Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar

Pálos Péter

Decentralizált Multi-Agent megoldások a forgalomirányításban

Tudományos Diákköri dolgozat

Konzulens:

Dr. Huszák Árpád

Budapest, 2020.

Absztrakt

A metropoliszok növekvő népessége a forgalom sűrűségének növekedését is eredményezi, mely a forgalom torlódásához és balesetveszélyes helyzetekhez vezet, ezáltal komoly városüzemeltetési probléma. A dolgozat célja egy olyan előremutató, azonnali beavatkozásra képes, gépi tanuláson alapuló forgalomirányító rendszer kivitelezése, mely a jövőben alkalmas lehet a jelenleg használatban levő módszerek bővítésére illetve helyettesítésére. A dolgozat több Q-learning alapú modellt, nevezetesen a Deep Q-Network, Double Deep Q-Network, Dueling Deep Q-Network és a Double Dueling Deep Q-Network (D3QN) módszereket hasonlítja össze. A kiértékelés a CO_2 kibocsátás, átlagsebességnek, helyben álló járművek számának és a várakozási időnek a mérésével valósult meg. A dolgozat eredményei alapján arra lehet következtetni, hogy egy csomópontos úthálózat esetén a súlyozott utazási szám maximalizálás, míg négy csomópontos úthálózat esetén a sorhossz minimalizálás és átengedő képesség maximalizálás a legcélravezetőbb jutalmazási függvény. Ezen felül érdemes kiemelni, hogy az eredmények alapján a teljes mértékben gépi tanulás alapú forgalomirányítás néhány esetben kiszámíthatatlanul viselkedhet, mely nem megengedhető biztonsági szempontból. Ennek ellenére a jelenleg használt módszerek bővítéseként, valamint jövőbeli fejlesztésével alternatívájaként is használható lehet.

Kulcsszavak: forgalomirányítás, gépi tanulás, deep Q-learning, multi-agent reinforcement learning

JEL klasszifikáció: C45, R41

Tartalom

1 Bevezetés

1.1 Tradicionális forgalomirányítás

A metropoliszok növekvő népessége a forgalom sűrűségének növekedését is eredményezi, mely a forgalom torlódásához, és balesetveszélyes helyzetekhez vezet, ezáltal komoly városüzemeltetési probléma. A forgalomirányítás optimalizálására a közlekedési lámpák hangolásával gazdasági és ökológiai szempontból is szükség van, a megfelelő minőségű városi élet biztosítása érdekében. De hogyan jutottunk el a modern forgalomirányítás korába? Az első elektronikus közlekedési jelzőlámpát 1912-ben fejlesztette ki egy rendőrtiszt Salt Lake Cityben, majd 1914-ben állt forgalomba az első példánya. Érdekesség, hogy a korai forgalmi lámpákon csak két szín, a piros és zöld volt jelen (Onion, Sullivan, & Mullen, 2009). Nálunk Magyarországon 1926-ig kellett várni az első jelzőlámpára, melynek gúnyneve igazán találó, a villanyrendőr szó ragadt rá, amit a lámpa alatt álló, a lámpát rúddal állítgató rendőrről kapott. A korabeli célok a mai napig érvényben maradtak, melyek szerint a forgalomirányító rendszerek feladata a biztonság javítása, a torlódások elkerülése, a jobb kapacitás kihasználás, valamint a környezeti szempontok. Bár a vezérlőrendszerek folyamatos fejlődésen mentek keresztül, az általánosan használt fogalomirányító rendszerek nem képesek reagálni a forgalom pillanatnyi dinamizmusára. A tradicionális rendszerek két nagy csoportból állnak, mely az úgynevezett „pre-timed control”, vagyis a fix időtartamú megoldás, valamint az „actuated control”, ami szabad fordításban aktiválás alapú forgalomirányítást takar (Mousavi, Schukat, & Howley, 2017). Az ilyen rendszerek különböző szcenáriókra tartalmaznak előre rögzített hosszúságú lámpafázisokat, melyeket múltbeli adathalmazok alapján alakítanak ki. Az aktiválás alapú forgalom ezen felül számításba vesz egyéb detektor információkat, például hurokdetektor vagy gyalogátkelő jelzőt a lámpafázis módosításához. Dolgozatom során lámpafázis alatt mindig az éppen adott jelzési képet (piros, sárga és zöld kombinációi sávonként), lámpaciklus alatt pedig az összes jelzési kép egyszeri lefutását értem.

1.2 Modern technológiai megoldások bevezetése

Korunk informatikai fejlődése a tárolás és számítási kapacitás terén lehetővé teszi, hogy a technológiai újítások által nyújtott előnyöket egyre szélesebb körben kihasználhassuk. Mivel a városok egyre bővülő detektor infrastruktúrával rendelkeznek, így az ebből generálódó hatalmas adathalmaz megfelelő alapot nyújt a gépi tanuláson (ML-machine learning) alapuló algoritmusok betanításához. A legaktívabban tanulmányozott terület melyet forgalomirányítás kapcsán vizsgálnak a reinforcement learning (RL), vagyis a megerősítéses tanulás. Ezen belül pedig a Q-learningnek és Deep Q-learningnek (DQN) van az egyik legnagyobb irodalma a

témában. Az utóbbi évek kutatásai alapján bebizonyosodott, hogy nyers adatok alapján ezen módszerek különösen hatékonyan képesek megoldani komplex problémákat is. A megerősítéses tanulás fő gondolatmenete a környezettel való interakció, mely során az ügynök -vagyis maga a tanuló algoritmus- képes akciókkal beavatkozni a szimulációba, és megvizsgálni ezeknek a beavatkozásoknak a jóságát. A legtöbb tanulmányban egyetlen RL módszert vizsgálnak meg egy választott jutalmazási metodika mellett. Ezzel ellentétben az alábbi dolgozatban több DQN algoritmus típust is összehasonlítok, nevezetesen a Double DQN-t, Dueling DQN-t és a Double Dueling DQN-t (D3QN). A tudományos munkák nagy része egyetlen csomópont RL alapú vezérlését vizsgálja, ahol egyetlen beavatkozó ügynök vezérli a közlekedési lámpák fázisait. Munkámban egy több csomópontból álló úthálózatot is vizsgáltam, ahol a csomópontokat vezérlő ügynökök egymással információt megosztva a globális optimum elérésére törekednek. Ennek érdekében úgy módosítottam az ügynököket vezérlő algoritmusokat, hogy azok kezelni tudják a szomszédos csomópontoktól kapott adatokat, és annak ismeretében határozzák meg Ezen felül a jutalmazási függvények szélesebb köre is a vizsgálat alapját képezi, elemzések és saját intuíciók alapján nyolc különböző módon tanított modell jóságát is megvizsgálom. A betanított modellek működésének kiértékeléséhez a várakozási időt, átlagsebességet, helyben álló járművek számát és a CO_2 kibocsátás mértékét vettem alapul. Hipotézisem szerint a vizsgálatba bevont modellek bizonyos része képes felülmúlni a jelenleg használatban levő fix időtartamos modellek hatékonyságát az előzőekben felsorolt mutatószámok alapján.

2 Irodalmi áttekintés

Gartner, Stamatiadis, és Tarnoff (1994) tanulmányukban a forgalomirányító rendszerek öt csoportját különítik el a döntési intelligencia mértéke alapján.

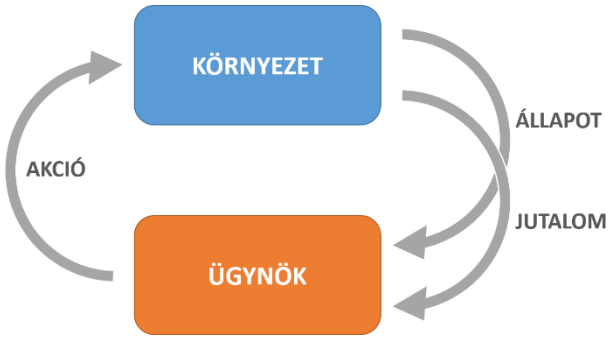
- Első csoportba a döntéshozó mechanizmusok azon sora tartozik, melyek előre meghatározott terveket tartalmaznak múltbeli adatok alapján. Ezek a rendszerek leginkább a napszak alapján váltják a beállításokat.
- A második fejlettségi szintű mechanizmusok már valós időben képesek megváltoztatni az előre meghatározott fázisidőterveket.
- A harmadik csoportba olyan, az előzővel egyező rendszerek szerepelnek, melyek a fázisidőterveket rövidebb idő alatt képesek megváltoztatni.
- Negyedik típusnak azokat a rendszereket hívja, melyek több alrendszert is integrálnak.
- Az ötödik, egyben legfejlettebb csoport pedig azon irányító mechanizmusoknak a köre, melyek valamilyen mértékű öntanuláson alapulnak, és képesek igazodni a pillanatnyi forgalmi dinamikához.

