**Akademia Górniczo-Hutnicza**

**im. Stanisława Staszica w Krakowie**

Wydział Elektrotechniki, Automatyki Informatyki i Inżynierii Biomedycznej

Katedra Informatyki Stosowanej



Praca inżynierska

Andrzej Tokarski

**Symulacja ruchu drogowego**

**metodami dyskretnymi i ciągłymi**

Promotor:

Dr inż. Jarosław Wąs

Kraków 2012

# Spis treści

[1 Wstęp 3](#_Toc342466413)

[1.1 Symulacja ruchu drogowego 3](#_Toc342466414)

[1.2 Plan pracy 3](#_Toc342466415)

[2 Badania literaturowe 4](#_Toc342466416)

[2.1 Podział 4](#_Toc342466417)

[2.2 Automaty komórkowe 4](#_Toc342466418)

[2.2.1 Rule 184 4](#_Toc342466419)

[2.2.2 Nagel i Schreckenberg 6](#_Toc342466420)

[2.3 Car-following model 8](#_Toc342466421)

[2.3.1 Gazis-Herman-Rothery (GHR) model 8](#_Toc342466422)

[2.3.2 Optimal velocity (OV) model 9](#_Toc342466423)

[2.3.3 Collision avoidance model (CA) 9](#_Toc342466424)

[2.3.4 Intelligent driver model (IDM) 10](#_Toc342466425)

[3 Implementacja 10](#_Toc342466426)

[3.1 Założenia 10](#_Toc342466427)

[3.1.1 Model 11](#_Toc342466428)

[3.2 Schematy UML 12](#_Toc342466429)

[3.3 Mapa 12](#_Toc342466430)

[3.3.1 Format danych źródłowych ( OSM XML ) 14](#_Toc342466431)

[3.3.2 Konwersja danych 15](#_Toc342466432)

[3.3.3 Projekcja mapy 15](#_Toc342466433)

[3.3.4 Normalizacja pozycji elementów 15](#_Toc342466434)

[3.3.5 Budowa sieci połączeń 16](#_Toc342466435)

[3.3.6 Znalezienie pozycji obiektów Lane i długości połączeń 18](#_Toc342466436)

[3.4 Symulacja 19](#_Toc342466437)

[3.4.1 Konfiguracja 19](#_Toc342466438)

[3.4.2 Krok symulacji 19](#_Toc342466439)

[3.4.3 Zachowanie na skrzyżowaniu 19](#_Toc342466440)

[3.4.4 Raporty 20](#_Toc342466441)

[3.5 Wybrane technologie 21](#_Toc342466442)

[3.5.1 Java 21](#_Toc342466443)

[3.5.2 XML 21](#_Toc342466444)

[3.5.3 GIT 21](#_Toc342466445)

[3.5.4 HTML 21](#_Toc342466446)

[3.6 Wnioski z implementacji 21](#_Toc342466447)

[4 Wyniki 21](#_Toc342466448)

[5 Bibliografia **Error! Bookmark not defined.**](#_Toc342466449)

# Wstęp

## Symulacja ruchu drogowego

Jedną z najbardziej dynamicznie rozwijających się dziedzin nauki jest symulacja ruchu drogowego. Możliwość precyzyjnego przewidywania natężenia ruchu w danym miejscu czy też określenia przepustowości skrzyżowania, stała się już podstawą planowania sieci drogowych dużych miast.

Co więcej, uzyskanie dokładnego modelu ruchu drogowego umożliwia rozwój inteligentnych systemów transportu (*intelligent transport system - ITS*), zmieniających rolę człowieka w transporcie kołowym z kierowcy na nadzorcę. Zabieg ten pozwoliłyby na skupienie uwagi na zajęciach niezwiązanych z prowadzeniem pojazdu (Dongxin, 2010). Niektóre z przedstawionych modeli odgrywają ważną rolę w systemach ostrzegania i unikania zderzeń, c*ollision-warning and collision-avidance - CW&CA* (Y. Weng, 2001).

Podstawowy podział symulacji ruchu drogowego dotyczy skali, w jakiej działają. Literatura rozgranicza modele mikroskopowe, dotyczące pojedynczych pojazdów i ich kierowców oraz makroskopowe, rozpatrujące ruch jako całościowe zjawisko, najczęściej w kontekście mechaniki płynów (Maurice J. Khabbaz, 2012). Inny podział dotyczy ciągłości (lub jej braku) reprezentacji czasu i przestrzeni, w których odbywa się symulacja (Traffic Simulation, 2012). W domenie dyskretnego czasu i przestrzeni prym wiodą modele oparte o automaty komórkowe, w dyskretnym czasie i ciągłej przestrzeni funkcjonuje model podążania (car-following model). Klasyfikacji tej umyka natomiast model Monte Carlo, przez użycie generatora liczb pseudo-losowych.

## Plan pracy

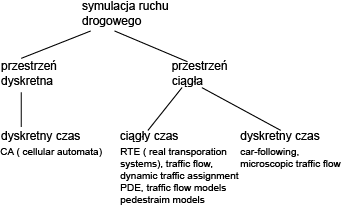
W ramach projektu inżynierskiego podjęto próbę implementacji dwóch modeli ruchu drogowego: opartego o idee automatów komórkowych, w dyskretnym czasie i przestrzeni oraz modelu podążania ,*car-following model*. Otrzymane dane porównano ze sobą oraz skonfrontowano z wynikami autorów modeli. Celem pracy jest budowa aplikacji pozwalającej na poglądową prezentację i porównanie różnych modeli ruchu.

# Badania literaturowe

Tematyka symulacji ruchu drogowego jest dynamicznie rozwijającą się gałęzią wiedzy. W tym rozdziale przedstawiono fragment systematyki tej dziedziny.

## Podział

Głównymi kryteriami rozróżniającymi są sposoby reprezentacji czasu i przestrzeni symulacji. Podstawowy podział przedstawia rysunek 1 wykonany na podstawie tabeli metod symulacyjnych (Traffic Simulation, 2012).



