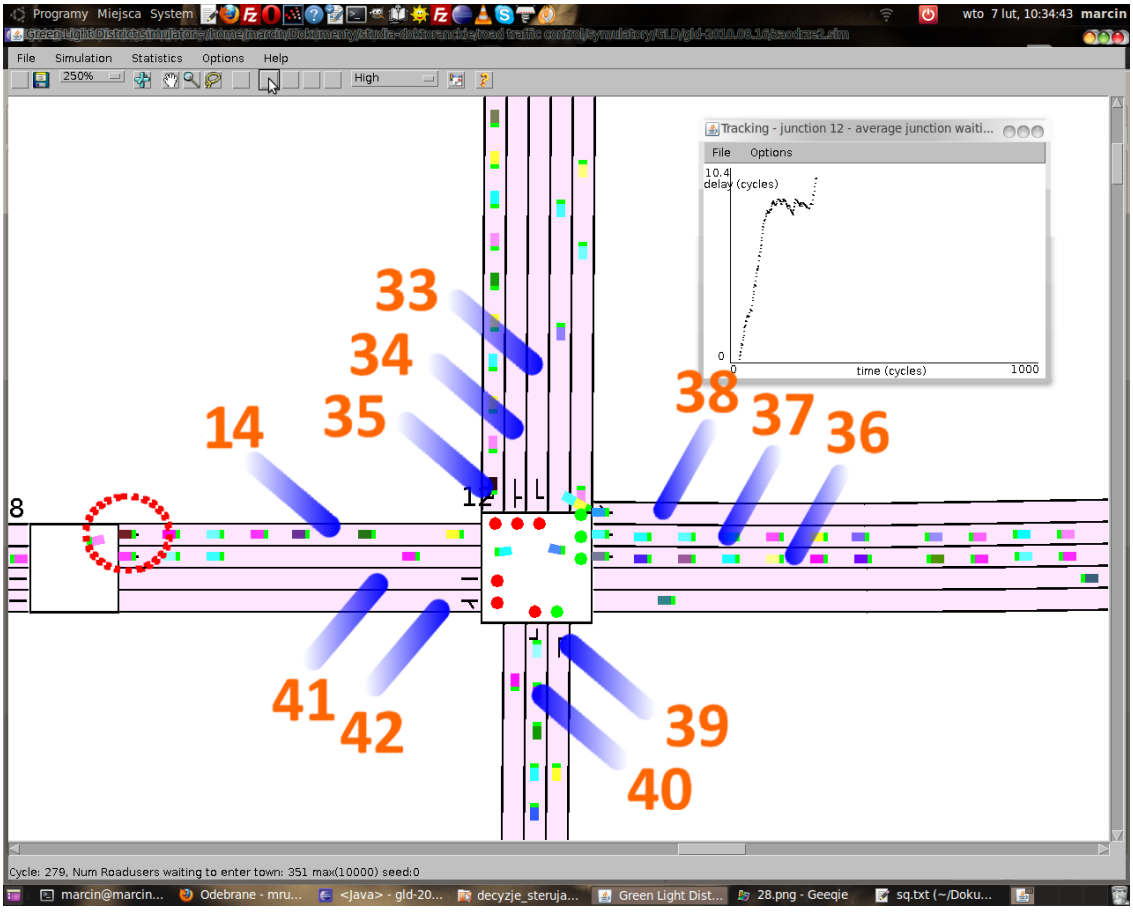


Rys. 67: Wizualizacja sterowania za pomocą autorskiego algorytmu w cyklu nr 268

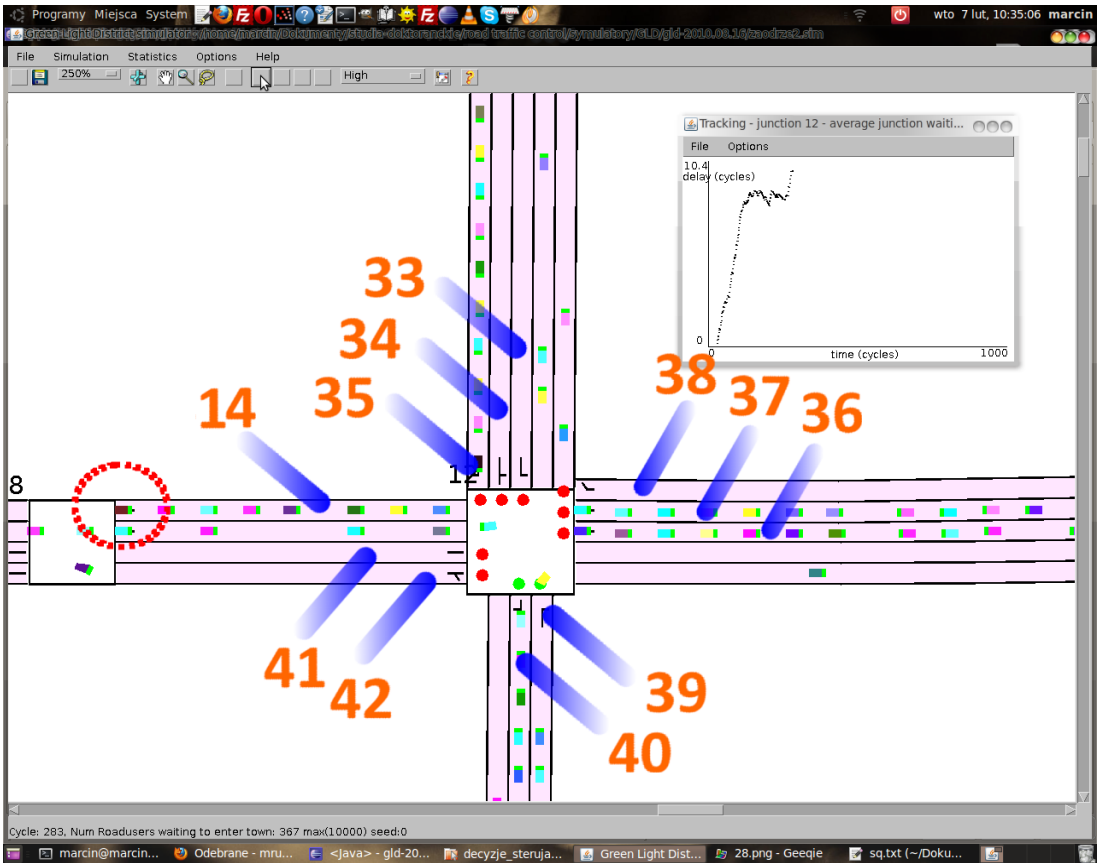
Nr cyklu	Zysk pasa ruchu										Pasy z sygnałem zielonym
	33	34	35	36	37	38	39	40	41	42	
277	0	0	0,88	4,41	4,41	0,86	0,92	0,88	0	0	36,37,38,39
278	0	0	0,06	3,24	0,65	0,93	1	0,65	0	0	36,37,38,39
279	0	0	0,88	0,65	0,65	5	1	0,18	0	0	36,37,38,39,42
280	0	0	3,24	0,18	0,18	0	0,92	3,24	0	0	39,40,38,41,42
281	0	0	2,65	0,06	2,65	0	1	0,29	0	0	36,37,38,39
282	0	0	0,29	0,06	0,65	0	1	3,24	0	1	39,40
283	0	0	0,29	0,29	0,06	0	1	3,82	0	1	39,40,38,39
284	0	0	4,41	4,41	0,18	0	1	0,93	0	1	36,37,38,41,42
285	0	0	4,41	0,88	4,41	0	1	0,93	1	1	36,37,38,41,42
286	0	0	1,47	1	0,29	0	1	0,93	1	1	35,38,41,42