A dolgozatban összehasonlításra kerülő összes gépi tanulás alapú modell az ötödik fejlettségi csoportba tartozik, megfelel a Gartner által leírt kritériumoknak.
Érdemes lehet röviden áttekinteni a klasszikus forgalomirányító rendszerek néhány fejlesztett típusát. A leggyakrabban használt módszer a Koonce és munkatársai által kifejlesztett Webster method (Koonce, és mtsai., 2008). Működésének alapja az összesített utazási idő minimalizálás a Websteri Egyenlet alapján kalkulált ciklus idő és fázis váltások befolyásolásával. Az egyenlet sokféle információt vesz alapul, köztük a gyorsulás, lassulás mértéket, a fázisok számát, a forgalom változását. Egy másik gyakran használt módszer a GreenWave, melyet Roess, Prassas és McShane fejlesztett ki, és a járművek szükséges megállását minimalizálja a kereszteződések közti időbeli eltolódás kiszámításával (Roess, Prassas, & McShane, 2004). A fejlettebb módszerek közé lehet sorolni a Cools, Gershenson és D'Hooghe által létrehozott Self-Organizing Traffic Light Control-t, röviden SOTL rendszert (Cools, Gershenson, & DâzHooghe., 2013). A módszer kulcseleme egy úgynevezett „kérés" a lámpafázis módosítására. Ez a kérés két féle lehet, vagy a pillanatnyi fázis hosszabbítására és továbbléptetésére irányulhat. Hosszabbítási kérés akkor valósul meg, mikor a fázisban eltöltött idő nem éri el a minimálisan meghatározottat, továbblépési igény pedig mikor a várakozó járművek száma elér egy meghatározott határértéket. A gépi tanulási módszerekkel is gyakran

keresztezett megoldás a Varaiya által leírt Max-pressure módszer (Varaiya, 2013). Logikája, hogy a forgalmi csomópontra helyeződő nyomást csökkenti, ahol a nyomást a beérkező útvonalakon tartózkodó járművek száma mínusz a kimenő útvonalakon haladó járművek száma határozza meg.

2.1 Megerősítéses tanulás

A gépi tanulási megoldások beintegrálása a forgalomirányító rendszerekbe évek óta a kutatók fókuszában van, leginkább pedig a megerősítéses tanulásra helyezve a hangsúlyt. A megerősítéses tanulás a gépi tanulás három ágának a tagja a supervised és unsupervised csoportok mellett. A tanuló algoritmust, mely általában és a dolgozat esetében is egy neurális hálózat, ügynöknek nevezzük. Az ügynök tanulási problémáját Markovi Döntési folyamatként (MDP) szokás kezelni, melynek három alapvető eleme a környezet, az észlelés és a cél. Vizuálisan az 1. ábrán szemléltetem a folyamatot.



1. ábra: Megerősítéses tanulás körforgása

Az ügynök számára létrehozunk egy környezetet, mely megismerésével tanulni tud. Esetünkben a környezet egy vagy több forgalmi csomópontból álló úthálózatként képzelhető el, melyről különböző detektorok szolgáltatnak információt. Ilyen detektorok lehetnek hurokdetektorok, mágneses szenzorok vagy éppenséggel térfigyelőkamera képek is. Az egy időpillanatban a környezetből származó információk összességét hívjuk állapotnak vagy angolul state-nek (S), az állapotot meghatározó vektor nagyságát pedig állapottérnek. Az ügynök az állapotot figyelembe véve döntést hoz, mely a szimuláció lefolyása során nem csak a jutalmat magát, hanem esetenként a további állapotokat is meghatározhatja. Az összes lehetségesen meghozható döntést hívjuk akciónak (A). Az ügynök számára létrehozunk egy, a környezethez kapcsolódó komplex célkitűzést, mely jutalom vagy büntetés formájában reprezentálódik (R), és ennek kumulált értékét maximalizálja az ügynök. Az ügynök számára nem határozzuk meg milyen akciót hozzon, hanem fel kell fedeznie melyik akció eredményezi

a legnagyobb jutalmat. Ezt az elsajátítandó stratégiát hívjuk policynak, szabad fordításban irányelvnek (π). Value-nak vagy értéknek (V) a várható diszkontált hosszútávú jutalmat nevezzük. $V_{\pi(s)}$ így az *s* állapot várható hosszútávú jutalma π policyt követve. A Q-érték, vagy más néven akció-érték páros (*Q*) hasonló az előzőekben tárgyalt *V* értékhez, azzal a kiegészítéssel, hogy *a* akció is tartozik hozzá. Így $Q_{\pi(s,a)}$ alatt a π policy által *s* állapotban meghozott *a* döntés várt hosszútávú jutalmát értjük. Az egyik fontos aspektus, ami elkülöníti az RL megoldást a többi ML technikától, hogy egyensúlyt kell találnia az újfajta tudás felderítése és az elsajátított tudás felhasználása között. Ezt a dilemmát nevezik a szakcikkek exploration and exploitation trade-off-nak (Sutton & Barto, 2018). A megfelelő egyensúly érdekében a megerősítéses tanulás során az úgynevezett Epsilon-Greedy algoritmust használják, melynél a két meghozható döntés az ügynök új információ felfedezésére való irányítása random választás segítségével, vagy a meglevő tudás hasznosítása, a policy szerint legmagasabb értékű akció választásával. A döntés meghozásához az algoritmus az $0 \leq \varepsilon \leq 1$ közti határértéket használja, melynek opcionálisan választható kezdőértéke és csökkenési mértéke van. Kezdetben egy magas, 1 közeli határértékről indul az ügynök, aminek minden akció meghozásakor egy szintén [0-1] közti véletlen számmal való relációját veszi. Ha a random szám kisebb mint az ε érték, akkor random dönt az ügynök, ha magasabb, policy szerint. Minden ilyen lépés során előre meghatározott mértékben csökken az ε értéke, így egyre kisebb valószínűséggel lesz alacsonyabb a véletlen szám, míg végül eléri a szintén opcionálisan meghatározható minimális ε értéket.

2.2 Q-learning

A Q-learning (Watkins, Learning from Delayed Rewards, 1989) egyike a leggyakrabban használt RL algoritmusoknak, köszönhetően az egyszerűségének és hatásosságának. Hasonló a működési elve mint Sutton Temporal Differences (TD) (Sutton R. , 1988), vagyis időbeli különbségeken alapuló modelljének. A Markovi Döntési probléma optimumát úgy igyekszik megtalálni, hogy az adott állapot esetén próbálkozik a lehetséges akciókkal, majd kiértékeli annak hatékonyságát a kapott jutalom vagy büntetés, valamint annak az állapotnak az értéke alapján, amibe juttatta. Ha elegendő hosszúságú iterációszám áll rendelkezésére minden állapot minden akciójának feltérképezésére, képes a megfelelő döntéseket megtalálni a hosszútávú jutalom figyelembevételével. Fontos kitétel, hogy a környezetnek diszkrétnek és végesnek kell lennie -ez a továbbiakban még fontos szerephez jut-, valamint az akcióknak is véges számúnak kell lennie. Az egyszerűség kedvéért képzeljünk el egy táblázatot, ami az állapotok és akciók összes kombinációjaként áll elő. Ezt a táblázatot nevezzük Q-táblának. Ezekhez a

kombinációkhoz rendel az ügynök egy úgynevezett Q-értéket, vagy más néven akció-állapot értéket. A Q-érték kalkulálásának módja a Q-függvény segítségével történik:

$$Q^{new}(s_t, a_t) = Q(s_t, a_t) + \alpha \cdot (R_t + \gamma \cdot maxQ(s_{t+1}, a) - Q(s_t, a_t) \quad (1)$$

ahol $Q(s_t, a_t)$ a *t* időpillanathoz tartozó Q-érték, α a tanulás mértéke vagy angolul learning rate (*lr*), R_t a kapott jutalom vagy büntetés és γ a diszkont faktor. A learning rate meghatározza, hogy az újonnan érkező információ milyen mértékben írja felül a már korábbi értéket. Az *lr* 0 és 1 közti skálán változtatható, ahol 0 esetén egy tanulás nélküli modellről beszélhetünk, 1 érték esetén pedig a prior tudást teljes mértékben figyelmen kívül hagyó modellről. A diszkont faktor a jövőbeli jutalom fontosságát határozza meg. 0 esetén teljes mértékben a pillanatnyi jutalom határozza meg az ügynök működését, miopikusnak vagy rövidlátónak nevezzük, míg 1 érték esetén a jövőbeli és a pillanatnyi jutalom azonos fontosságúnak tekintett. A szélsőséges értékek választása kerülendő, mivel terminális állapot híján az értékek konvergenciája sérülhet. A Q-érték tehát a várt és diszkontált jutalom mértékét határozza meg, egy adott állapotban hozott adott döntésnél. A Q-learning célja ennek megfelelően ezen Q-értékek becslése az optimális irányelv vagy policy megtalálása érdekében. A tanulási folyamat különálló szakaszokból, más néven epizódokból áll, ahol *n* epizód sorrendje a következő:

1. adott időpillanati x_n állapot feltérképezése
2. a_n akció kiválasztása és végrehajtása
3. következő y_n állapot feltérképezése
4. azonnali r_n visszajelzés
5. Q_{n-1}-értékek módosítása α_n learning rate használatával, a következő képletnek megfelelően (Watkins & Dayan, Technical Note: Q-Learning, 1992):

$$Q_n(x,a) = \begin{cases} (1-\alpha_n)Q_{n-1}(x,a) + \alpha_n\left[r_n + \gamma V_{n-1}(y_n)\right] & \text{ha } x = x_n \text{ és } a = a_n, \\ Q_{n-1}(x,a) & \text{máskülönben} \end{cases}, \quad (2)$$

ahol

$$V_{n-1}(y) = \max_b \left\{Q_{n-1}(y,b)\right\} \quad (3)$$

A Q-learning alkalmazása forgalomirányítási feladatokra széleskörűen vizsgált terület, több tanulmány is született a témában, köztük (Abdoos, Mozayani, & Bazzan, 2011), (Araghi, Khosravi, Johnstone, & Creighton, 2013), (Arel, Liu, Urbanik, & Kohls, 2010), (Zhao, Dai, & Zhang, 2012).