Rysunek 1. Kryteria podziału

## Automaty komórkowe

Automaty komórkowe (*cellular automata - CA)* zostały po raz pierwszy opisane w 1986 r przez Stephen-a Wolframa. [wolfram] Od tego czasu konsekwentnie zyskiwały na popularności i coraz częściej używane były w kontekście symulacji ruchu drogowego. Mogą z powodzeniem służyć jako narzędzie do obserwacji i analizy zachowania stanu sieci drogowych. Pomimo pewnej prostoty sieci automatów komórkowych*, cellular automata network* generują zachowania wykazujące komplikację odpowiednią do złożoności zagadnienia (Tao Yang, 2004).

### Rule 184

Prototyp takiego modelu został zaproponowany jako *Rule 184*. W pewnym przybliżeniu może opisywać on zachowanie pojazdów na drodze z jednym pasem ruchu.

Stan układu stanowi tablica wartości binarnych, w której 1 stanowi komórkę zajmowaną przez pojazd na drodze. W każdym kroku symulacji następuje jednoczesna zmiana stanu komórek zgodnie z *Rule 184*.

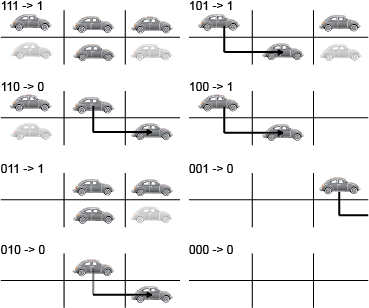
Nazwa *Rule 184* podchodzi od nazewnictwa zaproponowanego przez Stephena Wolframa i nazwanego później jego nazwiskiem (Wolfram, 1994). Liczba 184 powstała przez przekształcenie na kod dziesiętny binarnej reprezentacji funkcji zmiany stanu komórki układu.

Tabela 1 przedstawia zasady składające się na *Rule 184*. Pierwszy wiersz zawiera możliwe wartości rozpatrywanego pola i jego otoczenia. W drugim wierszu znajdują się odpowiednie wartości stanu w następnym kroku symulacji.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| stan n | 111 | 110 | 101 | 100 | 011 | 010 | 001 | 000 |
| stan n+1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |

Tabela 1. Zasady zmiany stanu

Zachowanie modelu można opisać następująco: jeśli w tablicy znajduje się para komórek o wartościach 1-0, wartości zamieniają się miejscami (jedynka porusza się w prawą stronę).



Rysunek 2. Przykładowy przebieg symulacji

Pomimo swojej prostoty model pozwala zaobserwować zachowania takie jak fale (zagęszczenia) w ruchu drogowym, gdzie samochody samoistnie łączą się w grupy. Podstawowym ograniczeniem modelu opartego o *Rule 184* jest stała prędkość przemieszczania się pojazdów, równa 1. Model ten stał się podstawą dalszych badań nad automatami komórkowymi w kontekście symulacji ruchu drogowego (M. Rickert, 1996).

### Nagel i Schreckenberg

Ważnym przykładem zastosowania CA w symulacji ruchu drogowego jest model zaprezentowany w publikacjach Nagela i Schreckenberga [nagel]. Podobnie jak model CA oparty jest o *Rule 184*. Dotyczy pojedynczego pasa ruchu podzielonego na dyskretnych fragmentów, z których każdy może być zajęty przez pojazd (wartość: 1) lub wolny (wartość: 0). Najważniejszą zaproponowaną zmianą jest wprowadzenie prędkość pojazdu **,** która może przyjmować wartości różne od jeden. Oznacza to, że podczas jednego kroku symulacji pojazdy mogą, o ile pozwalają na to warunki, przemieścić się o więcej niż jedno pole.

Model został poszerzony o następujące cechy:

* **przyspieszenie:** jeśli prędkość jest mniejsza niż ustalone to prędkość zwiększana jest o jeden
* **spowolnienie:** jeśli odległość do następnego pojazdu w ciągu jest nie większa niż to prędkość jest zmniejszana do granicy
* **element losowy:** z prawdopodobieństwem , prędkosc pojazdu jest zmniejszana o jeden
* **przemieszczenie:** każdy pojazd przemieszcza się o pól.

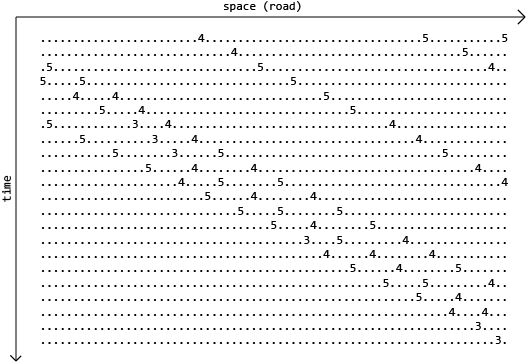
Oznaczenia ilustruje rysunek 3.



Rysunek 3 Ilustracja użytych symboli

Zasady stosowane są jednocześnie (*parallel update*). Dodatkowo, dzięki elementowi losowemu model generuje zachowania niedeterministyczne (M. Rickert, 1996).

Przykładową reprezentację i ruch pojazdów przedstawiono na rysunku 3.



Rysunek 4. Ruch pojazdów w modelu Nagel-Schreckenberg

Rysunek wykonano na postawie grafiki przedstawionej w publikacji autorów modelu [nagel].

W publikacji z 1994 Nagel i Schreckenberg przedstawiają analityczne rozwiązanie tego modelu dla . Dla istnieje rozwiązanie przybliżone.

## Car-following model

Mikroskopowy model podążania (*car-following model*) jest sposobem przedstawienia ruchu drogowego, opartym na zjawisku podążania pojazdów za poprzedzającymi je, na pojedynczym pasie ruchu. Model ten pozwala na łatwe odzwierciedlenie czynnika ludzkiego w prowadzeniu pojazdu [weng]. W tym podrozdziale przedstawiono trzy przykłady modeli opartych o idee podążania.

W ogólności, zasadę modelu można określić jako: „każdy pojazd zawsze zwalnia lub przyspiesza w odpowiedzi na bodziec” (Y. Weng, 2001).

