Optimizarea traficului folosind Reinforcement Learning

1. Overview

Optimizarea se face in principal prin calibrarea dinamica a timpului pentru fazele semafoarelor din intersectii.

Lucrarile din literature abordoneaza una sau mai multe intersectii in acelasi timp. Algoritmii folosite sunt de la Q-learning tabular pana la metode de tip Policy optimization folosind variatii ale metodei Actor-Critic.

Lucrarile abordeaza optimizarea si evaluarea pe 2 nivele:

(a) Microscopic (vehicule individuale), in mod deosebit cand se incearca optimizarea unei singure intersectii

(b) Macroscopic – divizeaza spatiul (e.g. benzi de circulatie, strazi) in sectoare si agregeaza inputul si deciziile pe aceste sectiuni.

2. Dataset-uri, simulare, evaluare metode.

2.1 Discutie despre generarea de date si simulare.

In literatura, orasele sunt considerate ca detin date ca :

- Senzori montati care capteaza datele despre traffic. E.g. specializati pentru masirarea densitatii si vitezei autovehiculelor pe anumite benzi de circulatie, intersectie.

- Camere video care permit segmentarea imaginilor si intelegerea densitatii/aglomerarii din intersectii in mod automat

- Date statistice privind populatia, traseele indivizilor, densitatea vehiculelor ca medie pe diferite artere de circulatie, distributii cu numarul de masini care traverseaza intersectii in diferite sensuri, etc.

In general, antrenarea modelelor se face folosind simulatoare folosind distributii de date cat mai reale. Un rezumat al acestor distributii de date, recomandari dar si unelte de simulare si modelare este publicat in [X17]. Cateva modele pentru forecasting sunt prezentate mai jos.

A. Modelul "***Four-Step***" [X18], X[19] este unul din cele mai folosite pentru replicarea distributiei de date a transportului folosind zone ale globului impartite in TAZ ("traffic analysis zones"). Modelarea transportului in aceste zone se face folosind date demografice, activitati (scoli, universitati, locuri de munca, magazine, etc), retelele de transport din zona, momente de timp, etc.

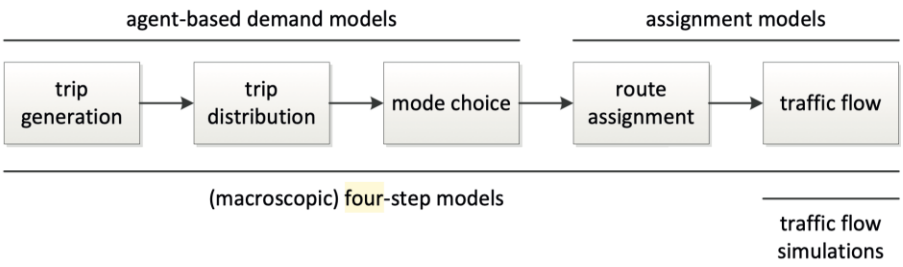


Figura ce explica cei 4 pasi (descrisi mai jos), preluata din [X17]

Cei 4 pasi sunt:

a) Trip Generation

Folosind date demografice, factori soci-economici din zona respectiva, se vor genera rute - distributii de posibile drumuri sursa-destinatie.

b) Trip Distribution

Date posibilele activitati in punctele sursa-destinatie ale rutelor si costul acestora, se vor genera distributii ce prezic cat de accesate vor fi acestea.

c) Mode Choice

Calculeaza mai detaliat distributia folosirii rutelor de mai sus folosind diferite moduri de transport (pedestrian, car, bicicleta, etc).

e) Route Assignment

Aloca efectiv calotarii intre surse si destinatii dat fiind un mode. Se aplica principiul Wardrop (echivalent cu echilibrul Nash), in care fiecare sofer (sau grup) alege calea cea mai scurta de calatorie, stiind ca si ceilalti soferi vor face acelasi lucru. Insa, problematic este faptul ca timpul de calatorie este in functie de cerere, iar cererea este in functie de timpul de calatorie (asa numita problema bi-level.)

B. Micro-similation models

Modelul de *Four-step* este unul macroscopic facand distributii statistice la nivel de rute. De aceea, se pierd informatii low-level cum ar fi: diferite restrictii are drumurilor, tipurile de vehicle indviduale care se aflau pe o anumite artera la momente de tip, coordonatele GPS individuale sau acceleratiile – in general informatii la nivel de participant la traffic. Desi este mai costisitor ca simulare, modelul de tip Micro poate aduce detalii in plus necesare algoritmilor de optimizare.

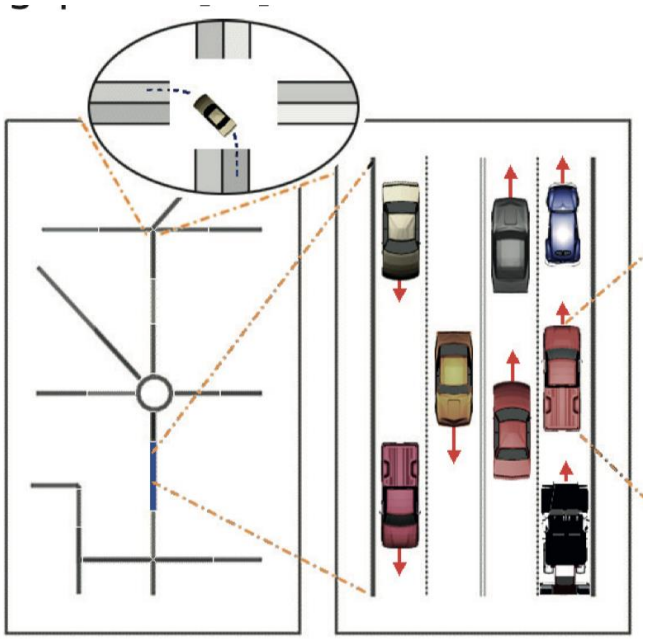
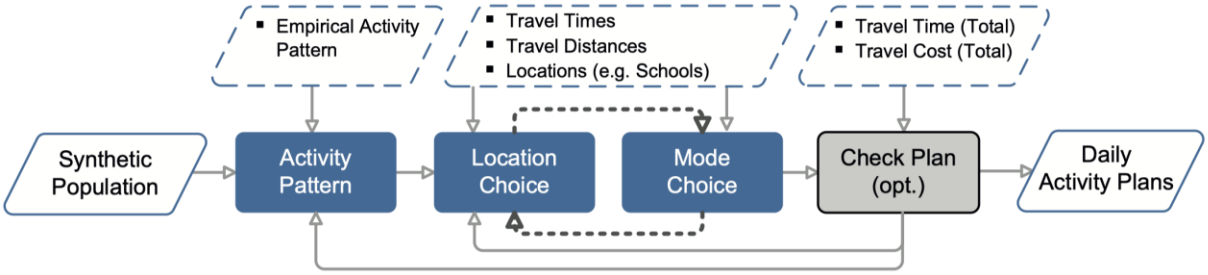