2.3 Deep Q-learning

Mivel a tradicionális Q-learning módszernek könnyen belátható korlátai vannak a Q-tábla tárolásának és módosításának okán, így a környezet állapota és az akciók száma csak alacsony komplexitású lehet. A probléma elkerülése érdekében kidolgozták a gépi tanulással való bővítését, mely esetben a Q-tábla helyét a Q-értékek becsléséért felelős neurális hálózat veszi át. 2013-ban publikálta a Google DeepMind csapata eredményeit, melyben Atari játékokon futtaták a Deep Q-learning-nek (DQN) nevezett megoldásukat (Mnih, és mtsai., 2013). A DQN modell esetében a környezet és ügynök körforgása megegyezik a tradicionális változatéval, a Q-értékek frissítéséért felelős Q-függvény pedig a következőképp néz ki:

$$Q(s_t, a_t) = R_t + \gamma \cdot maxQ(s_{t+1}, a') \quad , \tag{4}$$

ahol $Q(s_t, a_t)$ a neurális háló predikciója a környezet állapota alapján, $Q(s_{t+1}, a')$ ugyanennek a neurális hálónak a következő állapotra vonatkozó becslése, valamint gamma a korábbiakban részletezett diszkont faktor. A neurális háló tanítása során a célértékek a frissített Q-értékek, a bemenetek a környezet állapotának értékei, míg a kimeneti neuron szám a lehetségesen meghozható döntési akciók száma. A tanítás stabilizálásának érdekében egy úgynevezett Replay Buffert (Lin, 1992) hozunk létre, mely az ügynök memóriájaként funkcionál. Minden lépésben ide mentjük az adott időpillanati állapot értékeket, akciót, jutalmat és az állapotot amelybe léptünk. A Replay Buffer mérete előre deklarált, a legrégebbi értékek törlődnek belőle, ha elérte maximális értékét. A szimuláció futása során adott lépésközönként történik a hálózat tanítása a Replay Bufferből random mintavételezéssel létrehozott minibatchek segítségével.

2.4 Double Deep Q-learning

Ha újra megnézzük a Q-függvényt, láthatjuk, hogy az új Q-érték a jutalom és a következő állapot Q-értékének összegeként áll elő. Utóbbit szintén az a neurális háló becsüli, amelyik a szimuláció során folytonos módon tanul, így a tanítás célértékei a háló frissítésével együtt folyamatosan változnak. Az instabil célfüggvény pedig képes a tanulás konvergenciáját csökkenteni, lassítani. A gyakorlatban tehát elképzelhető, hogy az algoritmus gyakran rendel

egy nem optimális akcióhoz magasabb Q-értéket, így nehezebb lesz felfedeznie az optimális döntést, vagyis lokális optimumban ragad. A probléma megoldására Haaselt, Guez és Silver kidolgozta a Double DQN modellt (van Hasselt, Guez, & Silver, 2016), melyben a célértékek időlegesen rögzítésre kerülnek. Ezt egy második neurális hálóval éri el, mely az elsődlegesnek pontos másolata. A másodlagos háló tanítása a szimuláció teljes időtartama alatt fagyasztva van, a súlyok pedig adott időközönként szinkronizálásra kerülnek az elsődleges hálóéval. A Q-értékek frissítésének képlete a következőképp módosul:

$$Q(s_t, a_t) = R_t + \lambda \cdot Q^c(s_{t+1}, \arg\max Q(s_{t+1}, a')), \tag{5}$$

ahol $Q(s_{t+1}, a')$ az elsődleges háló predikciója a következő állapot alapján. Megkeressük a legmagasabb Q-értékkel rendelkező akció indexét, és a Q^c másodlagos neurális háló által becsült Q-értékek közül az indexnek megfelelően választunk.

2.5 Dueling Deep Q-learning

Előfordulhatnak olyan esetek, mikor egy állapot esetén minden akcióhoz ugyanolyan Q-érték tartozik, vagyis egyik akció választása sem juttat előnyösebb helyzetbe, mint a többi, ugyanolyan eredménnyel járnak. Wang, és munkatársai (2015) által leírt Dueling DQN modellje képes kiszűrni az ilyen állapotokat, melyek esetén nem szükséges betanulni a hozzájuk tartozó akciók értékét sem, ezzel is stabilizálva a tanítást. Különösen azokban a környezetekben releváns, ahol az akció nem mindig befolyásolja azt érdemben. Esetünkben ez kardinális kérdés, hiszen olyan esetekben, mikor nincs áthaladó forgalom, a modellnek el kell sajátítania azt a tudást, hogy nem szükséges a lámpafázist sem váltani. A technikai megoldás részleteinek megértéséhez szükséges a megerősítéses tanulásról szóló szekcióban bevezetett *V* és *Q* érték. *V* egy adott állapot értékét, míg *Q* egy adott állapotban meghozott adott akció értékét jelöli. Wang ezek segítségével egy új értéket vezetett be, melyet Advantage-nek vagy előnynek (A) nevez, értéke pedig meghatározza, hogy mennyire előnyös az akció választása a többihez képest. Kiszámításának módja a következő:

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s) \tag{6}$$

A Dueling DQN lényege a neurális háló szerkezetében rejlik. A háló végén két szeparált szálat hozunk létre, melynek egyike az eddigiekkel egyező módon a Q-értékeket becsüli, a másik szál pedig bemutatott Adventage functiont, mely egy neuronnal rendelkezik. A szeparált

szálakat követően egy aggregációs réteg következik az architektúrában. Kézenfekvő lehetne az értékek összeadása, azonban ekkor elveszne a két érték azonosíthatósága, mely gyenge teljesítményhez vezet. Wang a következő aggregáló képletet határozta meg a probléma orvoslására (Wang, és mtsai., 2015):

$$Q(s,a) = V(s) + A(s,a) - \frac{1}{n}\sum_{a'} A(s,a') \tag{7}$$

2.6 Double Dueling Deep Q-learning

Mivel a Dueling DQN által hozott módosítások csak a neurális háló architektúráját érintették, így a módszer szabadon keverhető a Double DQN-el. A tanítás továbbra is a $Q(s_t, a_t) = R_t + \gamma * maxQ(s_{t+1}, a')$ függvénnyel történik a DQN esetében, és $Q(s_t, a_t) = R_t + \gamma * Q^c(s_{t+1}, argmaxQ(s_{t+1}, a'))$ függvénnyel a Double DQN esetében. A Double Dueling DQN-t a szakirodalmaknak megfelelően D3QN-ként fogom rövidíteni.

2.7 Multi-agent

Könnyen belátható, hogy a valóságban való felhasználás esetén nem lesz elegendő egyetlen modell működtetése, mivel a városok, de még a falvak infrastruktúrája is bonyolultabb egy kereszteződésnél, így tehát elengedhetetlen a multi-agent reinforcement learning (MARL) megoldások vizsgálata és bevezetése. A MARL megoldások több dimenzió alapján is csoportosíthatók. A legmagasabb szintű csoportosítás szerint a módszerek besorolhatók a

- teljes mértékben kooperatív
- teljes mértékben kompetitív
- és a vegyes kategóriákba.