Bodźcami mogą być:

* Prędkość rozpatrywanego pojazdu
* Względna prędkość rozpatrywanego pojazdu **n** i pojazdu przed nim **n-1**
* Odległość między rozpatrywanym pojazdem **n** i pojazdu przed nim **n-1**

### Gazis-Herman-Rothery (GHR) model

Pierwszym, określającym powyższą relację, był model GHR [weng01]. Opisuje on przyspieszenie rozpatrywanego pojazdu, jako:

Gdzie określa przyspieszenie pojazdu w czasie to względną prędkość pojazdu w poprzedniej jednostce czasu, to czas reakcji kierowcy, jest współczynnikiem czułości odpowiedzi kierowcy na zmianę warunków.

Podstawową wadą tego modelu było jego nienaturalne zachowanie w warunkach dużej gęstości ruchu. W celu zbliżenia modelu do rzeczywistości wprowadzono element .

W którym, jest względną odległością między dwoma pojazdami. Dodatkowym czynnikiem, na który zwrócono uwagę była prędkość pojazdu na szosie. Ostatecznie, wzór rozszerzony o ten czynnik przedstawia się następująco:

Najważniejszym elementem modelu GHR są parametry , których wartości radykalnie wpływają na zachowanie symulacji, w efekcie ograniczając możliwość wykorzystania modelu (Y. Weng, 2001).

### Optimal velocity (OV) model

Model optymalnej prędkości (*optimal velocity*) opiera się na założeniu, że kierowca w reakcji na bodziec nie określa pożądanego przyspieszenia chwilowego, a jedynie prędkość, z jaką pragnie się poruszać.

Gdzie reprezentuje pożądaną prędkość pojazdu w chwili .

Szczególnym przypadkiem powyższego wzoru jest model GHR, gdy za podstawimy prędkość następnego pojazdu . Jakkolwiek, model OV zakłada, że prędkość zależy również od odległości między samochodami.

Ostateczny wzór modelu przedstawia się więc następująco:

Istnieje wiele definicji , jedną z nich jest poniższy zestaw zależności:

dla

dla

dla

Model OV opisuje bezpieczną strategię poruszania się w ruchu drogowym. Przy zachowaniu odpowiedniej prędkości kierowca zachowuje bezpieczną odległość od pojazdu przed nim, jednocześnie biorąc pod uwagę względną prędkość obu pojazdów (Y. Weng, 2001).

### Collision avoidance model (CA)

Przeciwnie do powyższych, model CA nie opisuje odpowiedzi na bodziec. Zamiast tego, wyznacza bezpieczny (z perspektywy kierowcy) dystans, który powinien dzielić go od następnego pojazdu. [weng]

Przy pomocy modelu CA można z powodzeniem opisać wiele sytuacji na drodze, w tym propagację zakłóceń ruchu. Można go również kalibrować w oparciu o wzorce intuicyjnych zachowań kierowców. Jest szeroko stosowany w rozwiązaniach takich jak model SISTM departamentu ruchu drogowego Anglii, model SPEACS i INTRAS.

### Intelligent driver model (IDM)

W poniższym dziale opisano model przedstawiony przez grupę badawczą Daimin Tang, Xiang Li, Yijuan Jiang w pracy „Microscopic Traffic Simulation Oriented Road Network Data Model” (Daimin Tang, 2010).

Sieć drogowa, według IDM, przedstawia się następująco:

Podstawowym elementem jest oś szosy (*RoadCenterline*), której parametry pochodzą z map czy zdjęć satelitarnych. Kształt i parametry nadają drodze segmenty (*RoadSegment*). Z tych danych powstaje kierunek (*Carriageway*), czyli fragment szosy odpowiadający jednemu kierunkowi ruchu, który natomiast można podzielić na pasy ruchu (*Lane*). Osobnym elementem jest skrzyżowanie (*Intersection*).

Model opisuje poniższe równanie:

gdzie

# Implementacja

Podstawowym celem projektu było stworzenie aplikacji pozwalającej na symulację ruchu drogowego w modelu dyskretnym i ciągłym.

## Założenia

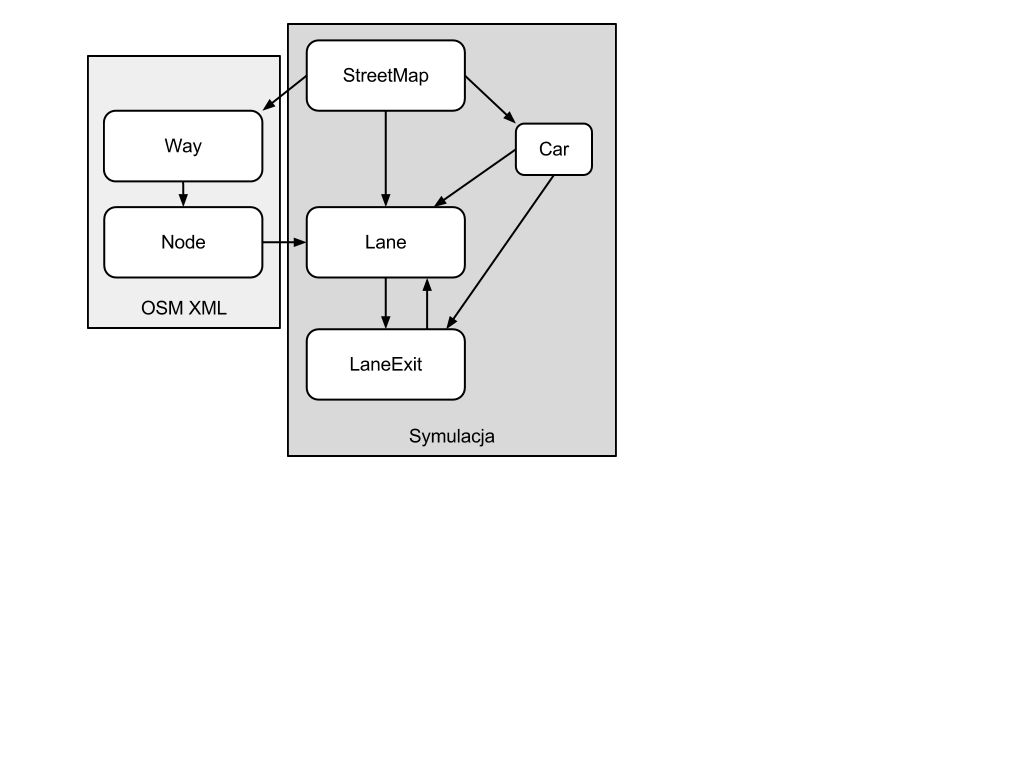
Aby uczynić implementację możliwą do wykonania konieczne było ustalenie podstawowych założeń na temat mechanizmu symulacji:

* Czas symulacji jest dyskretny,
* Pojazdy są rozróżnialne i poruszają się po pasach ruchu,
* Zmiana pasa możliwa jest tylko na skrzyżowaniu lub zakończeniu drogi,
* Droga posiada dwa pasy ruchu (po jednym w każdym kierunku),
* Na skrzyżowaniu możliwe jest wykonanie skrętu w trzech kierunkach.

Każdy z powyższych punktów zostanie rozwinięty w dalszej części tego dokumentu.

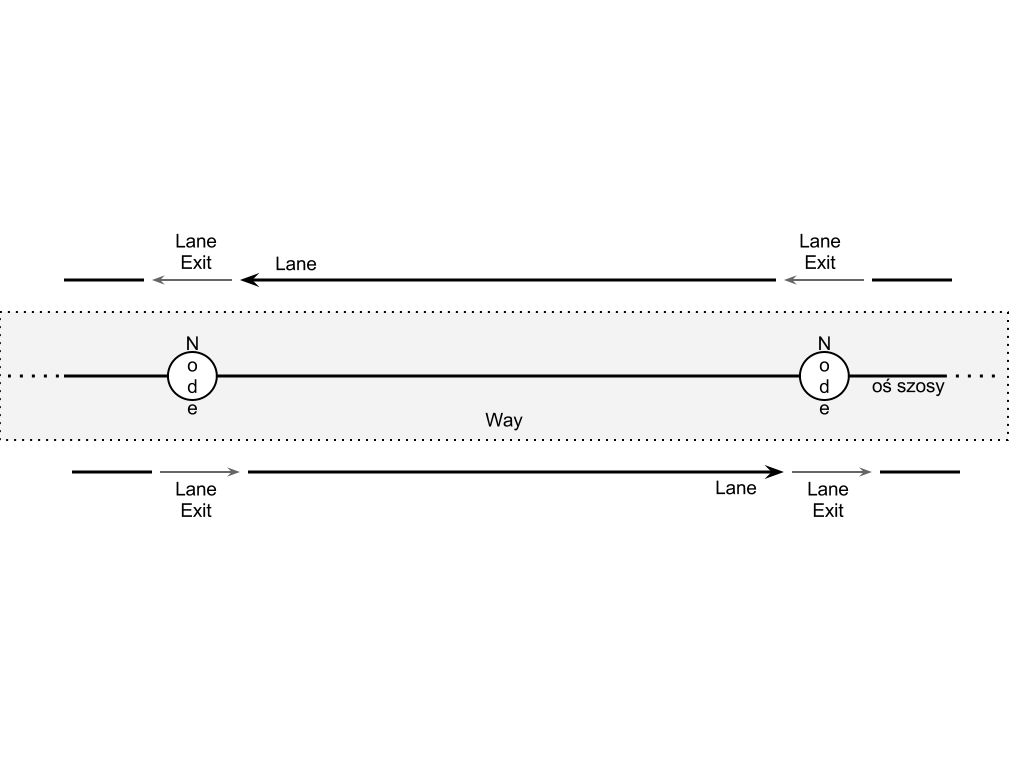
### Model

Sieć drogowa reprezentowana w programie jest przez zbiór obiektów Way, Node, Lane i LaneExit. Obiekty Way i Node pochodzą bezpośrednio z informacji pobranych z mapy. Obiekty Lane i LaneExit powstają w wyniku konwersji danych na potrzeby symulacji. Dodatkowo, do reprezentacji pojedynczego pojazdu służy obiekt Car. Cały stan symulacji zawiera się w obiekcie StreetMap. Relacje zachodzące między obiektami ilustruje Rysunek 5.



Rysunek 5. Struktura relacji obiektów modelu

Obiekty Node i Lane posiadają zestaw współrzędnych pozwalający na umiejscowienie ich w świecie symulacji i na ekranie. Pozycja LaneExit jest zdeterminowana przez łączone obiekty Lane. Miejsce, w którym znajduje się pojazd (obiekt Car) określamy poprzez wskazanie obiektu Lane i odległości, jaką przebył od jej początku.