Tab. 55: Wizualizacja sterowania za pomocą autorskiego algorytmu, cykle 278-286

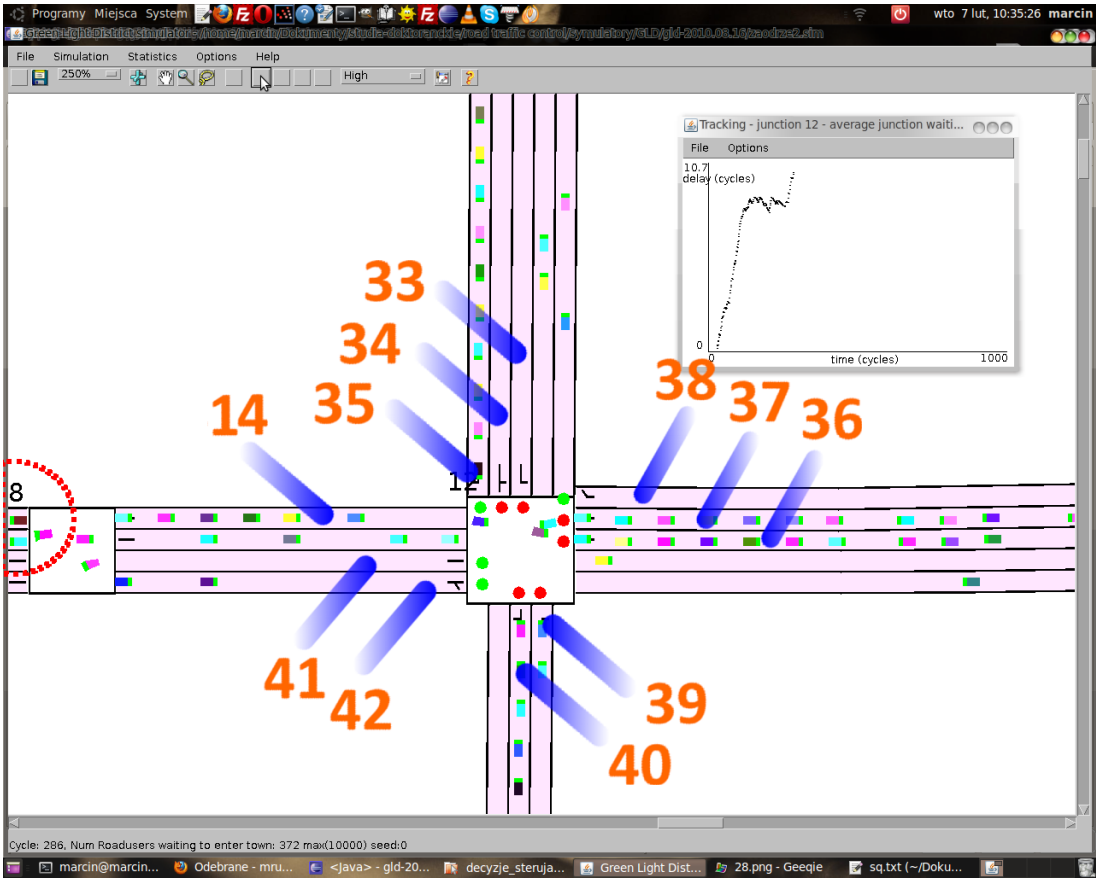
W kolejnych cyklach sterowania autorski algorytm nadal rozładowuje powstałe na pasach wjazdowych kolejki, nie faworyzując wybranych pasów (Tab. 55). W tym okresie pojazdy oczekujące na pasie ruchu nr 35 otrzymały sygnał zielony dopiero w cyklu nr 286. Powodem tego było zablokowanie pasa wyjazdowego nr 14 między 279 a 284 cyklem symulacji. Rys. 68 oraz Rys. 69 pokazują brak ruchu czarnego pojazdu w tym przedziale czasu. Dopiero odblokowanie pasa nr 14 spowodowało przydzielenie sygnału zielonego dla pojazdów oczekujących na pasie nr 35 (Rys. 70).



Rys. 68: Wizualizacja sterowania za pomocą autorskiego algorytmu w cyklu nr 279



Rys. 69: Wizualizacja sterowania za pomocą autorskiego algorytmu w cyklu nr 283



Rys. 70: Wizualizacja sterowania za pomocą autorskiego algorytmu w cyklu nr 286

10.7. Podsumowanie Badań Autorskiego Algorytmu In-and-Outbound Lane Control

Badania przeprowadzone dla odosobnionego skrzyżowania wykazały, że algorytm autorski osiąga najlepsze efekty dla parametrów *wtt*=2, *f*=2 i *rb*=0,0. Okazuje się jednak, że dla innych wartości tych parametrów algorytm osiąga minimalnie gorsze rezultaty. Nasuwa się w tym miejscu wniosek, że decydujące znaczenie ma parametr *wtt*, który wprowadza równouprawnienie dla sygnału zielonego dla każdego wjazdu skrzyżowania. W porównaniu do algorytmów bardziej złożonych czasowo, jak TC-1 Bucket 2.0, RL SARSA 5 oraz GenNeural dużym atutem po stronie autorskiego jest krótki czas wykonywania. Pod względem średnich czasów oczekiwania oraz liczby pojazdów osiągających cel podróży wymienione algorytmy osiągały praktycznie takie same rezultaty. Spośród algorytmów bardziej złożonych czasowo wyróżniał się algorytm Local Hill-Climbing, który wprowadzał faworyzowanie kierunków ruchu. Taka strategia przyniosła jednak negatywny skutek.

Podobnie jak dla odosobnionego skrzyżowania, w arterii algorytm autorski osiągał najlepsze rezultaty dla parametrów *wtt*=2, *f*=2 i *rb*=0,0. W tej infrastrukturze różnica między efektami osiągniętymi przez algorytm autorski i algorytm bazowy Most Cars jest znaczna. Wprowadzenie więc wyższych priorytetów dla pojazdów oczekujących przez czas co najmniej równy *wtt* oraz dla pasów ruchu zajmowanych na całej długości ma duży wpływ na płynność ruchu w takiej sieci. Również uwzględnianie zapełnienia pasów wyjazdowych ma pozytywny wpływ na osiągane wyniki. W takiej infrastrukturze najlepsze rezultaty osiągnął algorytm uczący się TC-1 Bucket 2.0. Cechą charakterystyczną tego algorytmu jest analizowanie ruchu każdego pojazdu łącznie z uwzględnieniem informacji dotyczącej liczby pasażerów jadących w pojazdach. Chociaż algorytm ten był w arterii bezkonkurencyjny, to ilość informacji jaka jest potrzebna do jego realizacji, ogranicza do minimum na dzień dzisiejszy możliwość zastosowania go w rzeczywistych sieciach drogowych. Zaimplementowany system z algorytmem TC-1 Bucket 2.0 wymaga nie tylko odpowiednich detektorów, lecz również odpowiedniego wyposażenia wszystkich pojazdów w sieci objętej sterowaniem w urządzenia wysyłające do systemu wymagane

informacje. Pozostałe algorytmy RL SARSA 5, Local Hill-Climbing oraz GenNeural w porównaniu do autorskiego wypadły nie satysfakcjonująco.