A dolgozatban kutatott forgalomirányító rendszerek a kooperatív MARL rendszerek közé tartoznak, a különálló ügynökök feladata egy közös cél, vagyis az úthálózat forgalmának valamely tulajdonságának maximalizálása vagy minimalizálása. A teljes mértékben kooperatív megoldás feltételezi, hogy a közös cél egyetlen közös jutalmat jelent, nem pedig az ügynökök szeparált jutalmát. A dolgozat szempontjából a legfontosabb csoportosítás az ügynökök tudomásának mértéke a szimulációban résztvevő többi ügynökről. Ez alapján a legalapvetőbb megoldás az információ megosztás nélküli, szeparáltan tanuló ügynökök esete, melyet Independent Q-learningnek (IQL) nevezünk (Tan & Ming, 1993). Kézenfekvő lehet az általam „ügynök kihelyezésnek” elnevezett, szintén információmegosztás nélküli megoldás, ahol az

egy csomóponton betanított modellt helyezzük ki a bővített úthálózat minden kereszteződésébe, együtt történő tanulás nélkül. Ugyan szakirodalmi utalást nem találtam erre vonatkozóan, de mindenképp érdemesnek tartom a vizsgálatát. A gyorsabb és stabilabb tanulás érdekében azonban az ügynökök meg is oszthatnak egymás között információkat, ez lehet a kapott jutalom, a meghozott döntés, az észlelt állapot vagy egy elsajátított policy is. Ez a többletinformáció az ügynökök állapotterének bővítését jelenti. A harmadik, dolgozat szempontjából releváns MARL elkülönítés, hogy centralizált vagy decentralizált rendszerről beszélünk. Centralizált rendszer esetén vagy egyetlen ügynök kezeli az összes kereszteződést, ami hatalmas állapotteret feltételez, vagy a több kihelyezett ügynök felett hierarchikusan áll egy központi irányító. Ezekben az esetekben beszélhetünk az eddigiekben tárgyalt standard Markovi Döntési mechanizmusról. A dolgozatban azonban kizárólag decentralizált, központi irányítást nélkülöző MARL megoldásokat vizsgálok (DeMARL), mely esetben az MDP generalizált formája áll fent, amit sztochasztikus játéknak nevezünk. A sztochasztikus játék és alapvetően a MARL is rendkívül sok nehézséget és megoldandó problémát rejt magában, melyek közül kiemelek párat. A legintuitívabb probléma a számítási és memória igény exponenciális növekedése az ügynökök számának gyarapodásával. Erre az állapottér és a megosztott információ formájának megfelelő kialakításán kívül jelenlegi tudomásom szerint nincs más megoldás. Szintén felmerülő probléma lehet a stacionáriusság hiánya. Míg a single-agent környezetben az ügynök csak a saját akciójának hatásával szembesül, addig a multi-agent környezetben a többi ügynök viselkedése is meghatározó. Ráadásul az ügynökök kiismerését az is nehezíti, hogy a tanulás szimultán módon történik, minden ügynök esetén egyszerre. A probléma több ponton fejti ki hatását. Ahogy említettem, az RL algoritmusok egyik központi eleme az egyensúly megtalálása a felfedezés és meglévő információ felhasználás között, vagyis az exploration–exploitation tradeoff mechanizmus. MARL esetén az ügynököknek nem csak a környezetéről, hanem a többi ügynökről is fel kell fedeznie információt, azonban a túl nagymértékű információ kutatás, ami kiszámíthatatlan működéssel párosul, destabilizálhatja a többi ügynök tanulási dinamikáját, ezáltal a tanuló ügynök magát is nehezebb helyzetbe hozza (Busoniu, Babuska, & De Schutter, 2008). Ezen felül a többi ügynök jelenléte és viselkedése által ismét egy mozgó célfüggvénnyel állunk szemben, hasonlóan a Dueling DQN megoldásnál. A közös jutalom is tovább mélyíti a problémakört, mivel mértéke jobban függ a többi ügynök döntésétől, mint az adott ügynökétől. Amennyiben eltekintünk a közös jutalomtól, és egyénileg kalkuláljuk azt ügynökönként, szembesülnünk kell az „önérdekű” ügynök problémájával. Ez alatt azt értjük, mikor az ügynöknek érdekében áll a többi ügynök kárára cselekedni a saját

jutalmának maximalizálása érdekében, de közben mégsem veheti teljes mértékben semmibe a környezet többi résztvevőjét (Nguyen, Nguyen, & Nahavandi, 2019).

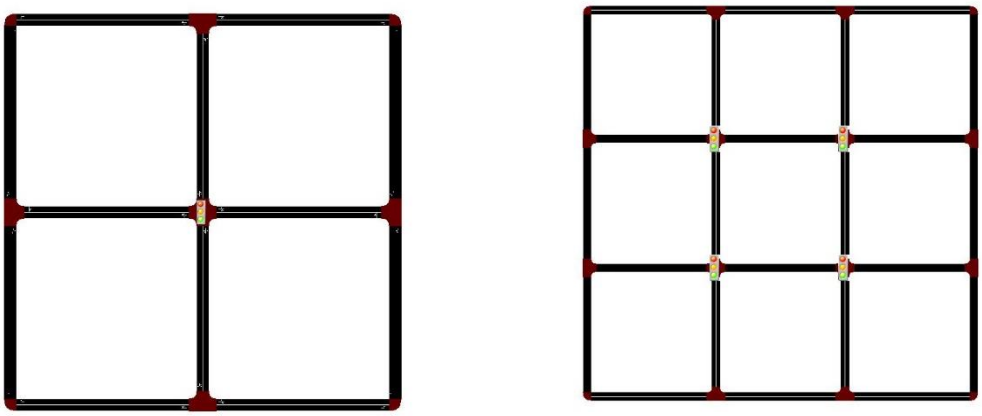
2.8 FingerPrint

A környezet non-stacionáriusságának megoldására Foerster és munkatársai két módszert dolgoztak ki: az importance samplinget, valamint a FingerPrint-et (Foerster, Nardelli, Farquhar, & Afouras, 2018). Előbbi hatásosságát több tanulmány is kétségbe vonja, így a dolgozat a FingerPrint megoldás bevezetésére koncentrálódik. Az ötlet alapja a MARL probléma gyökeréből adódik, vagyis hogy a többi agent policy-jének változása teszi non-stacionáriussá a tanuló ügynök környezetét. Kézenfekvő megoldás lehet az ügynök állapotterének bővítése a többi ügynök policy-jével, így ismét egy stacionárius környezetet kapunk, azonban mivel neurális háló becsüli DQN esetén a policy-t, így a háló teljes súly összetételét kéne megadni, ami méretét tekintve ellehetetleníti a számítást. A feladat tehát egy olyan, alacsony dimenziójú ujjlenyomat, vagyis fingerprint létrehozása, ami információt hordoz a többi ügynök irányelveiről. Foersterék úgy találták, a tanulási iteráció szám és az exploration–exploitation tradeoff aránya elégséges mértékben korrelál a hatékonysággal, így ezzel a két értékkel való állapottér bővítés segíthet az optimális policy megtalálásában.

3 Szimulációs környezet részletei

3.1 Úthálózat

A forgalom szimulálása a SUMO nevű mikroszkopikus közlekedés szimulációs szoftver segítségével történt, mely nyílt forráskódú és kellően részletes beállítási lehetőségeket nyújt. Vizuális GUI felületén valós időben követhető a forgalom lefolyása, valamint a lámpaciklusok változása. A modellek betanításához és teszteléséhez egy és négy csomópontból álló úthálózat készült a SUMO NetEdit szoftverének használatával, a 2. ábrán látható módon.



2. ábra: felhasznált úthálózatok képei

Az egy csomópontos úthálózat a különböző jutalmazási függvények összevetéséhez, valamint a DQN variánsok tanulási dinamikájának és működési hatékonyságának összehasonlítására lett felhasználva, míg a négy csomópontos úthálón a már szűkített kombinációk újra futtatása és a MARL modellek betanítása történt. A szimuláció futása során a szükséges adatok lekérdezésére a SUMO TraCI (Trafic Control Interface) felülete állt rendelkezésre, mely segítségével Python alapon lehet adatokat lekérni, valamint forgalmat módosítani. A lekérdezés elkészítésénél fontosnak tartottam, hogy az OpenAI Gym környezeti standardeknek megfeleljen. Annak előre elvárt parancsai lehetővé teszik a környezet generalizált használatát egyéb eszközökkel is. Minden útszakasz két sávból áll a két útirány biztosítására, mindegyiken 50 km/h sebesség korlátozással. A kereszteződésben a járművek minden irányba kanyarodhatnak, így egymás útvonalát is keresztezhetik. A forgalom detektálására az adott lámpától való 50 méteres táv került kijelölésre, mind a négy bejövő forgalmat generáló sávon, amelyről származó adatok úgy lettek kialakítva, hogy szimulálja a valós kameraképből kinyerhető információt. Azért a kamerára esett a választás, mivel ennek a detektor típusnak használhatók fel legáltalánosabb módon az információi, kevésbé szituáció

függő, ellentétben a hurokdetektorokkal, melynél a kereszteződéstől való távolság eltérései a neurális háló újratanítását igényelnék. A kialakított mérőszámok a következők:

- a „kamera” által látott járművek száma (0-10 skála, darab)
- a járművek átlagsebessége (0-10 skála, m/s)
- adott időpillanati lámpafázis
- adott lámpafázisban eltöltött idő (másodperc)

Mivel a modell valós időben dolgozza fel a kameraképek adatait, így képes lehet az azonnali beavatkozásra, esetünkben lámpafázis váltásra. Ennek elérése érdekében a gyakran használt lámpaciklus hossznyi szimulációs lépés helyett a valós TraCI lépést, vagyis 1 másodpercet választottam, az ügynök pedig minden másodpercben két féle akció közül választhat:

- meghosszabbítja a pillanatnyi lámpafázist
- vagy megszakítja azt, és a következőbe lép.