Figura ce explica cele 3 nivele de modelare a cererii de trafic: macroscopic in stanga, microscopic in dreapta si sub-microscopic in cerc, preluata din [X17].

C. Agent-Based Demand

Acest tip de modelare isi propune sa genereze folosind la nivel de agenti individuali (sample the agenti din zone) posibile traictorii folosind patternuri precedente. Pentru mai multe detalii puteti consulta [X17] pentru un revew si figura de mai jos, dar si [X20].



Facand media apoi intre agenti de un anumit tip, putem afla care sunt planurile agentilor in functie de timp, zona si genera trasee cat mai realiste.

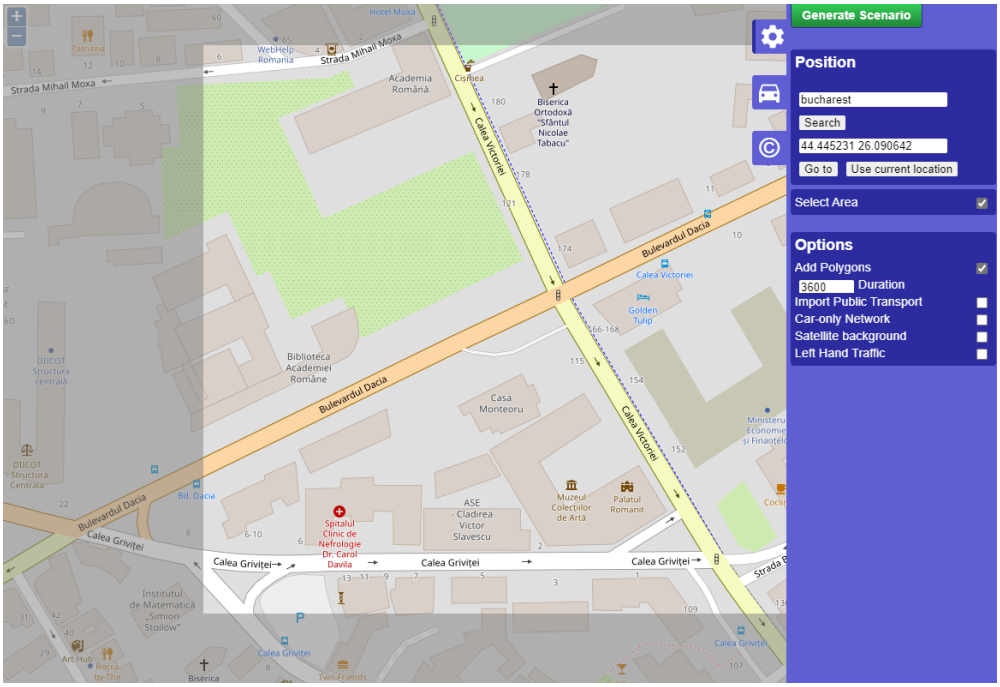
Dupa un studiu aprofundat, noi am ales ca simulator SUMO [X6]. Pentru vizualizarea rezultatelor in 3D si folosirea feature-urilor de computer vision, asa cum am scris si in sectiunea de propuneri cred ca [X3] ar fi cel mai adecvat, avand un plugin de SUMO, ambele fiind descries detaliat in acest capitol.

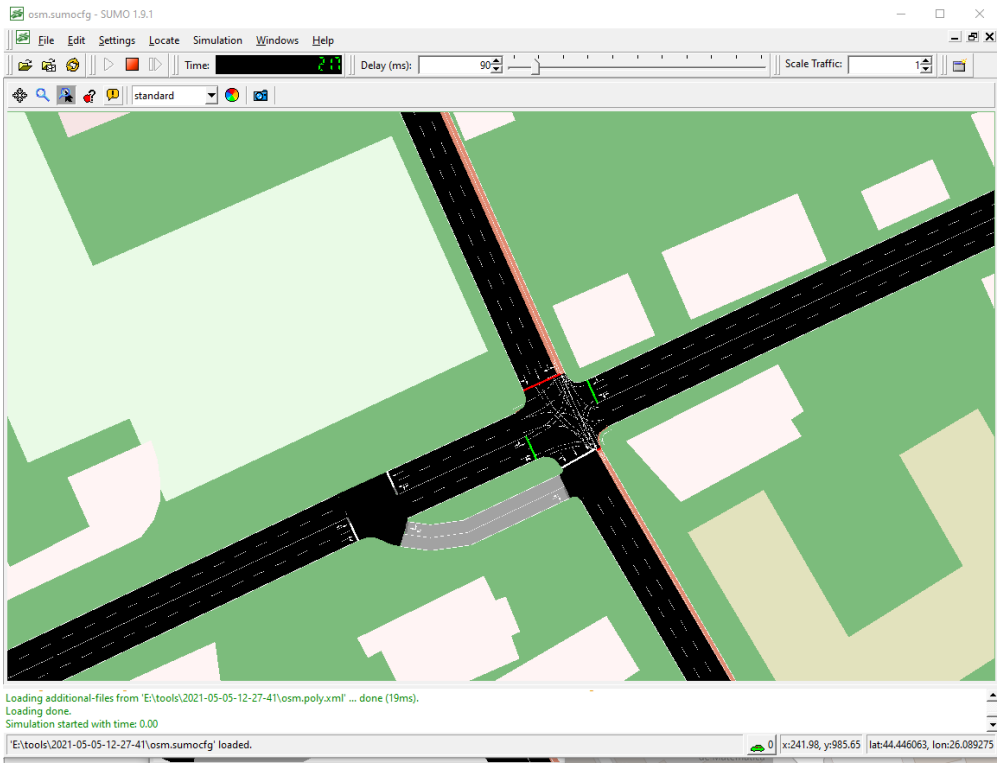
Acestea folosesc drept input pentru algoritmii de clasificare, optimizare a mangementului de traffic.