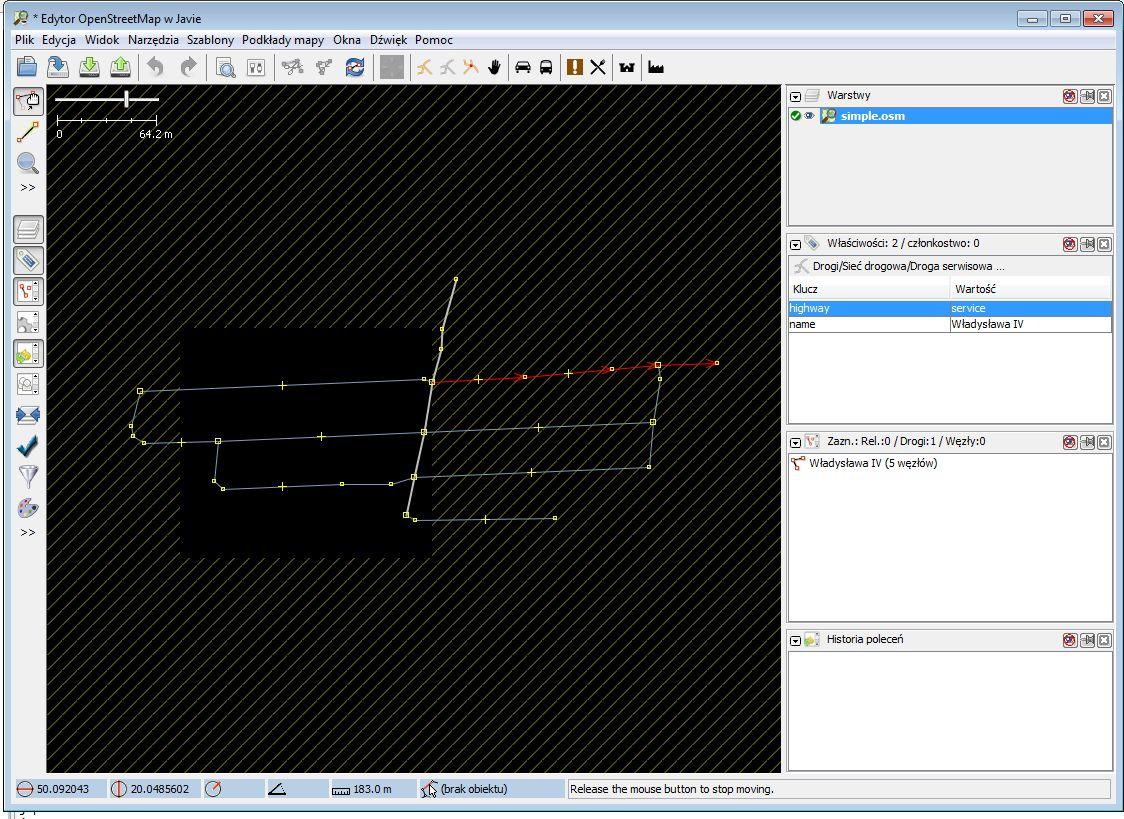


Rysunek 6. Obiekty modelu tworzące fragment drogi po dodaniu obiektów symulacji

## Schematy UML

## Mapa

Aby symulacja mogła dać realistyczne efekty, potrzebna była siatka dróg możliwie podobna do rzeczywistości, przez co konieczne było użycie rzeczywistej mapy. Siatkę dróg uzyskano z serwisu OpenStreetMaps w postaci pliku XML (zwanym OSM XML). Dane te udostępniane są na licencji Creative Commons Attribution-ShareAlike 2.0 co oznacza, że warunkiem ich użycia jest podanie źródła ich pochodzenia. Inicjatywa OpenStreetMaps dostarcza również edytor formatu OSM XML o nazwie JSOM pozwalający na ręczną obróbkę danych.

Rysunek 7. Edycja mapy OSM XML w programie JSOM

Do przeprowadzenia symulacji wybrano fragment mapy, zawierający niewielką ilość dróg o niskiej klasie, spełniających jednocześnie założenie o ruchu dwukierunkowym (istnienie dwóch pasów ruchu, po jednym w każdym kierunku). Wybrany fragment zawiera odpowiednią ilość skrzyżowań, pozwalającą na obserwację zachowania pojazdów w ich okolicy.



Rysunek 8. Źródłowy fragment mapy OSM

### Format danych źródłowych ( OSM XML )

Główną cechą formatu OSM XML jest mnogość typów tagów dostępnych dla twórców map. Droga reprezentowana jest przez kolekcje tagów node.

<node id='**691538992**' lat='50.09368642997591' lon='20.044782298879543' />

Każdy tag node reprezentuje pojedynczy punkt na mapie. W szczególności może być to jeden z punktów z krzywej tworzącej oś szosy. W takim przypadku numery id są użyte w tagu way.

<way id='29401317'>

<nd ref='**691538992**' />

<nd ref='691538992' />

<tag k='highway' v='residential' />

<tag k='name' v='Jana Kazimierza' />

</way>

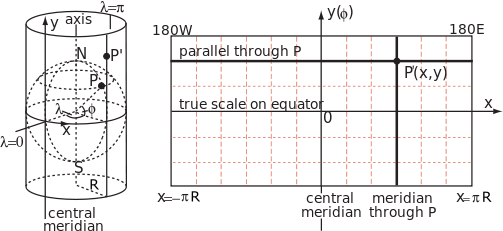
Dodatkowo, tag way zawiera informacje takie jak nazwa drogi czy jej klasa. Z listingów pominięto część informacji nieistotną z punktu widzenia symulacji, takich jak: autorzy, numer wersji czy data ostatniej zmiany.

### Konwersja danych

Podczas uruchomienia program dokonuje się konwersji danych na potrzeby symulacji. Podstawowym zadaniem jest utworzenie pasów ruchu Lane i sieci dróg pobranej z mapy. Problem wymaga wykonania przekształceń geometrycznych z zachowaniem relacji między elementami sieci. W następnych podrozdziałach przedstawione zostały kolejno kroki konwersji mapy.

### Projekcja mapy

Dane w formacie OSM XML opisują pozycję elementów sieci dróg za pomocą długości i szerokości geograficznych. W celu przeprowadzenia symulacji i wyświetlenia reprezentacji tych punktów na ekranie, konieczne jest przeprowadzenie projekcji punktów geograficznych (długość i szerokość geograficzna) na płaszczyznę (punkty x, y). Projekcją wykorzystywaną podczas prezentacji map przez systemy takie jak OpenStreetMap czy GoogleMaps, jest odwzorowanie walcowe równokątne (*Mercator projection*). Zachowuje ona kąty między odcinkami utworzonymi przez południki i równoleżniki. Mechanizm ten ilustruje rysunek 9 (Mercator projection, 2012). Podstawowym ograniczeniem jest spadek dokładności w kierunku oddalania się od równika.



Rysunek 9. Ilustracja odwzorowania walcowego równokątnego

Do przeprowadzenia projekcji użyto biblioteki PROJ.4 (<http://trac.osgeo.org/proj/>) udostępnianej na licencji Apache License, Version 2.0.