Érdemes megjegyezni, hogy lámpafázis váltás esetén automatikusan sárga ciklus kezdődik, mely három másodperces időtartama során nincs mód beavatkozásra.

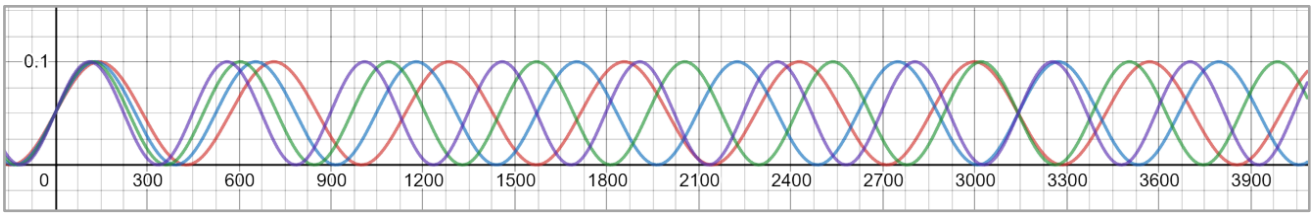
3.2 Forgalomgenerálás

Annak érdekében, hogy az ügynökök minden lehetséges forgalmi szituációt megtapasztalhassanak, és azokon tanulni tudjanak, a forgalom generálására sajátkezű megoldás született. Az *i* darab, úthálózat széléről befelé haladó lehetséges irányokból zajlik a közlekedés a következő szinusz függvény alapú indulási valószínűségekkel:

$$max\_prob \cdot \frac{\left(\text{math.sin}\left(\text{math.radians}\left(\frac{current\_time}{100} + \left(i \cdot \frac{current\_time}{1000}\right)\right)\right) + 1\right)}{2}$$

ahol *max_prob* a függvény értékkészletének maximális értékét határozza meg, így gátolva meg a túl sűrű járműgenerálást, *current_time* a szimuláció pillanatnyi lépésszáma, *i* pedig az aktuális

útirány sorszáma, ezzel egyedi hosszúságúvá téve a függvény periódusát. A négy útirány gépjármű-elhelyezési valószínűségei az alábbi 3. ábrán bemutatott grafikon szerint alakulnak:



3. ábra: Forgalomgenerálás szinusz függvényei

Mivel a tanítás több százezer lépésen keresztül történik, így a fenti szinuszfüggvény-együttállások is ismétlődni fognak, ezzel lehetőséget adva az ügynöknek a forgalom megismerésére és a túltanulásra. Ennek elkerülése érdekében minden lépés esetén a gépjármű elhelyezésének valószínűsége véletlenszerűen kerül kiválasztásra a szinuszfüggvény által meghatározott érték és egy [0-*max_prob*] közti random szám közül.

A gridlock jelenségével elengedhetetlen foglalkozni. Az ilyen szituációk alatt azt értjük, mikor a forgalom olyan mértékben és módon torlódik fel, hogy zavarja a keresztbe haladó sáv forgalmát is, blokkolva a kereszteződést. Mivel az ügynökök folyamatosan kísérleteznek, így ez nem lehetetlen jelenség a szimuláció során sem. A dolgozatba ennek megoldása érdekében két hiperparaméter került bevonásra, a gridlock-nak minősítés ideje másodpercben, amit a leghosszabb ideje várakozó járművel mérünk, valamint a büntetés, amit gidlock esetén az ügynökök egységesen kapnak. Amennyiben gridlock miatt áll be a forgalom, a szimuláció újraindul, melyet 180 másodperc forgalom helyreállítási idő követ. Az ügynökök megkapják a büntetett jutalmi értéket, ezt követően pedig a tisztított úthálózaton folytathatják a tanulást.

3.3 Jutalmazási függvények

A dolgozat egyik központi eleme az eltérő jutalmazási módszerek összevetése, mivel ez az egyik legfontosabb paraméter a tanulás hatékonyságát illetően. Ennek ellenére, vagy éppen emiatt nagyon széleskörű megoldások születtek és nincs konszenzus a „best practice”-t illetően. A jutalom függvények hatást fejtenek ki egyben a tanulási dinamikára, valamint a betanított modell hatékonyságára is. A probléma önmagában azért is nehéz, mert nehéz megfogalmazni mi számít a forgalom szempontjából a legfontosabb tulajdonságnak: legyen minél nagyobb az átlagsebesség, legyen minél alacsonyabb a CO_2 kibocsátás, netán a minél kevesebb megállás a jó forgalmi dinamika mértéke? A kérdés vizsgálata érdekében 8 jutalmazási függvényt vontam be a dolgozatba, melyek a következők:

- Várakozási idő minimalizálás (VIMin): 0,1 m/s-nál lassabban haladás időtartama

(Zhang, Ishikawa, Wang, B., & Tonguz, 2020) alapján:

- Átlagsebesség maximalizálás (ÁSMax)

(Chu, Wang, Codecà, & Li, 2019) alapján:

- Sorban álló járművek számának minimalizálása (SSzMin)

(Hajbabaie & Benekohal, 2013) alapján:

- Utazási idő minimalizálás (UIMin)
- Utazási szám maximalizálás: az útjuk végén a szimulációt elhagyó járművek száma (USzMax)
- Súlyozott utazási szám maximalizálás: az előző, súlyozva a legrövidebb útszakasz hosszával (SUSzMax)
- Throughput minus queue maximalizálás (TMQMax): az útjukat megkezdő (szimulációba belépő) és sorban álló (0,1 m/s-nél lassabban haladó) járművek számának különbsége

Az utolsó három jutalmazási függvény esetén az úthálózat széleire telepített hurokdetektorokat is feltételezünk, mely a valószerűtlenség mellett a Q-learning alapvető mechanikáját is zavarhatja. Minden szimulációs lépésben az ügynök döntést hoz, melynek eredményét a következő lépés alapján jutalomként visszakapja, a jutalom mértéke alapján pedig értékeli a döntése hatásosságát. Esetünkben egy lépés egy másodpercnek felel meg, így az akció és a jutalom, vagyis példaként az útját befejező autó észlelése között - mely egy 50-100 méterre található hurokdetektorból származik- között sok lépés telik el. Ugyan az ügynök a jutalmak egészét igyekszik maximalizálni a Q-tábla segítségével, mégis problémát okozhat a túlzott mértékű késleltetés. Stevens és Yeh (2016) munkája nyomán nem az úthálózatba belépő és elhagyó járművek számát vettem a számítások alapjául, hanem a kereszteződésre szűkítettem a vizsgált területet (Stevens & Yeh, 2016). Ezáltal a jutalmat az ügynök jelentősen hamarabb kapja, mely feltevésem szerint javítja a modell hatékonyságát, valamint a hurokdetektorok is elhagyhatók, elegendő az ellenirányú kamerakép vizsgálata, melynek minden esetben adottnak kell lennie, az ügynök működésének alapfeltétele. Az említett hipotézis vizsgálata érdekében a hurokdetektorokkal operáló TMQ jutalmazás továbbra is az elemzés része marad, a módosított változata pedig

- reversed TMQ (rTMQMax) néven szerepel az elemzésben.

A jutalom tárgyán felül a jutalom számításának módja is hatással van a tanulásra. Ahogy a multi-agent szekcióban említettem, a kooperatív MARL rendszerek feltételezik a közös, megosztott jutalmat. A hatékonyságának eldöntésére a globális és lokális jutalmazási rendszert is az elemzésbe vontam. Mivel a négy csomópontos úthálózat kicsinek számít, így nincs szükség a globális jutalom módosítására, de nagyobb úthálózatok esetén lehet kísérletezni a globális tér méretével, vagyis hogy hány szomszédnyi távolságra levő csomópontok jutalmait összegezzük.

3.4 Elemzésbe bevont modellek

Az összehasonlított modellek listája a következő: DQN, Double DQN, Dueling DQN, D3QN, ügynök kihelyezés, IQ, egyszerűsített információmegosztású (IM) DeMARL mely a szomszédos kereszteződések lámpafázisára, az abban eltöltött időre, valamint detektorok által látott járművek szummázott átlagsebességére és számára terjed ki, valamint a teljeskörű információmegosztású DeMARL mely a szomszédos kereszteződések lámpafázisára, az abban eltöltött időre, valamint sávonként elkülönítve a detektorok által látott járművek átlagsebességére és számára terjed ki.