We fragmencie sieci miasta Krapkowice autorski algorytm ustępował algorytmowi macierzystemu Most Cars pod względem liczby pojazdów osiągających cel podróży, jak również nieznacznie pod względem czasu wykonania algorytmu. Atutem algorytmu autorskiego w tej sieci są krótsze średnie czasy oczekiwania pojazdów, zarówno wartości średnie jak i chwilowe. Duże chwilowe ŚCOS oraz duża liczba pojazdów osiągających cel podróży osiągane przez algorytm Most Cars, potwierdzają fakt oczekiwania kolejek przed skrzyżowaniami przez dłuższy czas w trakcie rozładowywania kolejek znajdujących się na innych pasach. Wprowadzenie w algorytmie autorskim parametru *wtt,* ogranicza możliwość występowania tego typu zjawisk. Okazuje się jednak, że w sieci miasta Krapkowice podejście algorytmu Most Cars jest efektywniejsze. Te same rezultaty, tj. ŚCOS i liczbę pojazdów osiągających cel podróży, które osiągnięto przy sterowaniu za pomocą algorytmu Most Cars, można osiągnąć, gdy autorskiemu algorytmowi w wersji 1.1 zostaną nadane parametry *f*=1 i *rb*=0,0. Wynika stąd również konkluzja, że w sieci miasta Krapkowice uwzględnianie zapełnienia pasów wyjazdowych ze skrzyżowań nie ma pozytywnego wpływu na sterowanie ruchem. Sieć krapkowicka charakteryzuje się czterema skrzyżowaniami z sygnalizacją świetlną jak badana arteria w Rozdz. 10.3. Poza tą cechą wspólną sieci różnią się przepływem ruchu i charakterystyką ruchu. W arterii każde skrzyżowanie ma cztery pasy wjazdowe. Ponadto natężenie ruchu na wjazdach pionowych jest znacznie większe w przypadku arterii. W takich warunkach algorytm Most Cars nie radził sobie ze sterowaniem ruchu.

Algorytm	**ŚCOS [cykl]**		**Liczba pojazdów osiągających cel podróży [pojazd]**	
	rb=0,0	*rb*=0,02	*rb*=0,0	*rb*=0,02
In-and-Outbound Lane Control *wtt*=2 *f*=2	50,76	52,16	69504	68768
In-and-Outbound Lane Control *wtt*=2 *f*=3	39,28	38,82	78120	78591
In-and-Outbound Lane Control *wtt*=2 *f*=4	35,62	37,09	80883	80419
In-and-Outbound Lane Control *wtt*=2 *f*=5	35,69	36,19	81223	80919
In-and-Outbound Lane Control *wtt*=2 *f*=6	35,52	36,19	81896	81263
In-and-Outbound Lane Control *wtt*=3 *f*=4	43,06	44,33	76559	75317
In-and-Outbound Lane Control *wtt*=4 *f*=4	50,27	49,09	67540	71878

Tab. 56: Wartości średnie dla wszystkich symulacji dla miasta Krapkowice

Algorytm autorski w wersji 1.2 w tej sieci osiągał ogólnie najlepsze rezultaty dla parametrów *wtt*=2, *f*=6 (Tab. 56). W tej sieci współczynnik losowości nie wpływał znacząco na efektywność algorytmu.

We fragmencie sieci miasta Opola, w której na sześć skrzyżowań cztery są z sygnalizacją świetlną oraz charakteryzuje się drogami z dwoma i trzema pasami ruchu w jednym kierunku algorytm autorski osiągał najlepsze rezultaty spośród przebadanych algorytmów w dziewięciu z dwunastu przebadanych przypadków. W trzech przypadkach efektywniejszym okazał się algorytm uczący się TC-1 Bucket 2.0, który jak wspomniano wcześniej na dzień dzisiejszy nie nadaje się do zastosowań w rzeczywistych sieciach miejskich. Drugim algorytmem spośród algorytmów bardziej złożonych czasowo, który dawał najbliższe rezultaty, osiąganym przez autorski, był algorytm uczący się RL SARSA 5. Pozostałe algorytmy, jak Local Hill-Climbing i GenNeural, nie stanowiły konkurencji dla algorytmu autorskiego.

Dla tej sieci algorytm wykazał największą efektywność dla parametrów o wartości *wtt*=2, *f*=4 i *rb*=0,02 (Tab. 57). Okazuje się, że współczynnik losowości *rb* w tej sieci ma niewielki wpływ na osiągane rezultaty.

Algorytm	**ŚCOS [cykl]**		**Liczba pojazdów osiągających cel podróży [pojazd]**	
	rb=0,0	*rb*=0,02	*rb*=0,0	*rb*=0,02
In-and-Outbound Lane Control *wtt*=2 *f*=2	35,27	33,75	102383	104859
In-and-Outbound Lane Control *wtt*=2 *f*=3	18,38	17,52	110971	110542
In-and-Outbound Lane Control *wtt*=2 *f*=4	12,75	10,39	111288	109723
In-and-Outbound Lane Control *wtt*=2 *f*=5	14,61	12,52	113499	112532
In-and-Outbound Lane Control *wtt*=2 *f*=6	12,66	11,8	112286	110882
In-and-Outbound Lane Control *wtt*=3 *f*=4	13,68	16,49	108217	109807
In-and-Outbound Lane Control *wtt*=4 *f*=4	19,45	17,3	107025	107340

Tab. 57: Wartości średnie dla wszystkich symulacji dla miasta Opola

Czas wykonania algorytmu stanowił kryterium, pod względem którego algorytm ten ustępował w niewielkim stopniu algorytmowi Most Cars.

Badania w obu sieciach drogowych wykazały niewielkie różnice w efektywności działania algorytmu dla parametrów *wtt*=2 i *f*={4,5,6}.

Autorski algorytm In-and-Outbound Lane Control okazuje się być algorytmem o dobrych własnościach sterowania ruchem w zatłoczonej sieci miejskiej. Rozładowuje powstające zatory oraz efektywnie steruje ruchem w przypadku wystąpienia zdarzeń wyjątkowych. Algorytm wspiera możliwie szybkie dotarcie do celu oraz zwiększa przepustowość skrzyżowań rozumianą jako liczbę pojazdów przejeżdżających przez skrzyżowanie w jednostce czasu. Świadczy o tym względnie krótki średni czas oczekiwania przed skrzyżowaniami oraz duża liczba pojazdów osiągających cel podróży.

Wyniki badań przedstawione w niniejszej pracy oraz w [4,163] wykazały większą efektywność autorskiego algorytmu w porównaniu do algorytmu Most Cars zwłaszcza dla sieci o bardziej złożonej strukturze i większym natężeniu ruchu w sieci niż przykładowa sieć miasta Krapkowice.

11. WNIOSKI

- Opracowano nowy algorytm In-and-Outbound Lane Control o mniejszej złożoności obliczeniowej od algorytmów uczenia się ze wzmocnieniem (TC-1 Bucket 2.0, RL SARSA 5), algorytmu przeszukiwania Local Hill-Climbing, algorytmu sieci neuronowej z nadzorem za pomocą algorytmu genetycznego GenNeural. Algorytm autorski rozładowywał zatory w przebadanych drogowych sieciach miejskich, dawał średnie czasy oczekiwania krótsze lub porównywalne jak również umożliwiał osiąganie celu podróży dużej liczbie pojazdów.