2.2 Descriere tehnica succinta despre capabilitatile SUMO, OSRM si Carla si cum planuim sa le folosim.

Ce putem face cu SUMO pentru a simula date reale?

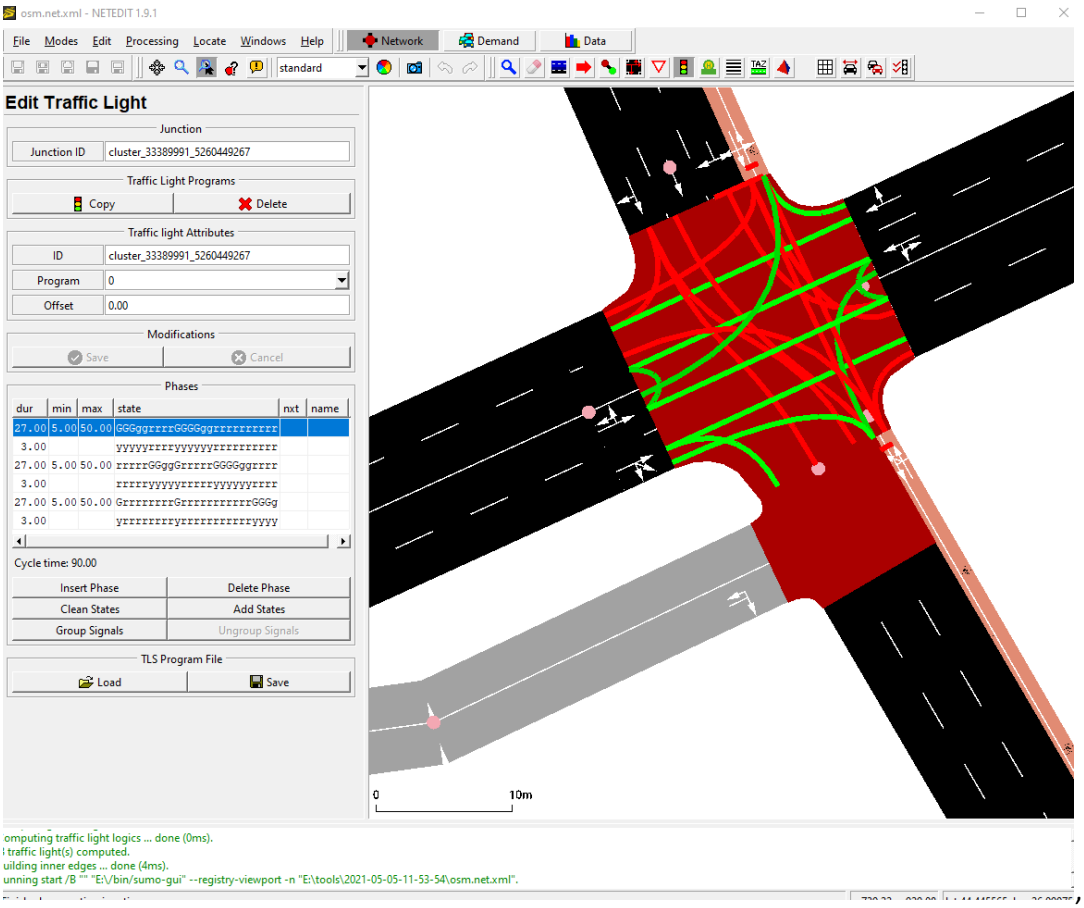
Exemplul A. putem folosi OSM (Open Street Map) [X11] pentru a importa harti reale in simulator:





Exemplul B (aceeasi intersectie de mai sus cu zoom-in):

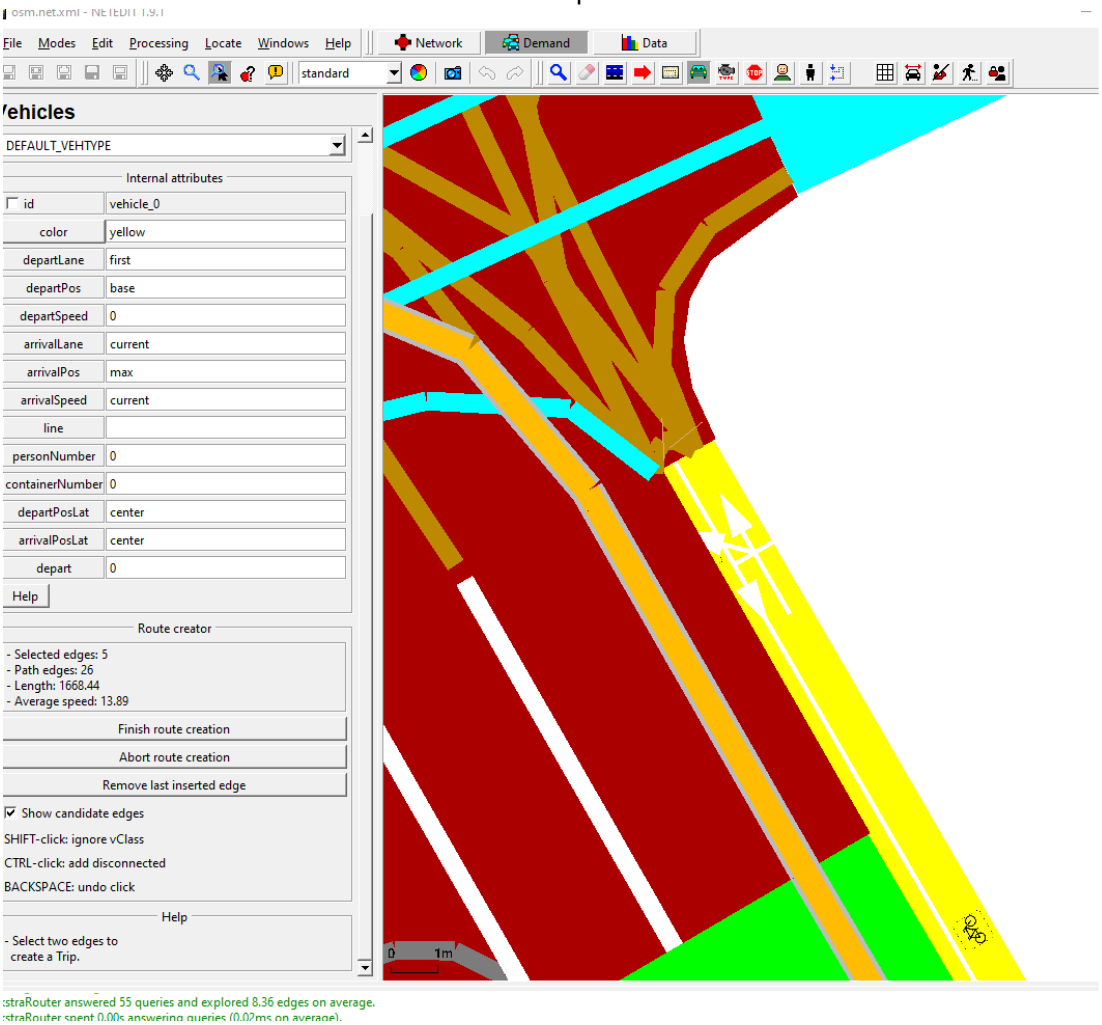
Putem controla timing-ul semafoarelor, atat **static** ca in figura de mai jos pe fiecare sens si banda:



, dar si **dinamic** folosind SUMO drept sever si controland semaforizarea cu un script de Python (https://sumo.dlr.de/docs/Tutorials/TraCI4Traffic_Lights.html) . Acest lucru ajuta foarte mult pentru optimizarea dinamica a metodelor de optimizare a fluxlui si simulare, asa cum este necesar domeniului de reinforcement learning.

Exemplul C:

Putem crea folosind “Demand” fluxul individual pe anumite artere.



De asemenea, este permisa:

- crearea sau editarea benzilor fiind disponibila

- marcarea acestora doar pentru anumite tipuri de entitati, e.g. biciclisti sau pietoni, mijloace de transport, etc.

Exemplul 4:

Putem apoi importa si simula traficul nu doar in 2D folosind SUMO ci si in 3D folosind Carla: https://carla.readthedocs.io/en/0.9.7/how_to_make_a_new_map/ . Cele doua unelte pot fi conectate prin socketi.

Modelarea cererii de flux (*Demand model)* in SUMO

O descriere tehnica completa despre capabilitati este aici
https://sumo.dlr.de/docs/Demand/Introduction_to_demand_modelling_in_SUMO.html

Pe scurt, putem:

- Plasa **manual** valorile in scop de testare (https://sumo.dlr.de/docs/Demand/Shortest_or_Optimal_Path_Routing.html si https://sumo.dlr.de/docs/Demand/Dynamic_User_Assignment.html , https://sumo.dlr.de/docs/Demand/Automatic_Routing.html ,
- **random** https://sumo.dlr.de/docs/Tools/Trip.html#randomtripspy
- La nivel **macroscopic** , putem simula induction loops (dfrouter + https://sumo.dlr.de/docs/Demand/Routes_from_Observation_Points.html) sau prelua din date reale cum sunt descries in dataset-urile de mai jos.

  De asemenea trebuie avut in vedere conceptul de "Fundamental diagram of traffic flow" care leaga fluxul de traffic (numarul de vehicule pe ora) de densitate. Poate prezice astfel capabilitatile unei retele stradale in general sau atunci cand sunt restrictionate limite de viteza [X21].

- La nivel **microscopic** putem fie genera demand generand populatii statistice (modelul four-way descries mai sus), folosind https://sumo.dlr.de/docs/Demand/Activity-based_Demand_Generation.html , sau prelua aceste date din realitate folosind dataset-urile de mai jos.
- La nivel **sub-microscopic**, putem folosi https://sumo.dlr.de/docs/jtrrouter.html pentru a realiza distributii de destinatii in intersectii.
- Folosirea matriciler Origin-Destination (OD_matrices).
- Alte surse de date enumerate si simulari de blocaje in traffic, safety, randomness in intersectii, etc: https://sumo.dlr.de/docs/index.html#demand_modelling

OSRM [X15] este un engine de routare care indica drumul pentru a ajunge optim dintr-o locatie A in locatia B folosind hartile din OSM pentru diferite tipuri de entitati: pietoni, masini, biciclete, trenuri. De asemenea suporta restrictii de viteza si multe alte setari. Este un engine rapid scris in C++ folosind mechanisme de caching ale cailor (Contraction hierarchies [X16]

2.3 Discutie despre dataseturi si cum pot fi folosite in studiul nostru

A. *MatSim data* [X12]

Folosind un plugin intern, SUMO poate incarca datele din MatSim. Datele folosite despre traficul pietenilor, masinilor, orarul autobuzelor, etc sunt disponibile din mai multe regiuni ale globului: https://www.matsim.org/gallery/ . Practic aceste date contin harta si informatiile statistice despre

traficul din fiecare regiune. Folosind acestea, putem aplica metodele de optimizare pe sub-zone in interiorul simulatorului SIMO, unde metoda noastra ghideaza astfel traficul folosind scripturi Python.

Spre exemplu din regiunea Ile-de-France: https://github.com/eqasim-org/ile-de-france/blob/develop/docs/simulation.md. Conține codul și datele pentru a crea populații sintetice din diferite zone ale lumii care reprezintă îndeaproape atributele socio-demografice ale persoanelor și gospodăriilor din regiune, precum și tiparele lor de mobilitate zilnică (cum ar fi locul de muncă, educație, cumpărături, etc) și care sunt conectate prin călătorii cu un anumit mod de transport. Se știe din date reale statistice când și unde se întâmplă aceste activități.