3.5 Paraméterbeállítások

Mivel rendkívül sok hyperparaméterrel rendelkezik a modell, így elengedhetetlennek tartottam valamilyen szintű hyperparaméter optimalizáció használatát. Mivel esetemben a modellek összehasonlítása öt változó alapján történik, így nincs egyetlen mérőszám, mely alapján a paraméter kombinációk jóságát mérni lehetne, a jutalom sokszor tévútra visz. Ebből kifolyólag a piacon levő megoldásuk, például az Optuna sajnos nem alkalmazhatók. A hyperparaméter optimalizációnak öt válfaja létezik, a manuális keresés, random keresés, rács keresés, automatizált hangolás és ANN alapú hangolás. Mivel a MARL rendszer tanítása komplex, és rendkívül sok CPU óra egy modell betanítása, így a random search optimalizációra esett a választásom. Az optimalizáció során a program az összes lehetséges hyperparaméter kombinációt tartalmazó táblából választ véletlen módon, majd betanítja és teszteli azt. Összesen 100 kombináció, 10000 lépésen történő tanítása került tesztelésre, majd az öt mérőszám szerinti bontásban gyakoriságok segítségével kerültek meghatározásra a legjobb illetve legrosszabb eredményeket elérő modellek paraméterei. A dolgozat során használt paraméterek és paraméter értékek listája a következő:

1.táblázat Alkalmazott paraméterértékek

Paraméter	Érték	Jelentés
eps_start	1	epsilon greedy policy kezdőértéke
eps_decay	0,9999	epsilon greedy policy diszkont faktora
eps_end	0,02	legkisebb epsilon értéke
gamma	0,9	Q-érték frissítés diszkont faktora
lr	0,0005	learning rate, neurális háló tanulási sebességének mértéke
batch_size	32	tanulás során használt egységek nagysága
target_update	500	mekkora lépésközzel mentsük át a neurális háló súlyait a target modellbe
learn_freq	1	mekkora lépésközzel végezzünk tanítást (mini batchel)
learn_start	1000	mennyi lépés után legyen az első tanítás
memory_size	50000	replay buffer mérete
ep_length	100000	mennyi TraCI lépésen keresztül tanuljon a modell
first_layer	32	első rejtett réteg neuron száma
second_layer	128	második rejtett réteg neuron száma
third_layer	128	harmadik rejtett réteg neuron száma
gridlock_time	100	hány mp után sorolja be gridlocknak
gridlock_rew	-100	jutalom értéke gridlock esetén

A neurális hálóban az output réteg kivételével minden esetben Rectified Linear Unit (ReLU) aktivációs függvényt, valamint He uniform (He, Zhang, Ren, & Sun, 2015) súly inicializálást alkalmaztam.

3.6 Tesztelési folyamat

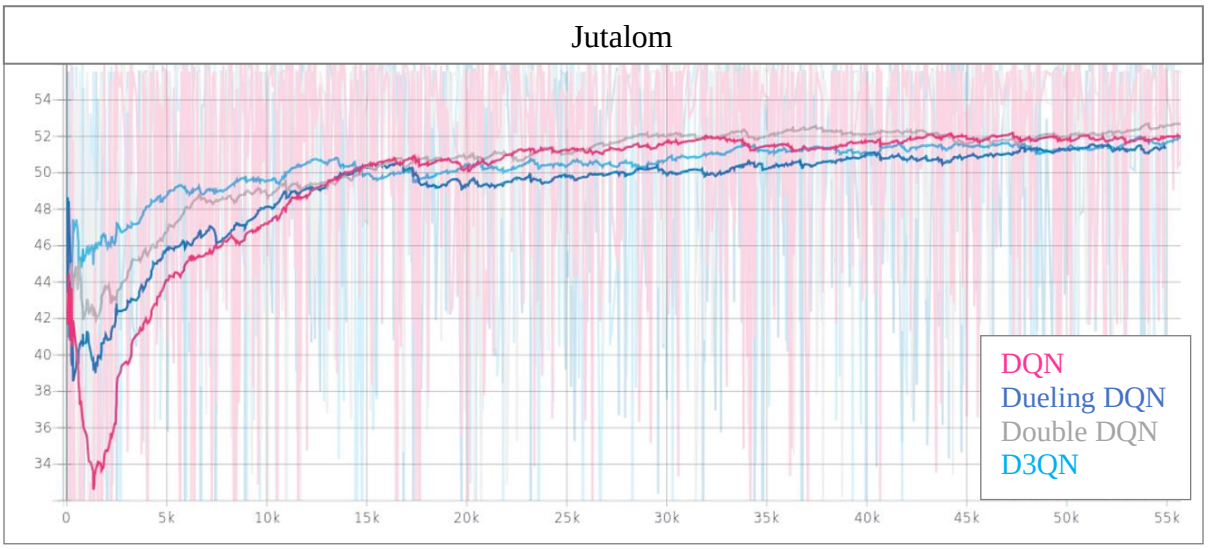
Annak érdekében, hogy a modelleket ne csak egymással lehessen összehasonlítani, egy Baseline modell is elkészítésre került, melynek lámpaciklusai a vizsgált úthálózatok esetén megegyeznek. Minden közlekedési lámpaciklus négy fázisból áll: 12 másodperces z/p/z/p, 3 másodperces s/p/s/p, 12 másodperces p/z/p/z és 3 másodperces p/s/p/s ahol *„p”* a piros, *„s”* a sárga és *„z”* a zöld jele. A fázishosszok a szimulált útvonal sebességhatárát figyelembe véve lettek kialakítva, a való életből vett általános javaslatnak megfelelően. A tesztelési folyamat során a forgalomgenerálás a betanítás során használt szinusz függvény alaptól teljes egészében eltér, a túltanulás torzításának elkerülése érdekében. Három forgalmi dinamikából áll össze a tesztelés: alacsony forgalom [0-0.05] közti jármű indítási valószínűséggel, közepes forgalom [0-0.15] közti valószínűséggel és sűrű forgalom [0-0.25] közti valószínűséggel. Minden forgalmi dinamika 5 blokkból áll, ahol egy blokk 300 lépés hosszúságú, és minden blokkban az adott forgalmi dinamikának megfelelő random jármű indítási valószínűség kerül meghatározásra külön minden befelé induló sávra. Ezek a valószínűségek a blokkon belül változatlanok maradnak. A teljes folyamat tehát 3x5 blokkból áll, így 4500 másodpercet vagyis

75 percet jelent. A randomizálás magja a tesztelés során minden esetben rögzítésre került az eltérő modellek összehasonlíthatósága érdekében. Az összehasonlításhoz a helyben álló járművek számát, a várakozási időt, az átlagsebességet, a CO_2 kibocsátást, valamint a megszerzett jutalmat vettem alapul, melyeket a TensorBoard vizualizációs felület segítségével ábrázoltam és vizsgáltam. Az egy csomóponton tanított modellek tesztelése egyszerűsített folyamattal történt, mely egy 1000 lépéses, SUMO randomTrips python scriptjével készült forgalmat jelent. Annak érdekében, hogy a túl sok lámpafázis váltással operáló ügynökök könnyen kiszűrhetők legyenek, a sárga ciklus egy lépésként lett feltüntetve, a vizsgált értékek pedig szummázódnak. Ennek segítségével a diagram hosszából következtetni lehet a sárga lámpafázisok számára.

4 Teszteredmények

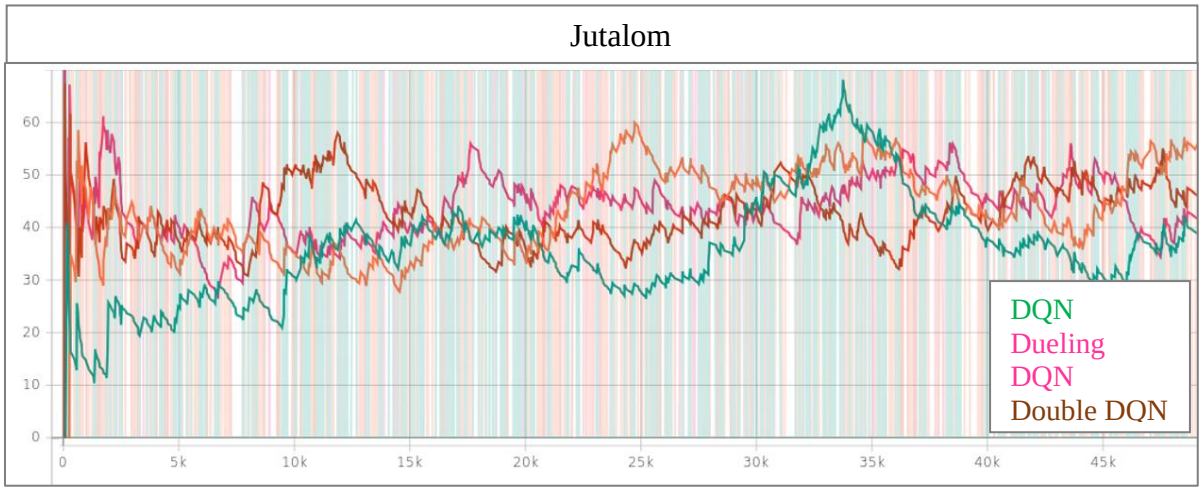
4.1 DQN variánsok

A dolgozat teszteredményeinek első felében a különböző DQN variánsok hatékonyságát vizsgálom a tanulási dinamika, illetve a választott mérőszámok alapján. A tanítás a legegyszerűbb egy csomópontos úthálózaton történt, mivel az alapvető dinamikai eltérések ebben az esetben is megmutatkoznak, továbbá elkerülhető a sokkal nagyobb számítási kapacitást igénylő összetett úthálózaton való újratanítás. Az alábbi 4. ábrán az átlagsebesség maximalizálással tanított különböző Q-learning alapú ügynökök teljesítményei láthatók, x tengelyen a lépésszámmal, y tengelyen a jutalommal (m/s).