  Teza została udowodniona.
- Algorytm autorski sprawdza się w warunkach symulacyjnych w sieciach drogowych o dużym natężeniu ruchu, w których odcinki dróg między skrzyżowaniami są zajmowane przez pojazdy na całej długości. Dobre rezultaty osiągnięto dla sieci przedstawionych w niniejszej rozprawie jak również w [4,163].
- Algorytm autorski osiąga satysfakcjonujące rezultaty w porównaniu do badanych bardziej złożonych algorytmów w różnych warunkach ruchu, również w przypadku wystąpienia zdarzeń wyjątkowych jak wypadki drogowe czy remonty dróg.
- Za pomocą parametrów strojonych algorytm można dopasować do sterowanej sieci drogowej i panujących w niej warunków.
- Zatory w sieciach miejskich na odcinkach dróg między skrzyżowaniami o długościach odpowiadających długości kilku lub kilkunastu pojazdów wpływają na zaburzenie ruchu w większych fragmentach sieci a niekiedy w całej sieci.
- Dalszym kierunkiem rozwoju autorskiego algorytmu może być okresowe automatyczne sprawdzanie jego efektywności i automatyczne dopasowywanie parametrów algorytmu do warunków ruchu.

- Algorytm autorski nadaje się do zrównoleglenia przez to, że do każdego skrzyżowania odnosi się indywidualnie. Jest to kolejna możliwość skrócenia czasu obliczeń.
- Osiągnięte rezultaty zachęcają do sprawdzenia efektywności autorskiego algorytmu w warunkach rzeczywistych.
- Kierunkiem dalszych badań może być sprawdzenie wpływu długości kroku czasowego (cyklu) od infrastruktury sieci drogowej na efektywność działania algorytmów sterowania.
- Metody sterowania ruchem wykorzystujące logikę rozmytą charakteryzują się szybkością obliczeń. Zachęca to do porównania tego typu metod z algorytmem autorskim i innymi badanymi.
- W przeprowadzonych badaniach wszystkie pasy wjazdowe do skrzyżowania traktowano z założenia jako równorzędne. Nie przypisywano z góry wyższych zysków dla określonych kierunków, co jest czynione w przypadku sterowania ruchem w arteriach. Taka modyfikacja traktowania pasów wjazdowych do skrzyżowania warta jest dalszego wysiłku badawczego.

BIBLIOGRAFIA

[1] Komisja Wspólnot Europejskich, Plan działania na rzecz wdrażania inteligentnych systemów transportowych w Europie, KOM (2008) 886 wersja ostateczna, Bruksela, 2008

[2] CEMT/ITF, CONGESTION: A GLOBAL CHALLENGE, The Extent of and Outlook for Congestion in Inland, Maritime and Air Transport, Europejska Konferencja Ministrów Transportu, Sofia, CEMT/ITF (2007) 6, 2007

[3] Wiering M., van Veenen J., Vreeken J., Koopman A., Intelligent Traffic Light Control, Technical Report UU-CS-2004-029, The University of Utrecht, 2004

[4] Ruchaj M., A comparison of selected algorithms to control acyclic traffic lights in urban road networks, Proceedings of the 8th annual Ph.D. workshop WOFEX 2010, VSB - Technical University of Ostrava, 2010, s. 364-369

[5] Serbeńska A., Wdrażanie systemów zarządzania ruchem w polskich miastach cz. II, 2011, http://edroga.pl/inzynieria-ruchu/its/3968-wdrazanie-systemow-zarzadzania-ruchem-w-polskich-miastach-cz-ii

[6] Gordon R. L., Tighe W. P. E., Traffic control systems handbook, Federal Highway Administration, Dunn Engineering Associates, P.C., Westhampton Beach, 2005

[7] Gartner N. H., OPAC: A Demand-Responsive Strategy for Traffic Signal Control, Journal of the Transportation Research Board 906, National Research Council, 1983, s. 75-81

[8] Mirchandani P. B., Lucas D., RHODES – ITMS, TEMPE FIELD TEST PROJECT Implementation and Field Testing Of RHODES, A Real-Time Traffic Adaptive Control System, Departament Transportu w Arizonie, Phoenix, Arizona 85007, 2001

[9] Lowrie P. R., The Sydney Coordinated Adaptive Traffic System – Principles, Methodology, Algorithms, Proceedings of the IEE International Conference on Road Signalling, London, United Kingdom, 1982, s. 67-70

[10] Peek Traffic Limited, Siemens Traffic Controls and TRL Limited, SCOOT - The World's Leading Adaptive Traffic Control System, 2011, http://www.scoot-utc.com/

[11] Mauro V., Taranto C. D., UTOPIA, Proceedings of the 6th IFAC/IFORS Conf. on Control, Computers and Communications in Transport, Paris, France, 1989, s. 245-252

[12] Robertson D. I., and R. D. Bretherton, Optimal Control of an Intersection for Any Known Sequence of Vehicle Arrivals, Proceedings of the 2nd IFAC-IFIP-IFORSSymposium of Traffic Control and Transportation Systems, North-Holland, Amsterdam, Netherlands, 1974, s. 3-17

[13] Wulffius H., TRAVOLUTION – optymalizacja całej sieci sygnalizacji ulicznej przy pomocy genetycznych algorytmów (I), 2010, http://edroga.pl/inzynieria-ruchu/its/2102-travolution-optymalizacja-calej-sieci-sygnalizacji-ulicznej-przy-pomocy-genetycznych-algorytmow-i

[14] Royani T., Haddadnia J., Alipoor M., Traffic Signal Control for Isolated Intersections Based on Fuzzy Neural Network and Genetic Algorithm, Proceedings of the 10th WSEAS international conference on Signal processing, computational geometry and artificial vision, Wisconsin, USA, 2010, s. 87-91

[15] Lindsey C. R., Verhoef E. T., Congestion Modelling, Tinbergen Institute, Rotterdam, Netherlands, 1999, s. 1-17

[16] Hu J., Wang Y., Zhang Z., Li D., Analysis on traffic flow data and extraction of nonlinear characteristic quantities, 13th International IEEE Conference on Intelligent Transportation Systems (ITSC), Funchal, Portugal, 2010, s. 712-717

[17] Gaca S., Suchorzewski W., Tracz M., Inżynieria ruchu drogowego. Teoria i praktyka, W.K.Ł., Warszawa, 2008

[18] Kaczmarek M., System zarządzania ruchem – cz. I, 2009, http://edroga.pl/inzynieria-ruchu/its/647-system-zarzadzania-ruchem-cz-i

[19] Gartner N., Messer C. J., Rathi A. K., "Traffic Flow Theory" – Revised Monograph on Traffic Flow Theory, Transportation Research Board, Washington, USA, 2005

[20] Taggart I., Inventing history: Garrett Morgan and the traffic signal, 2011, http://www33.brinkster.com/iiiii/trfclt/

[21] Mueller E. A., Aspects of the History of Traffic Signals, IEEE Transactions on Vehicular Technology, vol. VT-19, 1970, s. 6-17

[22] Papageorgiou M., Diakaki Ch., Dinopoulou V., Kotsialos A., Wang Y., Review of Road Traffic Control Strategies, Proceedings of the IEEE, 91 (12), 2003, s. 2043-2067