O astfel de populație sintetică este utilă pentru multe aplicații de cercetare și planificare. În special, o astfel de populație sintetică servește ca element de intrare la simulările de transport pe bază de agenți, care simulează comportamentul zilnic al mobilității oamenilor pe o scară detaliată spațial și temporal. Mai mult, astfel de date au fost utilizate pentru a studia răspândirea bolilor sau plasarea de servicii și facilități.

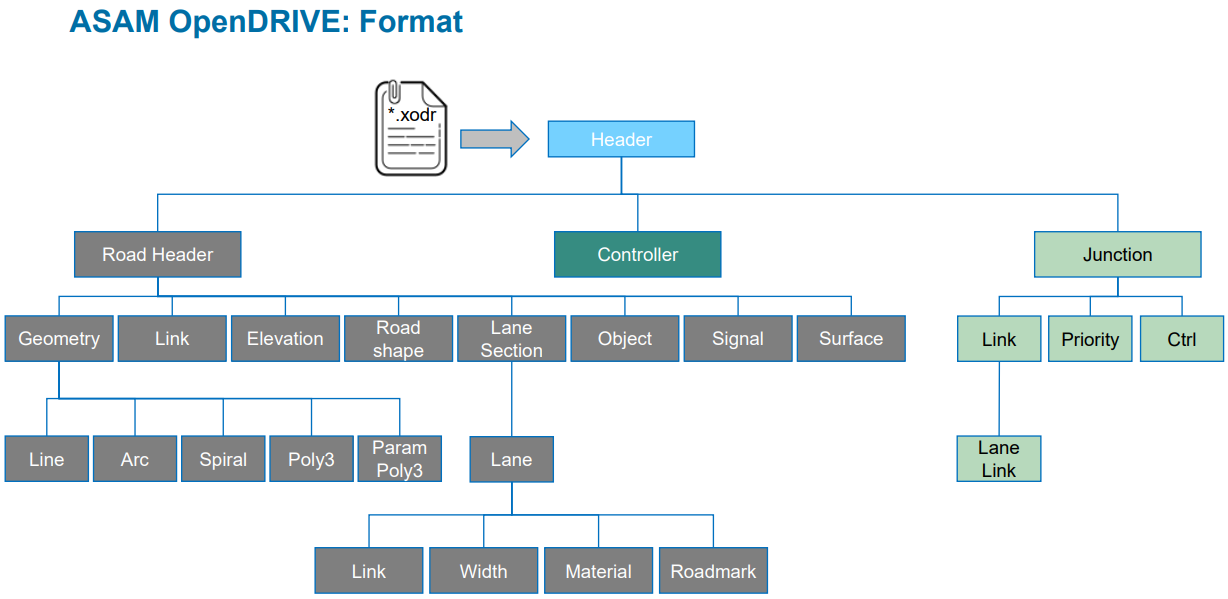
C. Caltrans – Deja amimtit in sectiunea A. Datele sunt asemanatoare cu cele care am putea sa le obtinem de la Primaria Bucuresti s iar putea fi integrate in SUMO direct ca date statistice corelate cu intersectii fizice reale.

D. OpenStreetMap (OSM).

Putem importa orice regiune a lumii dorim folosind OSM si apoi crea distributii statistice de tipul Poisson pentru diferite rute / intersectii pentru a demonstra capabilitatile algoritmilor nostri in comparative cu altele.

E. ASAM OpenDrive (X[13])

Diferenta esentiala fata de datele din OSM este aceea ca OpenDrive ofera informatii suplimentare despre geometria terenului, a benzilor si alte lucruri aditionale cum ar fi semnele de circulatie, etc. Este folosit direct in practica pentru invatarea modelelor de navigatie autonoma in spatii reale 3D. Acestea vin intr-un format xodr ce poate fi important direct in Carla [X11] putat fi apoi antrenate. **Cu aceasta metoda putem astfel invata atat din datele de traffic cu distributii reale, dar si folosind computer vision pentru semne de circulatie, geometria drumului, etc.**



(Except din manualul OpenDrive [X13]).

F. Bolt dataset

(pending acceptance)

Datele furnizate de Bolt (subsect cu ce cele interesante pentru optimizarea traficului, nu a castigului generat) sunt date atat micro cat si macro scopic:

A. La nivel microscopic, ele vin de la fiecare sofer Boltsub forma unor time-series unde pentru fiecare intrare avem:
   - timestamp (~2 secunde)
   - locatie X,Y (long, latt)
   - orientare
   - acceleratie

B. La nivel macroscopic:
   - Viteze medii pe secvente de drumuri, raportate la perioade de timp, evenimente.
   - Pentru intersectii avem date distributii de traversare in diferite directii (e.g. stanga, dreapta, intoarcere).

Cei de la Bolt au create un simulator propriu pentru a testa algorimtii cu un review in [X14], care incearca optimizarea castigurilor generate. Insa noi putem folosi datele selectate mai sus din A si B cu SUMO pentru a optimiza per total traficul. **De asemenea, exista posibilitatea corelarii datelor de flux folosind senzorii de inductie din seturile precedente cu cel din Bolt pentru a avea metode ce inteleg mai bine pattern-urile mai bune la nivel microscopic sau hybrid.**

Interesant este de asemenea si cum folosind un model Markov pentru estimarea timului de ajungere la destinatie folosind routarea data de standardul OSRM [X15] intre doua locatii ale globului si stiind datele furnizate in B.

3. Review detaliat al metodelor de reinforcement learning folosite in state-of-the-art

Una din primele lucrarii care folosesc reinforcement learning pentru decongestionarea traficului prin controlul intelligent al semafoarelor este [X1], care seteaza de altfel si fundamentul pentru analiza si evaluarea rezultatelor pentru un astfel de sistem. Pe scurt, reteaua stradala de intersectii din oras este creata ca un graf G in care avem noduri:

(a) noduri de tip E - de intrare a masinilor in sistem pentru simulare

(b) noduri de tip intersectii semaforizate (fiecare controleaza o singura intersectie).