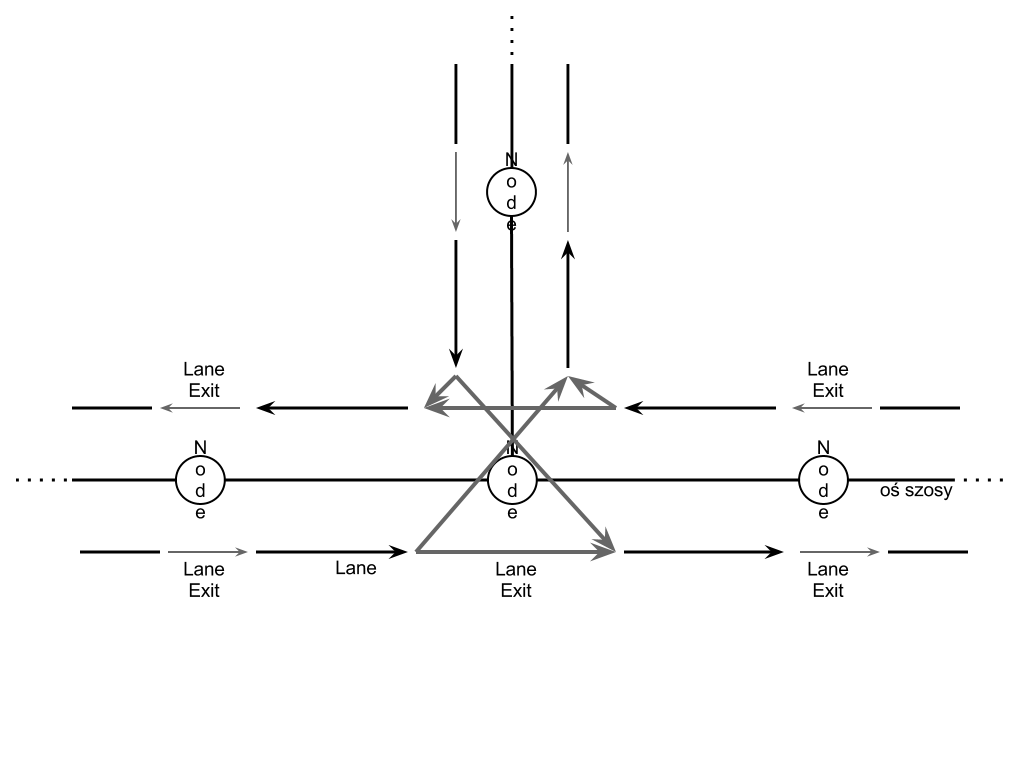
### Normalizacja pozycji elementów

Drugim krokiem przygotowania danych była normalizacja współrzędnych uzyskanych z projekcji mapy. Głównym celem tego działania było zmniejszenie liczbowych wartości współrzędnych dla większej czytelności i zmniejszenia błędu numerycznego. Wykorzystano fakt, iż rozpatrywane punkty znajdują się blisko siebie.

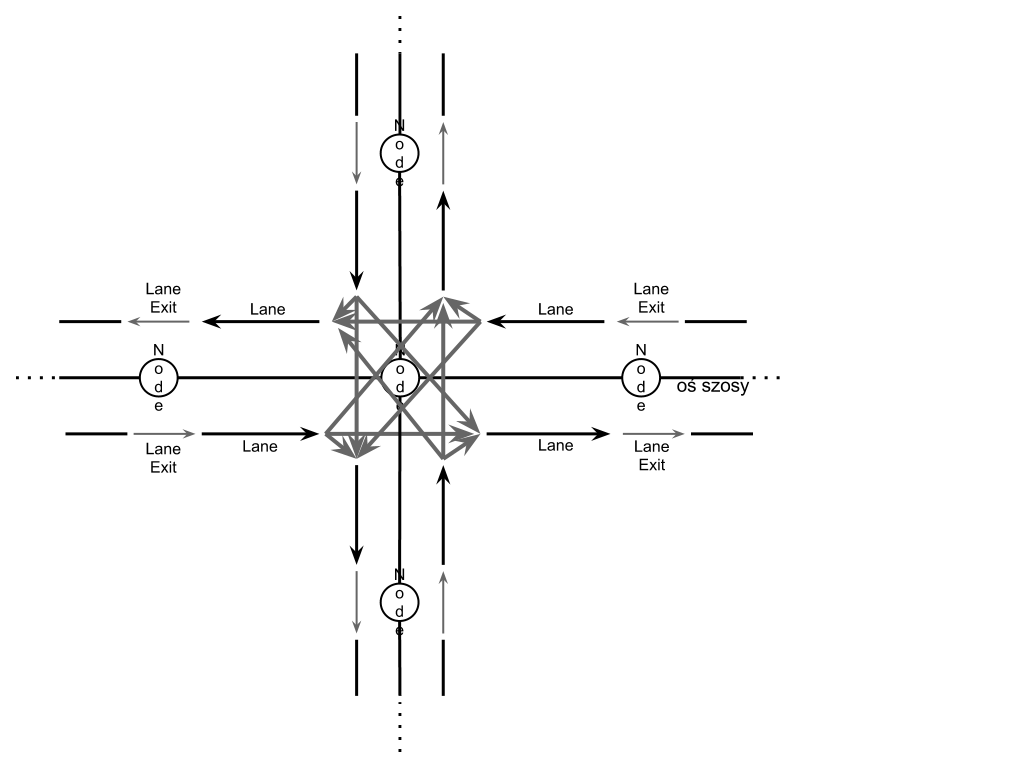
### Budowa sieci połączeń

W efekcie poprzednich kroków uzyskano zbiór elementów Node, zgrupowanych w obiektach Way. Model odzwierciedla teraz dane pozyskane z pliku OSM XML. Na tym etapie konieczne jest utworzenie obiektów Lane i złączenie ich w sieć, po której poruszają się pojazdy.

Szczególnym przypadkiem jest skrzyżowanie i zakończenie drogi (ślepa odnoga). Połączenia budowane w tych sytuacjach przedstawiają rysunki 10 i 11.