[23] Leśko M., Guzik J., Sterowanie ruchem drogowym - sterowniki i systemy sterowania i nadzoru ruchu, Wydawnictwo Politechniki Gliwickiej, Gliwice, 2000

[24] Lin S., Efficient Model Predictive Control for Large-Scale Urban Traffic Networks, Ph.D. dissertation, Delft University of Technology, Netherlands, 2011

[25] Hegyi A., De Schutter B., Hellendoorn J., Optimal coordination of variable speed limits to suppress shock waves, IEEE Transactions on Intelligent Transportation Systems, 6 (1), 2005, s. 102–112

[26] Hegyi A., De Schutter B., Hellendoorn J., Model predictive control for optimal coordination of ramp metering and variable speed limits, Transportation Research Part C: Emerging Technologies, 13 (3), 2005, s. 185-209

[27] Kesting A., Schönhof M., Lämmer S., Treiber M., Helbing D, Understanding Complex Systems, Chapter: Decentralized Approaches to Adaptive Traffic Control, Springer-Verlag Berlin, Heidelberg, 2008, s. 189-199

[28] Vrancken J. L. M., Soares M., Multi-level Control of Networks, the Case of Road Traffic Control, IEEE International Conference on Systems, Man and Cybernetics, Montreal, Canada, 2007, s. 1741-1745

[29] Kaczmarek M., Adaptacyjne sterowanie wielowarstwowe ruchem w sieciach ulic w ramach okien czasowych, Autostrady, 5/2004, 2004, s. 26-27

[30] Balke K., Charara H., Parker R., Development Of A Traffic Signal Performance Measurement System (TSPMS), Technical Report, FHWA/TX-05/0-4422-2, 2005

[31] Papageorgiou M., Ben-Akiva M., Botton J., Bovy P. H. I., Hoogendoorn S. P., Hounsell N. B., Kotsialos A., McDonald M., ITS and Traffic Management, Handbooks in Operations Research and Management Science, vol. 14, 2007

[32] Allsop R. B., SIGSET: A computer program for calculating traffic capacity of signal-controlled road junctions, Traffic Engineering & Control, vol. 12, 1971, s. 58-60

[33] Allsop R. B., SIGCAP: A computer program for assessing the traffic capacity of signal-controlled road junctions, Traffic Engineering & Control, vol. 17, 1976, s. 338–341

[34] Webster F. V., Traffic signal settings, Road Research Laboratory, London, United Kingdom, Road Research Paper Nr 30, 1958

[35] Little J. D. C., Kelson M. D., Gartner N., MAXBAND: A Program For Setting Signals On Arteries And Triangular Networks, Transportation Research Record No. 795, Traffic Flow Theory and Characteristics, 1981, s. 40-46

[36] Improta G., Cantarella G. E., Control systems design for an individual signalised junction, Transportation Research Board, vol. 18, 1984, s. 147–167

[37] Garfinkiel R. S., Nemhauser G. N., Programowanie całkowitoliczbowe, PWN, Warszawa, 1978

[38] Clausen J., Branch and Bound Algorithms - Principles and Examples, University of Copenhagen, Denmark, 1999

[39] Ruchaj M., Podstawowe problemy optymalizacji z ograniczeniami, Zeszyt Naukowy "Elektryka" Nr 59, vol. 322, Politechnika Opolska, 2007, s. 35-36

[40] Chang E. C.-P., Cohen S. L., Liu C., Chaudhary N. A., Messer C., MAXBAND-86: Program For Optimizing Left-Turn Phase Sequence In Multiarterial Closed Networks, Transportation Research Record No. 1181, Urban Traffic Systems and Parking, 1988, s. 61-67

[41] Chaudhary N. A., Pinnoi A., Messer C., Proposed enhancementsto MAXBAND-86 program, Transportation Research Record No. 1324, Communications, Traffic Signals, and Traffic Control Devices, 1991, s. 98-104

[42] Robertson D. I., TRANSYT method for area traffic control, Traffic. Engineering & Control, vol. 10, 1969, s. 276-281

[43] Li M.-T., Gan A .C., Signal timing optimization for oversaturated networks using TRANSYT-7F, Transportation Research Record No. 1683, Advanced Traffic Management Systems, 1999, s. 118-126

[44] Little J. D. C., The synchronization of traffic signals by mixed integer-linear-programming, Oparation Research, vol. 14, 1966, s. 568–594

[45] Stamatiadis C., Gartner N., MULTIBAND-96: A program for variable bandwidth progression optimization of multiarterial traffic networks, Transportation Research Record No. 1554, Advanced Traffic Management Systems and High-Occupancy-Vehicle Systems, 1996, s. 9-17

[46] Wong S. C., Derivatives of the performance index for the traffic model from TRANSYT, Transportation Research Part B: Methodological, 29 (5), 1995, s. 303-327

[47] Miller A. J., A computer control system for traffic networks, Procedings of the 2nd International Symposium on Traffic Theory, University of Birmingham, 1963, s. 200-220

[48] Vincent R. A., Young C. P., Self-optimizing traffic signalcontrol using microprocessor: The TRRL "MOVA" strategy for isolated intersections, Traffic Engineering & Control, vol. 27, 1986, s. 385–387

[49] Little J. D. C., The synchronization of traffic signals by mixed-integer-linear-programming, Oparation Research, vol. 14 (4), 1966, s. 568–594

[50] Koonce P., Rodegerdts L., Lee K., Quayle, Beaird S., Braud C., Bonneson J., Tarnoff Ph., Urbanik T., Traffic signal timing manual, 2008

[51] Xie Y., Development And Evaluation Of An Arterial Adaptive Traffic Signal Control System Using Reinforcement Learning, Ph.D. dissertation, Texas A&M University, 2007

[52] Gartner N. H., Prescription for Demand-Responsive Urban Traffic Control, Journal of the Transportation Research Board 881, Washington D.C., 1982, s. 73-76

[53] Hunt P. B., A Traffic Responsive Method of Coordinating Signals, Transport and Road Research Laboratory, 1981

[54] Abbas M. M., A Real Time Offset Transitioning Algorithm for Coordinating Traffic Signals, Ph.D. dissertation, West Lafayette, Indiana, USA, 2001

[55] Shelby S. G., Single-Intersection Evaluation of Real-Time Adaptive Traffic Signal Control Algorithms, Journal of the Transportation Research Board, Washington D.C., 2004, s. 183-192

[56] The Utah Traffic Laboratory (UTL), The "SCOOT" Urban Traffic Control System, 2006, http://www.trafficlab.utah.edu/documents/dft_roads_source_504749.pdf

[57] Sims A. G., Dobinson K. W., The Sydney Coordinated Adaptive Traffic (SCAT) System Philosophy and Benefits, IEEE Transactions on Vehicular Technology, 29 (2), 1980, s. 130-137

[58] Ozbay K. and others, Evaluation of Adaptive Control Strategies for NJ Highways, Dept. of Civil & Environmental Engineering, Center for Advanced Infrastructure & Transportation (CAIT), 2006

[59] Serbeńska A., Rzeszowski SCATS, 2010, http://edroga.pl/inzynieria-ruchu/wykonawstwo/1358-rzeszowski-scats

[60] Gustek Z., Olsztyńskie Centrum Sterowania Ruchem, 2009, http://edroga.pl/inzynieria-ruchu/wykonawstwo/158-olsztynskie-centrum-sterowania-ruchem