Simulatorul creat de autori foloseste diferite distributii pentru a genera noi vehicule in sistem in timpul simularii. **Starea** fiecarei vehicul este definita de o pereche $S = (P, D)$, reprezentand: P – pozitia curenta din infrastructura de simulare si D – punctul de destinatie.

Actiunile din sistem sunt date de comutarea semafoarelor de oras intre culorile red/green. Astfel, problema de optimizat in limbaj de reinforcement learning pentru gasirea valorii optime de asteptare a masinilor la semafor este data de perechile stare-actiune a tuturor masinilor din G pe o durata definita de timp. **Reward**-ul este -1 daca masina trebuie sa astepte si 0 daca poate merge mai departe. Autorii defines aceste perechi ca :

$$Q(s, red) = R(s, s) + \gamma V(s) = 1 + \gamma V(s)$$

$$Q(s, green) = \sum_{s'} P(s, s')(R(s, s') + \gamma V(s'))$$

,

unde R(s,s') este reward-ul pentru a trece din starea s in s', P(s,s') este probabilitatea de a trece din starea s in s', iar V(s) este media valorilor din starea s, mai precis pentru ca avem doar doua stari (red/green).

$$V(s) = P(green|s)Q(s, green) + P(red|s)Q(s, red)$$

.

In simularea autorilor, probabilitatile de tranzitie sunt obtinute efectiv din contorizari si pe masura simularii se updateaza acest sistem tabular pentru valori. Obtinem astfel o politica a valorilor pentru fiecare state-action. De aici, autorii folosesc o politica de shortest-path (best-first algorithm cu o eroare de 10%) pentru alegerea drumului (si benzilor) optime de deplasare al autovehiculelor.

Desi aplicata initial pentru decongestionarea traficului pe autostrazi, ideile din [X3] pot fi adaptate pentru optimizarea flow-ului de traffic din oras in mod deosebit la intrarile iesirile pe poduri, sau la intersectia dintre artere de circulatie mari cu unele mai mici. Lucrarea se bazeaza pe idea de limitare a

vitezei pe anumite sectiuni pentru a evita supra-aglomerarile. divizeaza o autostrada in N sectiuni si considera viteze v_i medii de pe fiecare sectiune, precum si densitatea k_i de pe acea sectiune. Metrica principala de evaluare este timpul total de deplasare al autovehiculelor. Se incearca astfel minimizarea timpului petrecut in traffic. Pentru ca ecuatiile din lucrare sunt speciale gandite pentru autostrazi, voi enumera doar principalele idei si cum le putem adapta pentru cazul de optimizare dintr-un oras. Autorii descriu problema ca un Markov Decision Process si folosesc apoi Q-learning pentru optimizarea metricii mentionate. In acest sistem, **state**-ul la momentul t este reprezentat in principal de $s_t = (v_i, k_i)$, unde i itereaza pe spatial tuturor sectiunilor strazii. **Reward-ul** considerat este 0 daca viteza minima de pe toate sectoarele este mai mare decat un threshold definit, altfel negativ in functie de timpul de asteptare al masinilor de pe toate sectiunile, intre 2 momente consecutive de timp:

$$r_t = \begin{cases} 0 & \min\{v_i((t+1)c) \mid i = 1, \ldots, N\} > u \\ -h(tc, (t+1)c) & \text{otherwise} \end{cases}$$

$$h(b, e) = T \sum_{p=b}^{e} \left(\sum_{i=1}^{N} [M_i L_i k_i(p)] + \sum_{i=0}^{N} w_i(p) \right)$$

, unde M_i, L_i $w_i p$ reprezinta numarul de sectiuni, benzi (lanes), segmente, lungimile cozilor de asteptare de la fiecare intrare/iesire pe artera.

Actiunile sunt reprezentate de intervalele de viteza limita aplicate pe sectiuni, cu scopul de a obtine reward-uri cat mai bune. Avem astfel un sistem prin care putem estima Q(state, action) din date reale si gasi o politica greedy care sa fixeze limitele de viteze in mod dinamic. Chiar daca limitarea de viteza nu se poate realiza, se poate calcula invers in cazul unui oras: cum setezi timing-ul semafoarelor a.i. sa obtinem aceasta limita de viteza daca am considera ca autoturismele se deplaseaza cu viteza legala.

In [X4], fata de [X3], autorii propun ca noutate o reprezentare mai buna a starii spatiului pentru a folosi Deep Reinforcement Learning de aceasta data. Initial, ei descompun similar oarecum cu ideea din [X2] benzile de circulatie in segmente de o capacitate maxima pentru discretizarea spatiului fizic. Formal, **starea** in lucrarea respective devine:

$S \in (\mathbb{B} \times \mathbb{R})^{\frac{l}{c} \times n} \times P$, unde: *l* este lungimea unei benzi, *c* lungimea unei celule in care s-a divizat spatial fizic al celulelor, n este marimea spatiului fizic (un grid); *B* este o valoare Boolean reprezentand daca exista o masina in locatia respectiva sau nu, *R* este viteza masinii din acea celula, iar P este considerat Phase-ul la momentul de timp respectiv. Daca consideram *A* multimea actiunilor din sistem, enumerate in tabelul de mai jos (table preluat din [X4]), atunci $P \in B^{|A|}$

TABLE I: Traffic Signal Phase Action Transitions

		Selected Action			
		NSG	**EWG**	**NSLG**	**EWLG**
Current Traffic Signal Phase	**NSG**	-	{NSY, R}	{NSY}	{NSY, R}
	EWG	{EWY, R}	-	{EWY, R}	{EWY}
	NSLG	-	{NSY, R}	-	{NSY, R}
	EWLG	{EWY}	-	{EWY. R}	-

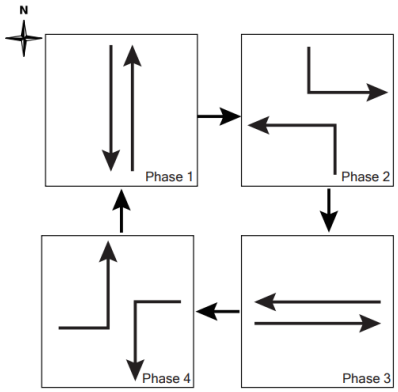
In acest tabel este reprezentata compatibilitate tranzitiilor dintr-o stare in alta (**actiunile agentului),** iar N,S,E,W sunt punctele cardinale, G/R/Y sunt starile semaforului din intersectia analizata. Deci spatiul actiunilor este dat de $A = \{NSG, EWG, NSLG, EWLG\}$. **Reward-ul** la un moment de timp t este dat de

diferenta delay-urilor care apar in trafic intre starile s_t s_{t+1}, i.e. deci cum afecteaza actiunea luata luata momentul t, a_t timpii de asteptare de la semafor.