Rysunek 10. Obiekty modelu tworzące skrzyżowanie z dwoma wyjazdami



Rysunek 11. Obiekty modelu tworzące skrzyżowanie z trzema wyjazdami

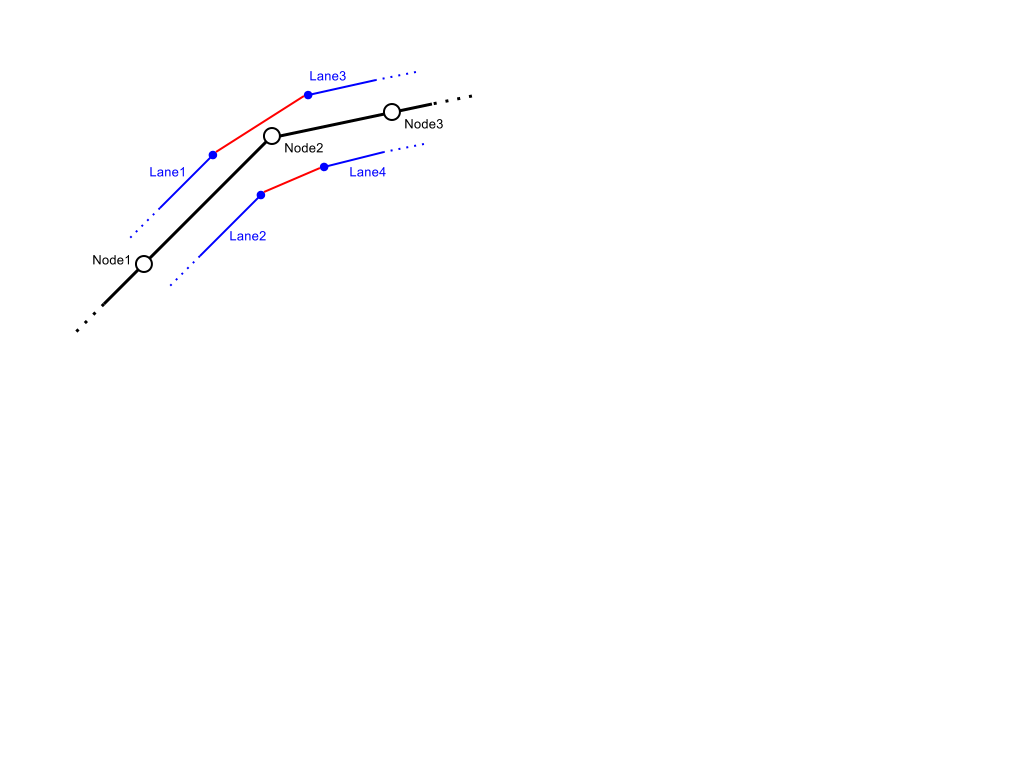
Algorytm tworzenia sieci obiektów Lane przedstawia się następująco: wykonywana jest jedna iteracja po wszystkich elementach Node zawartych w każdym z obiektów Way. Podczas niej tworzone są pary obiektów Lane odpowiadające połączeniu między dwoma obiektami Node (w obu kierunkach). Kolejnym krokiem jest zawiązanie połączeń z już istniejącymi obiektami Lane wchodzącymi i wychodzącymi z obiektów Node. W przypadku skrzyżowania, jeden element Node jest rozpatrywany dwukrotnie, ponieważ zawiera się w dwóch obiektach Way. Po pierwszej iteracji tworzone obiekty reprezentują drogę bez rozgałęzień. Druga iteracja tworzy dodatkowe obiekty Lane i uzupełnia brakujące połączenia.

1. for (Way way : map.ways) {  
2. Node lastNode = null;  
3. for (Node n : way.nodes) {  
4. if (n == way.nodes[0]) { *// pierwszy element drogi*  
5. lastNode = n;  
6. } else { *// kolejne elementy*  
7. Lane laneForward = new Lane(lastNode, n);   
8. Lane laneBackward = new Lane(n, lastNode);  
9.   
10. for (Lane l : lastNode.enters)   
11. l.exits.add(new LaneExit(laneForward));  
12.   
13. for (Lane l : lastNode.exits)   
14. laneBackward.exits.add(new LaneExit(l));  
15.   
16. for (Lane l : n.enters)   
17. l.exits.add(new LaneExit(laneBackward));  
18.   
19. for (Lane l : n.exits)   
20. laneForward.exits.add(new LaneExit(l));  
21.   
22. lastNode.exits.add(laneForward);  
23. n.enters.add(laneForward);  
24.   
25. n.exits.add(laneBackward);  
26. lastNode.enters.add(laneBackward);  
27. }  
28. lastNode = n;  
29. }  
30.}

**Listing 1. Algorytm tworzenia sieci obiektów Lane**

### Znalezienie pozycji obiektów Lane i długości połączeń

Obiekty Lane wymagają przypisania właściwych współrzędnych, aby prawidłowo symulować ruch pojazdów. Przyjęcie pozycji obiektów Node, jako pozycji obiektów Lane powodowałoby "przeskakiwanie" skrzyżowań i połączeń między fragmentami drogi. Ustalenie stałej wartości długości łącznika powoduje, że pojazdy wydają się zmieniać prędkość, ponieważ odległość na ekranie nie odpowiada odległości wewnątrz symulacji. Jak widać na rysunku 12 odległości łączące obiekty Lane1 z Lane2 i Lane2 z Lane4 są różne.



Rysunek 12. Ilustracja zagadnienia długości połączeń między obiektami Lane

## Symulacja

### Konfiguracja

### Krok symulacji

W każdym kroku symulacji, program iteruje po wszystkich pojazdach. Dla każdego samochodu znajduje podstawowe parametry konieczne do obliczenia zmiany prędkość w czasie :

przyspieszenie samochodu w normalnych warunkach, konfortowe dla kierowcy

opóźnienie samochodu przy chamowaniu

akutalna prędkość pojazdu

prędkość pożądana (pusta droga)

eksponenta przyspieszenia

- dynamicznie określana pożądana odległość od przeszkody

- odległość od najbliższej przeszkody lub pojazdu

- bezpieczna odległość między zderzakami pojazdów

- zapas czasu (margines bezpieczeństwa)

- różnica prędkość pojazdów

Parametry mogą być indywidualnie określane dla każdego z pojazdów.