[61] Kaczmarek M., System zarządzania ruchem – cz. II, 2009, http://edroga.pl/inzynieria-ruchu/its/663-system-zarzadzania-ruchem-cz-ii

[62] Bellman R. E., Dynamic Programming, Princeton University Press, Princeton, New Jersey, USA, 1957

[63] Henry R. D., Ferlis R. A., Kay J. L., Evaluation of UTCS Control Strategies - Executive Summary, U.S. Department of Transportation, Report No. FHWA-RD-76-149, 1976

[64] Holroyd J., Robertson D. I., Strategies for Area Traffic Control Systems Present and Future, 1973

[65] Gartner N. H., Pooran F. J., Andrews C. M., Implementation of the OPAC Adaptive Control Strategy in a Traffic Signal Network, Intelligent Transportation Systems Conference 2001, Oakland, California, 2001, s. 195-200

[66] Mirchandani P., and L. Head, A Real-Time Traffic Signal Control System: Architecture, Algorithms, and Analysis, Transportation Research Part C, 9 (6), 2001, s. 415-432

[67] Head L., Mirchiandani P., RHODES - ITMS, Arizona Department of Transportation, Report AZ-SP-9701, Phoenix, Arizona 85007, 1997, s. AZ-SP-9701

[68] Sen S., Head K. L., Controlled Optimization of Phases at An Intersection, Transportation Science, 31 (1), 1997, s. 5-17

[69] Dell'Olmo P., Mirchandani P. B., REALBAND: An Approach for Real-Time Coordination of Traffic Flows on a Network, Journal of the Transportation Research Board, National Research Council, Washington D.C., 1995, s. 106-116

[70] Donati F., V. Mauro, G. Roncolini, and M. Vallauri, A Hierarchical-Decentralized Traffic Light Control System, Proceedings of IFAC 9th Triennial World Congress, Budapest, Hungars, 1984, s. 2853-2858

[71] Davidsson F., Taranto C. D., Priority for Public Transport Using Advanced Transport Telematics, Proceedings of The 3rd International Conference on Vehicle Navigation and Information Systems, Oslo, Norway, 1992, s. 530-536

[72] Henry J. J., Farges J. L., Tuffal J., The PRODYN Real Time Traffic Algorithm, Proceedings of IFAC Control in Transportation Systems, Baden-Baden, Germany, 1983, s. 305-310

[73] Henry J. J., Farges J. L., PRODYN, Proceedings of the 6th IFAC/IFIP/IFORS Symposium on Control, Computers, and Communications in Transportation, Paris, France, 1989

[74] Henry J. J., Farges J. L., Tuffal J., The PRODYN real time traffic algorithm, Proceedings of the 4th IFAC/IFORS. Conference on Control in Transportation Systems, Baden-Baden, Germany, 1983, s. 307-312

[75] Farges J. L., Khoudour L., Lesort J. B., PRODYN: On Site Evaluation, Proceedings of Third International Conference on Road Traffic Control, London, United Kingdom, 1990, s. 62-66

[76] Dreyfus S. E., Averill M. L., The art and theory of dynamic programming, Academic Press Inc., London, 1977

[77] Porche I. R, Dynamic Traffic Control: Decentralized and Coordinated Methods, Ph.D. dissertation, Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, Michigan, USA, 1997

[78] Porche I., Sampath M., Sengupta R., Chen Y.-L., Lafortune S., A Decentralized Scheme for Real-Time Optimization of Traffic Signals, Proceedings of the 1996 IEEE International Conferance on Control Applications, Dearborn, Michigan, 1996, s. 582-589

[79] Porche I., Lafortune S., Adaptive Look-ahead Optimization of Traffic Signals, ITS Journal - Intelligent Transportation Systems Journal, 4 (3-4), 1999, s. 209-254

[80] Kelly T., Extending the ALLONS Traffic Control Algorithm to Systems of Signlized Intersection, University of Michigan, Artificial Intelligence Lab., 1998

[81] Porche I., Lafortune S., A game-theoretic approach to signal coordination, Transportation Research Board, The University of Michigen, Department of Electrical Engineering and Computer Science, Ann Arbor, 2005

[82] Yu X.-H., Recker W. W., Stochastic adaptive control model for traffic signal systems, Transportation Research Part C, 14 (4), 2006, s. 263-282

[83] Gosavi A., Simulation-Based Optimization: Parametric Optimization Techniques and Reinforcement Learning, Kluwer Academic Publishers, Norwell, Massachusetts, 2003

[84] Sutton R. S., Barto A. G., Reinforcement Learning: An Introduction, The MIT Press, Cambridge, Massachusetts, 1998

[85] Chiu S., Chand S., Adaptive Traffic Signal Control Using Fuzzy Logic, Proceedings of 2nd IEEE International Conferences on Fuzzy Systems, San Francisco, California, 1993, s. 1371-1376

[86] Cuena J., Hernandez J., Molina M., Knowledge-based models for adaptive traffic management systems, Transportation Research Part C, 3(5), 1995, s. 311-337

[87] Dimitriou L., Tsekeris T., Stathopoulos A., Adaptive hybrid fuzzy rule-based system approach for modeling and predicting urban traffic flow, Transportation Research Part C, 16 (5), 2008, s. 554-573

[88] Jiang J. S. R., Sun C. T., Mizutani E., Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence, Prentice Hall, Upper Saddle River, New Jersey, 1997

[89] Murat Y. S, Gedizlioglu E., A Fuzzy Logic Multi-Phased Signal Control Model for Isolated Junctions, Transportation Research Part C, 13 (1), 2005, s. 19-36

[90] Niittymaki J., Pursula M., Signal Control Using Fuzzy Logic, Fuzzy Sets and Systems, 16 (1), 2000, s. 11-22

[91] Pappis C. P., Mandani E. H., A Fuzzy Logic Controller for a Traffic Junction, IEEE Transaction on Systems, Man and Cybernetics, SMC-7, 1977, s. 707-717

[92] Trabia M. B., Kaseko M. S., Ande M., A Two-Stage Fuzzy Logic Controller for Traffic Signals, Transportation Research Part C, vol. 7, 1999, s. 353-367

[93] Zarandi Z. H. F., Rezapour S., A Fuzzy Signal Controller for Isolated Intersections, Journal of Uncertain System, 3 (3), 2009, s. 174-182

[94] Mehta S., Fuzzy Control System For Controlling Traffic Lights, Proceedings of the International MultiConference of Engineers and Computer Scientists 2008, vol. I, 2008, s. 19-21

[95] Zhang L., Li H., Signal Control for Oversaturated Intersections Using Fuzzy Logic, Submitted for consideration for presentation at the 2005 Annual Meeting of the TRB and publication in the Transportation Research Record, Washington, D.C., 2005, s. 179-184

[96] Czogala E., Leski J., Fuzzy and Neuro-Fuzzy Intelligent Systems, Physica-Verlag Heidelberg, NewYork, 1999

[97] Owen L. E., Stallard Ch. M., GASCAP - (Generalized adaptive signal control method and system), 2000, http://www.patentgenius.com/patent/6587778.html

[98] Owen L. E., Stallard C. M., Rule-Based Approach to Real-Time Distributed Adaptive Signal Control, Journal of the Transportation Research Board, TRB, National Research Council, Washington D.C., 1995, s. 95-101

[99] Elahi S. M., Radwan A. E., Goul K. M., Knowledge-Based System for Adaptive Traffic Signal Control, Journal of the Transportation Research Board, TRB, National Research Council, Washington, D.C., 1987, s. 115-122