4. ábra: Átlagsebesség maximalizálás – jutalom mértéke

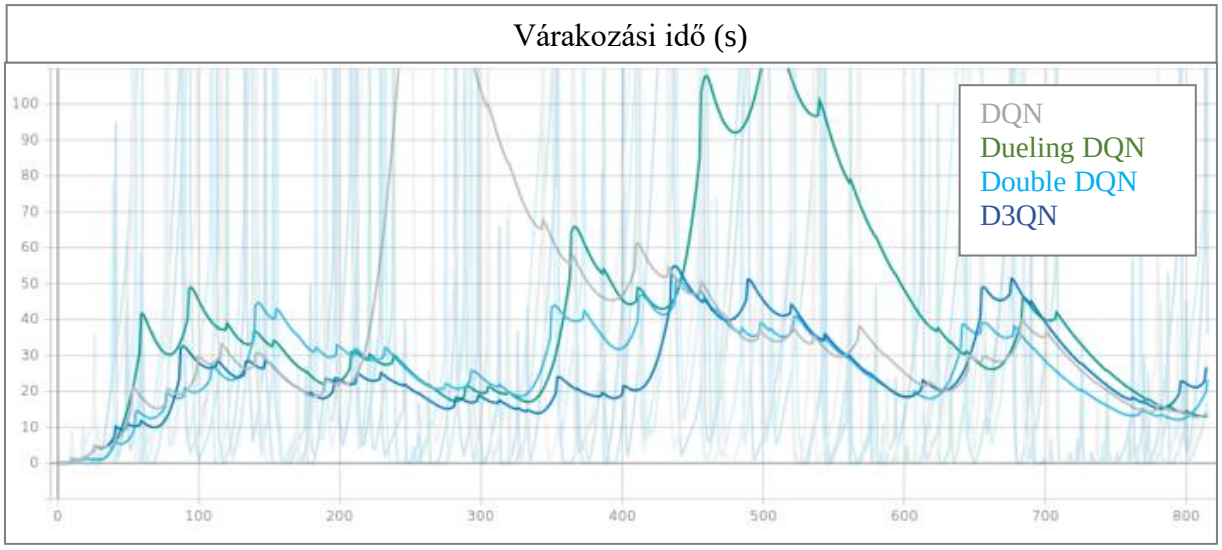
Látható, hogy a különböző modellek közel azonos jutalmazási szintre tanultak be a futás időtartama alatt, azonban a tanulás dinamikájában eltérések tapasztalhatók. Leglassabban egyértelműen a szimpla DQN modell tanult be, legyorsabb ütemben pedig a Double Dueling DQN. Alábbiakban a súlyozott utazási szám maximalizálása alapján tanított modellek jutalom értékei vannak feltüntetve.



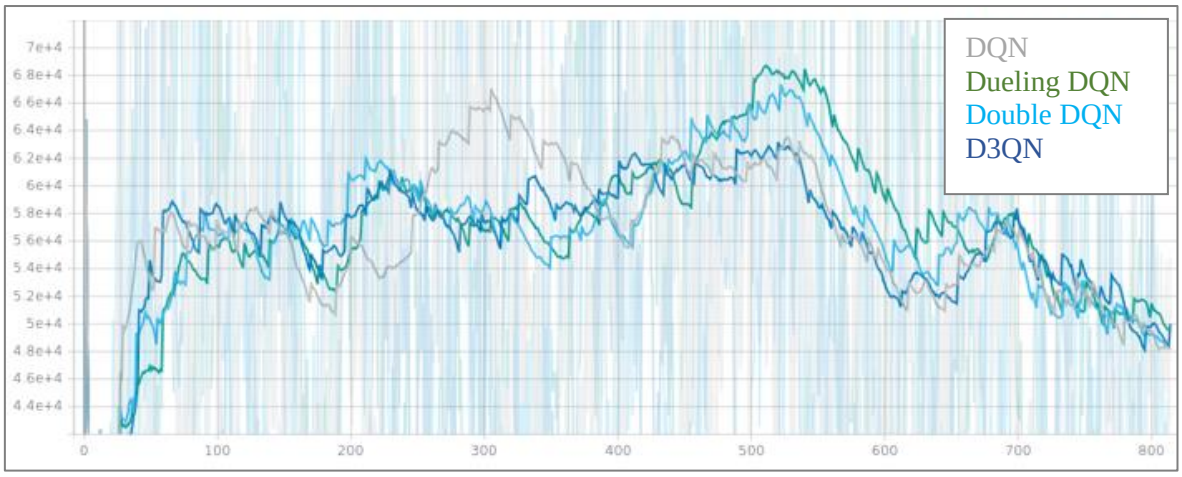
5. ábra: Jutalom értékek utazási szám maximalizálás esetén

Ebben az esetben már nehezebb egyértelmű különbségeket találni, mely kiváltó oka többek között az is lehet, hogy az átlagsebesség alakítására az ügynöknek jóval nagyobb ráhatása van, mint a szimulációt elhagyó járművek esetében, ezesetben a forgalom generálásnak kiemelt szerepe van.

Ugyan a különböző DQN architektúrák előnyei leginkább a jutalom maximalizálásának tanulási dinamikájában érzékelhetők, más módon is javíthatják az ügynök működését. Ennek ellenőrzése a teszteléshez használt mutatószámok segítségével történt. Az alábbi két grafikonon a súlyozott utazási szám maximalizálással jutalmazott modellek tesztfolyamatán keresztül figyelhető meg a CO_2 kibocsátás, valamint a várakozási idő alakulása. Az x tengelyen a lépésszám, az y tengelyen az első ábrán a kibocsátás található mg/s-ban, a második ábrán másodperc szerepel.



6. ábra: Várakozási idő mértéke súlyozott utazási szám maximalizálás esetén



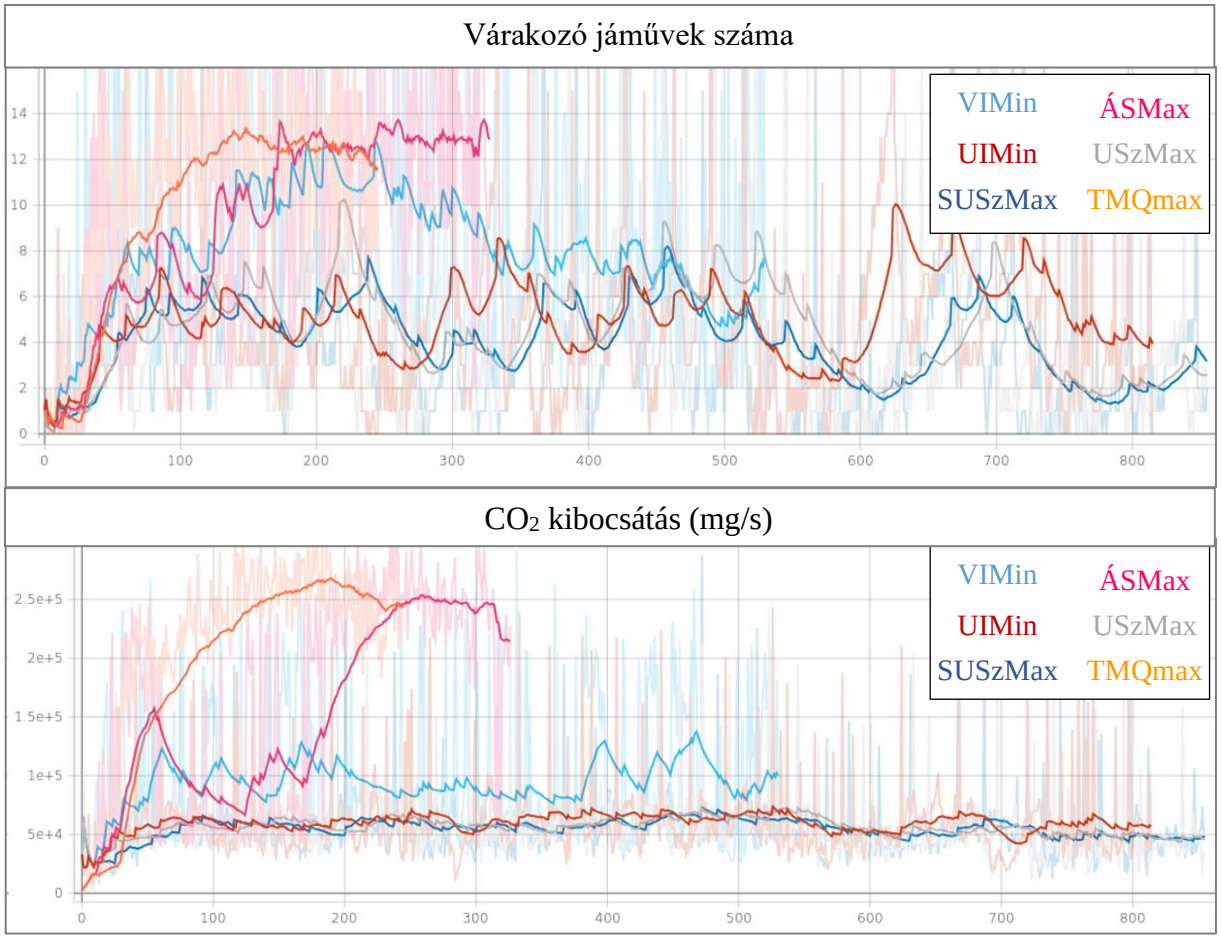
7. ábra: CO_2 kibocsátás mértéke súlyozott utazási szám maximalizálás esetén