Określana jest zmiana prędkości:

gdzie

#### Prędkość ( IDM - implementacja)

### Zachowanie na skrzyżowaniu

Dla uproszczenia modelu, wszystkie przecięcia dróg traktowane są, jako skrzyżowania typu *all-way stop*. Ten typ skrzyżowania jest najczęściej spotykany w USA i Kanadzie. Oznaczany znakiem stopu z dodatkową tabliczką: *"all way"* lub "*four way"* (All-way stop, 2012). Według regulacji zawartych w *Federal Highway Administration's Manual on Uniform Traffic Control Devices* (MUTCD) pojazd zbliżający się do takiego skrzyżowania powinien zatrzymać się niezależnie od innych czynników. Pojazdy ruszają w kolejności, w jakiej zbliżały się do przecięcia dróg. Skrzyżowania tego typu stosuje się na drogach niskiej kategorii, w terenach mieszkalnych.

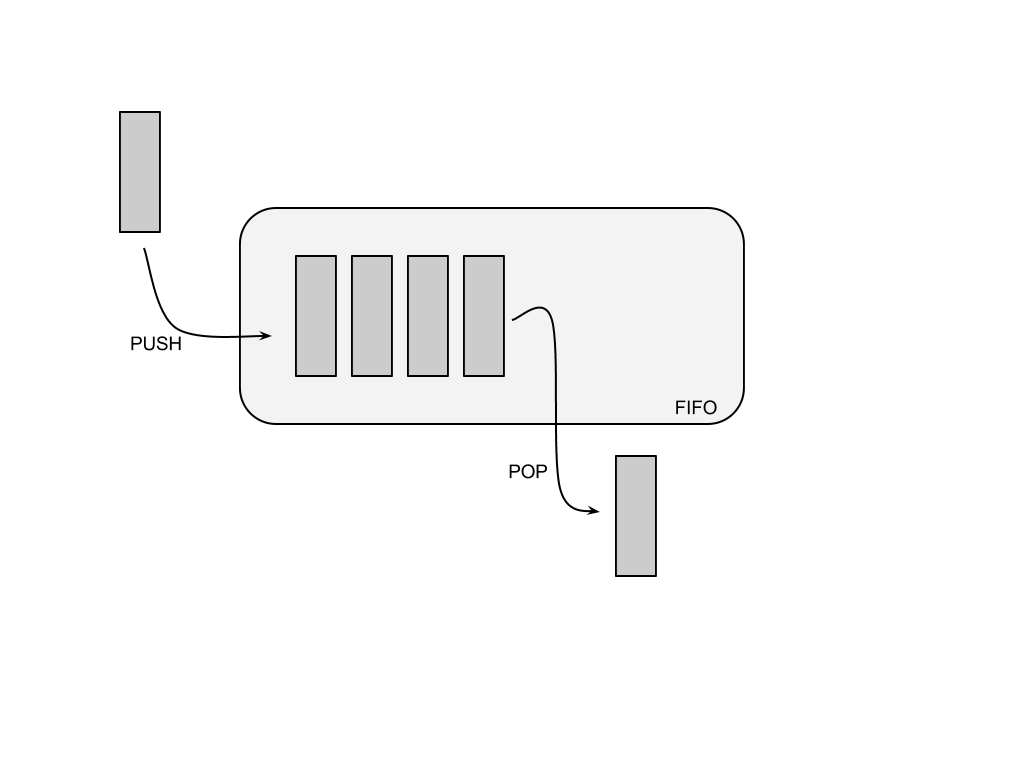
Zaletami skrzyżowania typu *all-way stop* są:

* Niska prędkość pojazdów w okolicy skrzyżowania, pozwalająca na dobrą ocenę sytuacji przez kierowcę
* Zwiększenie bezpieczeństwa pieszych
* Uspokojenie ruchu

#### FIFO

Strukturą opisującą kolejność pokonywania skrzyżowania przez pojazdy, jest kolejka FIFO (*First in, first out)*. Jest ona jednym z podstawowych konceptów informatyki. Skrótem tym określa się zachowanie struktury danych, w której elementy przechowywane najdłużej są usuwane w pierwszej kolejności. Operację dodana elementu do kolejki określamy, jako *push*, natomiast pobrania elementu *pop*.

Zachowanie kolejki FIFO ilustruje rysunek 13.



Rysunek 13. Kolejka FIFO

### Raporty

## Wybrane technologie

### Java

### XML

### GIT

### HTML

## Wnioski z implementacji

# Wyniki

4.1. Przyspieszanie, przeszkoda, zwalnianie

4.2. Skrzyżowanie

4.3. Osiedle

# Bibliografia

*All-way stop*. (2012). Pobrano 4 listopada 2012 z lokalizacji wikipedia.org: http://pl.wikipedia.org/wiki/All-way\_stop

*Mercator projection*. (2012). Pobrano 5 listopa 2012 z lokalizacji wikipedia.org: http://en.wikipedia.org/wiki/Mercator\_projection

*Traffic Simulation*. (2012). Pobrano 4 listopada 2012 z lokalizacji wikipedia.org: http://en.wikipedia.org/wiki/Traffic\_Simulation

Daimin Tang, X. L. (2010). Microscopic traffic simulation oriented road network data model. *Future Computer and Communication (ICFCC)*, (strony 87 - 91).

Dongxin, L. Z. (2010). Simulation of traffic flow on the basis of Road Network-Based Cellular Automata. *Computer Application and System Modeling (ICCASM)*, (strony 282 - 288).

Fishwick, P. (1996). Computer simulation. *Potentials, IEEE*, 24-27.

M. Rickert, K. N. (1996). Two lane traffic simulations using cellular automata. *Physica A: Statistical Mechanics and its Applications*, 534 - 550.

Maurice J. Khabbaz, W. F. (2012). A Simple Free-Flow Traffic Model for Vehicular. *Intelligent Transportation Systems, IEEE Transactions*, 1312 - 1326.

Tao Yang, C. L. (2004). Nonlinear Simulation for Street Network Traffic. *IEEE International Conference on Systems, Man and Cybernetics*, (strony 6232 - 6236).

Wolfram, S. (1994). *Cellular Automata and Complexity: Collected Papers.*

Y. Weng, T. W. (2001). Car-following model of vehicular traffic. *Info-tech and Info-net.* Beijing.