[100] Lin F. B., Comparative Analysis of Two Logics for Adaptive Control of Isolated Intersections, Journal of the Transportation Research Board, TRB, National Research Council, Washington, D.C., 1988, s. 6-14

[101] Logi F., Ritchie S. G., Development and evaluation of a knowledge-based system for traffic congestion management and control, Transportation Research Part C, TRB, National Research Council, Washington, D.C., 2000, s. 433-459

[102] Ritchie S. G., A knowledge-based decision support architecture for advanced traffic management, Transportation Research Part A, 24 (1), 1990, s. 27-37

[103] Sadek A. W., Demetsky M. J., Smith B. L., Case-based reasoning for realtime traffic flow management, Computer-Aided Civil and Infrastructure Engineering, 14 (5), 1999, s. 347-356

[104] Conrad M., Dion F., Yagar S., Real-Time Traffic Signal Optimization with Transit Priority: Recent Advances in the SPPORT Model, Transportation Research Record 1634, TRB, National Research Council, Waszyngton D.C., 1998, s. 100-109

[105] Dion F., Development and Evaluation of a Distributed System for the Real-Time Control of Signalized Netowks, Ph.D. dissertation, University of Waterloo, Waterloo, Ontario, 1998

[106] Dion F., Helling B., A Methodology for Obtaining Signal Coordination within A Distributed Real-Time Network Signal Control System with Transit Priority, Transportation Research Board, The National Academies, Waszyngton, 2001

[107] Dion F., Hellinga B., A rule-based real-time traffic responsive signal control system with transit priority: application to an isolated intersection, Transportation Research Part B, The National Academies, Waszyngton, 2002, s. 325-343

[108] Dion F., Yagar S., Distributed Approach to Real-Time Control of Complex Signalized Networks, Transportation Research Record 1554, TRB, National Research Council, Waszyngton D.C., 1996, s. 1-8

[109] Kim J.-T., Lee J., Chang M., Neural-network-based cycle length design for real-time traffic control, 2008, http://www.freepatentsonline.com/article/Canadian-Journal-Civil-Engineering/180518645.html

[110] Nakatsuji T., Kaku T., Development Of A Self-Organizing Traffic Control System Using Neural Network Models, Transportation Research Record No. 1324, Communications, Traffic Signals, and Traffic Control Devices, 1991, s. 137-145

[111] Nakatsuji T., Seki S., Shibuya S., Kaku T., Artificial intelligence approach for optimizing traffic signal timings on urban road network, Proceedings of Vehicle Navigation and Information Systems Conference, Yokohama, Japan, 1994, s. 199-202

[112] Ohira T., Inoue K., Takeshima Y., Adaptive Traffic Control Model with Neural Network Analogy, Proceedings of ICONIP'97, Dunedin, New Zeland, 1998, s. 939-942

[113] Teodorovic D., Varadarajan V., Popovic J., Chinnaswamy M. R., Ramaraj S., Dynamic programming - neural network real-time traffic adaptive signal control algorithm, Springer Science+Business Media, Annals of Operations Research, 2006, s. 123–131

[114] Wulffius H., TRAVOLUTION – optymalizacja całej sieci sygnalizacji ulicznej przy pomocy genetycznych algorytmów (II), 2010, http://edroga.pl/inzynieria-ruchu/its/2115-travolution-optymalizacja-calej-sieci-sygnalizacji-ulicznej-przy-pomocy-genetycznych-algorytmow-ii?start=2

[115] Taale H., Optimising traffic signal control with evolutionary algorithms, Proceedings of 7th World Congress on Intelligent Transport Systems (ITS'00), http://www.irpds.com/FileEssay/barnamerizi-1386-12-8-bgh(77).PDF, 2000, s. 83-91

[116] Turky A. M., Ahmad M. S., Yusoff M. Z. M, The use of genetic algorithm for traffic light and pedestrian crossing control, IJCSNS International Journal of Computer Science and Network Security, 9 (2), 2009, s. 88-96

[117] Almejalli K. A., Intelligent Real-Time Decision Support Systems for Road Traffic Management. Multi-agent based Fuzzy Neural Networks with a GA learning approach in managing control actions of road traffic centres, Ph.D. dissertation, University of Bradford, 2010

[118] Huang Z.-J., Li C.-G., Zhang Z.-F., Traffic signal control based on genetic neural network algorithm, Proceedings of International Conference on Intelligent Computing and Integrated Systems (ICISS), Guilin, China, 2010, s. 31-34

[119] Linsen Chong Abbas M., Neuro-Fuzzy Actor Critic Reinforcement Learning for determination of optimal timing plans, Proceedings of 13th International IEEE Conference on Intelligent Transportation Systems (ITSC), Funchal, Portugal, 2010, s. 545 - 550

[120] Łęski J., Systemy neuronowo-rozmyte, WNT, Warszawa, 2008

[121] Amt für Verkehrsmanagement und Geoinformation von Stadt Ingolstadt, AUDI, GEVAS Software, Technische Universität München, TRAVOLUTION in Ingolstadt, 2011, http://www.travolution-ingolstadt.de/

[122] GEVAS software, Audi i inni, TRAVOLUTION - Green Wave Optimization with Genetic Algorithms, Car-to-Infrastructure Communication, GEVAS software Systementwicklung und Verkehrsinformatik GmbH, Munich, Germany, 2009

[123] Braun R., Busch F., Kemper C., Hildebrandt R., Weichenmeier F., Menig C., Paulus I., Preßlein-Lehle R., TRAVOLUTION – Netzweite Optimierung der Lichtsignalsteuerung und LSA-Fahrzeug-Kommunikation, Straßenverkehrstechnik 06/2009, FGSV (Hrsg) Kirschbaum Verlag, Bonn, Germany, 2009, s. 365-374

[124] Friedrich B., Ein verkehrsabhängiges Verfahren zur Steuerung von Lichtsignalanlagen, Veröffentlichung des FachgebietsVerkehrstechnik und Verkehrsplanung, The Technische Universität München, 1999

[125] Bellemans T., De Schutter B., De Moor B., Models for traffic control, Journal A, 43 (3-4), 2002, s. 13-22

[126] Detering S., Kalibrierung und Validierung von Verkehrssimulationsmodellen zur Untersuchung von Verkehrsassistenzsystemen, Doktor-Ingenieurs Dissertation, Die Technische Universität Carolo-Wilhelmina zu Braunschweig, 2011

[127] red. Barceló J., Fundamentals of Traffic Simulation, Springer Science+Business Media, New York, USA, 2010

[128] Leutzbach W., Introduction to the theory of traffic flow, Springer Verlag, Berlin, 1988

[129] Leonard D. P., Gower P., Taylor N., CONTRAM. Structure of the model, Transport and Road Research Laboratory, TRRL RESEARCH REPORT 178, 1989

[130] Mahut M., Florian M., Tremblay N., Calibration and applicationof a simulation based dynamic traffic assignment model, Transportation Research Board, 1876, 2004, s. 101-111

[131] Burghout W., Kotsopoulos H., Andréasson I., Hybrid mesoscopic–microscopic traffic simulation, Transportation Research Board, Traffic Flow Theory 2005, 2005, s. 218-225

[132] Maerivoet S., De Moor B., Cellular automata models of road traffic, Physics Reports, 419 (1), 2005, s. 1-64