O noutate a acestei lucrari este faptul ca foloseste Deep Reinforcement Learning, mai precis 2 retele diferite folosind layere convolutionale + fully connected pentru a facem un mapping din starea $s_t \in S$ la valoarea actiunii intr-o anumita stare, i.e. Q(s, a). Autorii folosesc simulator SUMO [X6] cu diferite probabilitati de distributie a aparitiei masinilor in intersectii (Inverse Weibull, Burr). Desi nu specifica concret, autorii par sa foloseasca o versiune de A3C [X7] pentru paralelizarea adunarii experientelor din mediul simulat

Lucrarea din [X8] observa ca lucrarile precedente se bazeaza doar pe optimizari locale din intersectii si propune o extensie a metodelor pentru a putea gestiona simultan mai multe intersectii in acelasi timp. **State** space-ul este modificat, astfel incat considera ca fiecare intersectie are senzori pentru a capta 2 tipuri de informatii: numarul de vehicule in asteptare si viteza medie a vehiculelor, pe fiecare sens de mers. Avand maxim 4x4 sensuri, avem deci in fiecare intersectie un state compus din 2x4x4 valori numerice. Consideram ca grid-ul orasului este astfel descompus intr-o matrice WxHxC cu abstractizarea de mai sus, ce serveste ca input pentru modelele de RL.

Spatiul **Actiunilor** este compus din 4 faze prezentate in figura de mai jos, reprezentand toate starile posibile ale semafoarelor intr-o intersectie (note: oarecum mai incomplet decat in [X4]).



Agentul (sistemul de semafoare dintr-o intersectie), poate modifica phaseul la urmatoarea unitate de timp (5 secunde minim diferenta intre modificari considerate in paper). Daca avem un numar de intersectii notat cu N_{TLS} , atunci decizia / actiuena agentului este de forma <N_{TLS}, $2 >$, 2-ul indicand probabilitatea de a pastra sau schimba phase-ul curent din intersectie. **Reward-urile** considerate in lucrare sunt de asemenea modificate fata de versiunile precedente. Astfel, avem 2 tipuri de reward:

(A) Reward local: masoara diferenta dintre maximul vehiculelor aflate in intersectie din directia NS si WE:

$$r_t^{\mathrm{TLS}_i} = -\left|\max q_t^{\mathrm{WE}} - \max q_t^{\mathrm{NS}}\right|$$

(B) Reward global: Masoara diferenta dintre cate vehicule au trecut de semafor versus cate au intrat in asteptare la momentul t.

$$r_t^{\text{Global}} = \|\text{Veh}_t^{(\text{out})}\| - \|\text{Veh}_t^{(\text{in})}\|$$

Reward-ul total este calculat cu formula de mai jos, unde β porneste la inceputul antrenarii de la valoarea 0 si ajunge in final la 1 – intuitiv, se incearca optimizarea locala la inceput apoi se merge spre cea globala.

$$r_t = \beta r_t^{\text{Global}} + (1-\beta)\frac{1}{N_{\text{TLS}}}\sum_i^{N_{\text{TLS}}} r_t^{\text{TLS}_i}$$

Algoritmul de invatare foloseste o metoda de tip Actor Critic, unde Criticul are in final ca output valoarea pentru local si global, iar Actorul defineste policul adica vectorul de decizie $<N_{TLS}, 2>$. Ambele fac sharing la un set de layere deep bazate pe blocuri reziduale [X9].

La nivel de metodologie de antrenare, [X5] foloseste o metoda de tip Actor-Critic cu DDPG in timp ce [X8] foloseste Proximal Policy Optimization (PP0) in combinatie cu General Advantage Estimation (GAE), asa cum sustin autorii pentru stabilitatea rezultatelor in timp. Nu am vazut insa o comparatie efectiv intre DDPG si PPO..

Un alt element de noutate in [X10] este trecerea la un sistem multi-agent intre semafoare folosind un algoritm modificat de tip DDPG. Lucrarea propune astfel trecerea de la multi-objective learning (unde reward-ul era un singur vector simuland mai multi agenti), la un sistem multi-agent real. De asemenea, este mentionat faptul ca utilizarea unor layere de LSTM atat pentru Actor cat si pentru Critic stabilizeaza rezultatele dat fiind faptul ca in general agentii au acces partial la observatiile din environment.

4. Propuneri de prototipare, implementari

Scopul principal al implementarii este de a avea un model de reinforcement learning antrenat care pentru orasul Bucuresti (ideal, daca vom avea date), sau orice alta zona invata pattern-urile din traffic folosind retele deep si creaza o politica pentru timing-ul semafoarelor sau produce diferite semnale de traffic pentru optimizarea fluxului.

Vom avea ca distributie finala o metoda open-source utilizabila si adaptabila set-urilor de date enumerate mai sus folosind mecanisme simple de tip proxy.