[133] Wolfram S., Statistical Mechanics of Cellular Automata, Reviews of Modern Physics, 55 (3), 1983, s. 601-644

[134] von Neumann J., The general and logical theory of automata, John von Neumann COLLECTED WORKS, red. A. H. Taub, vol. 5, 1963, s. 288-326

[135] Wolfram S., A New Kind of Science, Wolfram Media, Champaign, USA, 2002

[136] Tonguz O. K., Viriyasitavat W., Bai F., Modeling Urban Traffic: A Cellular Automata Approach, IEEE Communications Magazine, 47 (5), 2009, s. 142-150

[137] Nagel K., Schreckenberg M., A cellular automoton model for freeway traffic, Journal de Physique, I 2, 1992, s. 2221-2229

[138] Chopard B., Luthi P. O., Queloz P.-A., Cellular Automata Model of Car Traffic in a Two-Dimensional Street Network, Journal de Physique, 29 (10), 1996, s. 2325-2336

[139] Bartodziej M., Modelowanie ruchu ulicznego za pomocą automatów komórkowych, praca dyplomowa, Politechnika Wrocławska, 2007

[140] Chopard B., Dupuis A., Luthi P., A cellular automata model for urban traffic and its application to the city of Geneva, Proceedings of Traffic and Granular Flow '97, World Scientific, 1996, s. 153-168

[141] Wooldridge M. J., An introduction to multiagent systems, Wiley & Sons Ltd., Chichester, West Sussex, United Kingdom, 2002

[142] Wooldridge M., Jennings N. R., Intelligent agents: theory and practice, The Knowledge Engineering Review, 10 (2), 1995, s. 115-152

[143] Almejalli K. A., Intelligent real-time decision support systems for road traffic management, Ph.D. dissertation, University of Bradford, 2010

[144] Genesereth M. R., Ketchpel S. P., Software agents, Communications of the ACM, 37 (7), 1994, s. 48-53

[145] Stone P., Veloso M., Multiagent Systems: A Survey from a Machine Learning Perspective, Autonomous Robotics, 8 (3), 2000, s. 345–383

[146] The Intelligent Systems Group of the Institute for Computer Science at the University of Utrecht, Netherlands, Green Light District, 2009, http://sourceforge.net/projects/stoplicht/

[147] Wiering M., van Veenen J., Vreeken J., Koopman A., Simulation and Optimization of Traffic in a City, Proceedings of IEEE Intelligent Vehicles Symposium (IV'04), Parma, Italy, 2004, s. 453–458

[148] Kuyer L., Whiterson S., Bakker B., Vlassis N., Multiagent Reinforcement Learning for Urban Trac Control using Coordination Graphs, Lecture Notes in Computer Science, vol. 5211, 2008, s. 656–671

[149] Cools S. B., A realistic simulation for self-organizing traffic lights, Thesis, Vrije Universiteit Brussel, Belgium, 2006

[150] Le T., Cai Ch., A New Feature For Approximate Dynamic Programming Traffic Light Controller, Proceedings of the Second International Workshop on Computational Transportation Science, San Jose, USA, 2010, s. 29-34

[151] Ruchaj M., Latawiec K., Review of selected road traffic simulators, Conference Archives PTETIS, vol. 25, X International PHD Workshop OWD 2008, Wisła, 2008, s. 151-153

[152] ORACLE, Java i Ty, 2011, http://www.java.com/pl/

[153] Eckel B., Thinking in Java, Helion, Edycja polska. Wydanie IV, 2006

[154] Free Software Foundation, GNU GENERAL PUBLIC LICENSE, 2007, http://www.gnu.org/licenses/gpl-3.0.txt

[155] The Intelligent Systems Group of the Institute for Computer Science at the University of Utrecht, Netherlands, Green Light District :: Documentation, 2001, http://sourceforge.net/projects/stoplicht/files/GLD%20Docs/GLD%201.01%20Docs/

[156] Ernst T., de la Fortelle A., Car-To-Car and Car-To-Infrastructure Communication System Based on NEMO and MANET in IPv6, Proceedings of 13th World Congress and Exhibition on Intelligent Transport Systems and Services, London, 2006, s. 1-7

[157] Schweiger B., Raubitschek Ch., Bäker B., Schlichter J., ElisaTM – Car to infrastructure next term communication in the field, Computer Networks, 55 (14), 2011

[158] Mitchell T. M., Machine Learning, McGraw-Hill, chapter 13, 1997, s. 367-390

[159] Iša J., Kooij J., Koppejan R., Kuijer L., Reinforcement Learning of Traffic Light Controllers Adapting to Accident, Design and Organisation of Autonomous Systems 2006, University of Amsterdam, Netherlands, 2006, s. 1-14

[160] L'Ecuyer P. the Département d'Informatique et de Recherche Opérationnelle (DIRO), Université de Montréal, SSJ: Stochastic Simulation in Java, 2011, http://www.iro.umontreal.ca/~simardr/ssj/indexe.html

[161] TLS Kosendiak Tomasz, Projekt korekt programów pracy sygnalizacji ulicznych, MZD Opole, Wrocław, 2008

[162] Toth V. T., Calculators and the Gamma Function, 2011, http://www.rskey.org/gamma.htm

[163] Ruchaj M., Wybrane algorytmy sterowania stosowane w acyklicznej sygnalizacji świetlnej na skrzyżowaniach dróg w drogowej sieci miejskiej, Zeszyt Naukowy "Informatyka", Politechnika Opolska, 2010, złożony do druku

[164] Ruchaj M., Stanisławski R., Approaches to control acyclic traffic lights in an exemplary urban road network, Proceedings of the 16th International Conference on Methods and Models in Automation and Robotics (MMAR) 2011, Międzyzdroje, 2011, s. 387-392

[165] Johnson A. W., Jacobson S. H., A class of convergent generalized hill climbing algorithms, Applied Mathematics and Computation, 125 (2-3), 2002, s. 359-373

[166] Johnson A. W., Jacobson S. H., On the convergence of generalized hill climbing algorithms, Discrete Applied Mathematics, 119 (1-2), 2002, s. 37-57

[167] Cichosz P., Systemy uczące się, WNT, Warszawa, 2009

[168] Barto A. G., Bradtke S. J., Singh S. P., Learning to act using real-time dynamic programming, Artificial Intelligence, 72, 1995, s. 81–138

[169] Tadeusiewicz R., Sieci Neuronowe, Akademicka Oficyna Wydawnicza RM, Warszawa, 1993

[170] Osowski S., Sieci neuronowe w ujęciu algorytmicznym, WNT, Warszawa, 1996

[171] Google Inc., Mapy Google, 2012, http://mapy.google.pl/

[172] Tracz M. i inni, Pomiary i badania ruchu drogowego, W.K.Ł., Warszawa, 1984

[173] Gasz K., Modelowanie ruchu w sieci ulic w warunkach ograniczonej przepustowości skrzyżowań, Politechnika Wrocławska, rozprawa doktorska, 2007

[174] The Scilab Consortium, Scilab - The Free Software for Numerical Computation, 2011, http://www.scilab.org/

[175] Scott D. W., Multivariate density estimation: theory, practice, and visualization, Wiley, USA, New York, 1992

[176] MathWave Technologies, EasyFit - Distribution Fitting Software, 2011, http://www.mathwave.com/products/easyfit.html

Dodatek 1: Spis rysunków

Dodatek 2: Spis tabel