4.1 La nivel de idei de nisa:

- Optimizari pentru semaforizarea pietonala optionala acolo unde detectam ca un semafor intelligent (apasare buton pietoni ar fi avut sens pentru optimizare). Vezi Blvd. Iuliu Maniu ! (Referinta tehnica pentru implementare in SUMO: https://sumo.dlr.de/docs/Tutorials/TraCIPedCrossing.html)
- Optimizari pentru benzi de circulatie autobuze / mijloace de transport. E.g. cum ar putea la anumite ore sa foloseasca si soferii aceste benzi fara sa incurce mijloace de transport pe

anumite rute. Vezi Centrul, cum banda de biciclete / autobuze e goala de multe ori ! (Referinta tehnica in SUMO:
https://sumo.dlr.de/docs/Tutorials/TraCI4Traffic_Lights.html
)

- Corelarea imaginilor de la camere din orase cu cele statistice date de senzorii de inductie astfel incat sa ne putem prinde ca o banda defapt nu e agglomerate, dar este blocata de o masina parcata neregulamentar si atunci fluxul va fi automat incetinit.
- Din date reale, am putea extrage spre exemplu cum au ajutat agentii de circulatie fluidizarea traficului si cum ar fi cu / fara acestia, unde ar fi mai bine sa ii amplasam pe baza acestor date. Putem modela problema ca un de multi-objective RL.
- Ar fi interesant sa putem punem panouri in oras cu limitatoare de viteze dinamice si coordonare a traficului cu indicatoare, de exemplu: recomandat schimbare banda dreapta / stanga, sub pod etc. Putem astfel recomanda fluidizarea dinamica a traficului si modela tot ca pe o problema de RL.

4.2 La nivel de algoritm / metoda folosind reinforcement learning, metodologie, imbunatatiri incrementale.

- Putem incerca usor [X1, X4] folosind insa datele reale obtinute din traffic in loc de contorizare / simulare. Putem folosi simulatorul Carla [X2] + SUMMO (SUMO co-simulation - CARLA Simulator) pentru a facilita procesul de invatare si vizualizare real-time.
- Ar fi interesant sa incercam [X3] daca avem date pentru intrarile / iesirile de pe poduri.
- Similar cu [X4], putem paraleliza masiv adunarea experientelor folosind o versiune de A3C [X7].
- Putem extinde apoi cu o implementare similara cu [X8] dar folosind date reale si combinatii intre starile din [X8] si [X4]. La fel, spatial starilor din [X4] cred ca este mai potrivit pentru un caz real.
- [X8] foloseste doar 9 intersectii, am putea extinde la un graf mai complex de intersectii problema.
- [X10] pare un sistem mai real, dar foarte greu de antrenat pe un caz complex. Totusi ideea cu LSTM poate fi foarte utila.

4.2 La nivel de dataset, corelatii si combinare nivel macroscopic cu cel microscopic.

Combinarea datelor intre ele din date statistice cum ar fi senzori inductie, MATSim statistical data si datele la nivel microscopic furnizate de soferii Bolt (daca aceste date vor putea fi utilizate macar in scop de sampling).

Referinte

[X1] Marco Wiering, Jilles Vreeken, Jelle Van Veenen, and Arne Koopman. Simulation and optimization of traffic in a city. In Intelligent Vehicles Symposium, 2004 IEEE, pages 453–458. IEEE, 2004.
[X2] Alexey Dosovitskiy and German Ros and Felipe Codevilla and Antonio Lopez and Vladlen Koltun, CARLA: An Open Urban Driving Simulator, 2017, [1711.03938] CARLA: An Open Urban Driving Simulator (arxiv.org).

[X3] Erwin Walraven, Matthijs T.J. Spaan, Bram Bakker, Traffic flow optimization: A reinforcement learning approach, Engineering Applications of Artificial Intelligence Journal, http://dx.doi.org/10.1016/j.engappai.2016.01.001
[X4] W. Genders and S. Razavi, "Using a Deep Reinforcement Learning Agent for Traffic Signal Control," arXiv:1611.01142 [cs], Nov. 2016. [Online]. Available: http://arxiv.org/abs/1611.01142
[X5] N. Casas, "Deep Reinforcement Learning for Urban Traffic Light Control," Master Thesis in Advanced Artificial Intelligence, Department of Artificial Intelligence Universidad Nacional de Educación a Distancia , 2017 Deep Reinforcement Learning for Urban Traffic Light Control (uned.es)
[X6] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO - Simulation of Urban MObility," International Journal On Advances in Systems and Measurements, vol. 5, no. 3&4, pp. 128–138, December 2012.
[X7] Volodymyr Mnih and Adrià Puigdomènech Badia and Mehdi Mirza and Alex Graves and Timothy P. Lillicrap and Tim Harley and David Silver and Koray Kavukcuoglu, Asynchronous Methods for Deep Reinforcement Learning, [1602.01783] Asynchronous Methods for Deep Reinforcement Learning (arxiv.org)
[X8] Yilun Lin, Xingyuan Dai, Li Li, and Fei-Yue Wang, An Efficient Deep Reinforcement Learning Model for Urban Traffic Control, [1808.01876] An Efficient Deep Reinforcement Learning Model for Urban Traffic Control (arxiv.org)
[X9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[X10] T. Wu *et al*., "Multi-Agent Deep Reinforcement Learning for Urban Traffic Light Control in Vehicular Networks," in *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 8243-8256, Aug. 2020, doi: 10.1109/TVT.2020.2997896.
[X11] Open Street Map (OSM) https://www.openstreetmap.org
[X12] MATSim (Multi-Agent Transport Simulation) *[https://www.matsim.org/]*
[X13] ASAM OpenDrive, https://www.asam.net/standards/detail/opendrive/
[X14] Simulating cities for a better ride-hailing experience at Bolt
[X15] OSRM Routing Machine Project OSRM (project-osrm.org)
[X16] Customizable Contraction Hierarchies, [1402.0402] Customizable Contraction Hierarchies (arxiv.org).
[X17] Project SUMBA, GUIDANCE FOR TRANSPORT MODELLING AND DATA COLLECTION, https://www.eltis.org/resources/tools/guidance-transport-modelling-and-transport-data-collection-intermodality
[X18] J. Ortúzar, L. G. Willumsen (2011) Modelling Transport, 4th Edition, ISBN: 978-0-47076039-0.
[X19] D. A. Hensher, K. J. Button (eds.) (2007) Handbook of Transport Modelling, ISBN: 978-008-045376-7, eISBN: 978-0-857-24567-0.
[X20] M. G. McNally, C. Rindt (2012) The activity-based approach. In: Handbook of Transport Modelling, D. A. Henscher, K. J. Button, editors, Elsevier, pp. 53-69.
[X21] Fundamental diagram of traffic flow
https://en.wikipedia.org/wiki/Fundamental_diagram_of_traffic_flow