A részletes teszt során a DQN és Dueling DQN modell variáns nyújtotta a legkevésbé stabil eredményeket, többszöri és nagymértékű kilengés látható a trajektórián. A Double DQN és a D3QN minden vizsgálatba vont paraméter esetén stabil és alacsony értékeket produkált, így a tanulási dinamikában mutatott fölényt is figyelembe véve a további összehasonlítások alapjául a D3QN modell fog szolgálni. Az alábbi táblázatban numerikusan is megtekinthető a különböző mérőszámok átlagos értékei modellenkénti bontásban.

2.táblázat Modellek által elért eredmények

	CO2 kibocsátás (mg/s)	**Helyben álló járművek száma**	**Várakozási idő (s)**
DQN	56252	4.126	39.97
Dueling DQN	56846	4.222	40.31
Double DQN	56376	3.926	27.75
D3QN	**56181**	**3.645**	**26.33**

4.2 Jutalmazási függvények egy csomópontos úthálózaton



8. ábra: Várakozó járművek száma (felül) és CO_2 kibocsátás (alul) különböző jutalmazási függvények esetén

3.táblázat Jutalmazási függvények tesztelési mérőszámainak átlaga

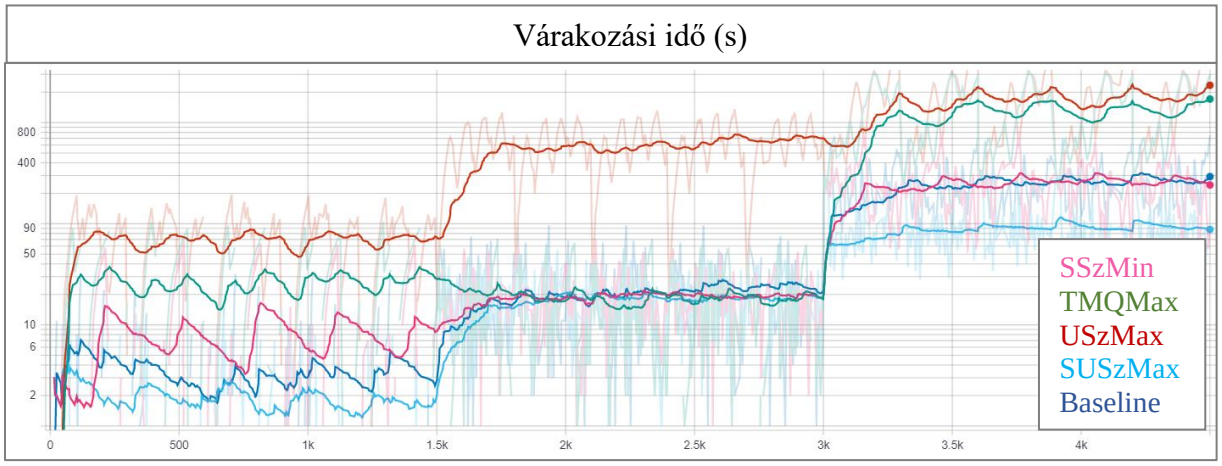
Átlag értékek			
	CO_2 kibocsátás (mg/s)	***Helyben álló járművek száma***	***Várakozási idő (s)***
ÁSMax	162900	10.23	34.52
TMQMax	213633	10.51	**14.81**
USzMax	56449	4.548	41.90
SUSzMax	**56181**	**3.645**	26.33
UIMin	60229	5.127	47.77
VIMin	95474	8.264	44.49

Továbbiakban az egy csomóponttal rendelkező úthálózaton, különböző jutalmazásokkal tanított ügynökök hatékonysága kerül bemutatásra. Egyből szembetűnő jellegzetesség a mindkét grafikonon rövid lépésszámmal jelenlevő TMQ maximalizálás és átlagsebesség

maximalizálás alapú modellek. Ezt a túl sok lámpafázis váltás okozza, mely a SUMO vizuális GUI felületét vizsgálva is szembetűnő volt. Kisebb mértékben ugyan, de a várakozási idő minimalizáláson alapuló ügynöknek is hasonló gyengesége mutatkozik meg, mind a helyben álló járművek számát, mind a CO_2 kibocsátás tekintetében. A hibának lehetséges oka lehet, hogy a folyamatos fázisváltás a hozzá tartozó sávokon araszolásra kényszeríti a járműveket, mely így nem számít várakozásnak. A súlyozott utazási szám maximalizálás alapú ügynök működése hatékonyabbnak tűnt súlyozás nélküli párjánál, és a teljes teszt alatt a legstabilabb és legalacsonyabb értékeket produkálta, ahogy az a táblázatban is látszik. A CO_2 kibocsátás mértékét tekintve a teljes tesztfázis egésze alatt a legalacsonyabb értékeket érte el. Utazási idő minimalizálás során az értékek alacsonyak, ugyanakkor időközönként nagymértékben megnőtt a helyben álló járművek száma. A fenti táblázat is alátámasztja a grafikon alapján tett következtetéseket, miszerint a súlyozott utazási szám maximalizálás érte el a legoptimálisabb értékeket. A TMQ maximalizáláshoz tartozó különösen alacsony átlagos várakozási idő jó példája, hogy miért nem lehet pusztán egy mérőszámra hagyatkozni az összehasonlítások során. Minden bizonnyal talált valami olyan anomáliát, mely nem számít sikeres forgalom allokálási módnak, az átlagos várakozási időt mégis alacsonyra csökkentette. Egy példa lehet az ilyen eredményre, mikor az ügynök csak az egyik irányból érkező forgalmat engedi, nem vált fázist, ugyanakkor ez az átlagos mutatóban nem feltétlenül mutatkozik meg.

4.3 Jutalmazási függvények négy csomópontos úthálózaton

Az átlagsebesség maximalizálás és várakozási idő minimalizálás a nem megfelelő működés okán a MARL megoldások betanításánál eltávolításra került. A TMQ maximalizálás megtartottam az rTMQ maximalizálással való összehasonlíthatóság érdekében. A dolgozat további részében az IQL módon tanított MARL modellek három lépcsős tesztelése kerül bemutatásra, különböző jutalmazási metodikák szerint. Néhány esetben az *y* tengely léptékét sűrítettem a jobb átláthatóság érdekében.



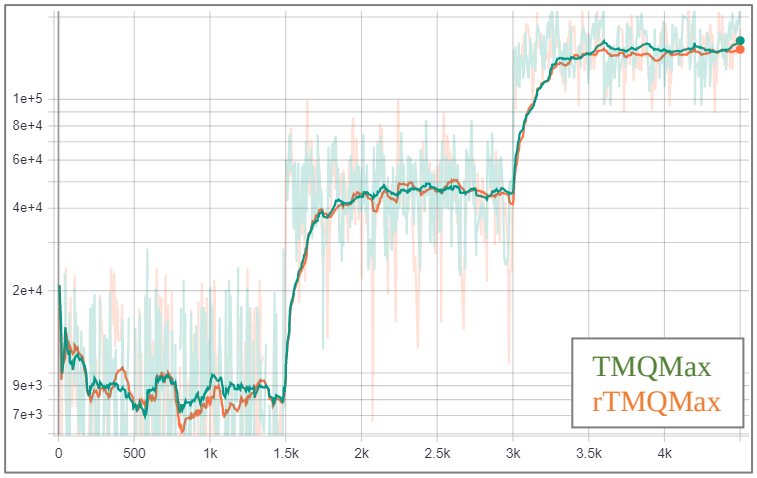
9. ábra: Várakozási idő különböző jutalmazási függvények esetén IQL modelleknél

A várakozási időket tartalmazó grafikon alapján látható, hogy az utazási szám maximalizálás alapú ügynök nagymértékben alulmúlta a többi modell eredményét. Ez minden más tesztelési mérőszámnál is tapasztalható volt, így a továbbiakban az elemzés bizonyos részeiből kivonásra került. Érdekes ugyanakkor, hogy a súlyozott párja a dolgozat eredményei alapján az egyik legjobban működő jutalmazási alap. A grafikonon valódi különbségek csak az alacsony, valamint az erős forgalomú blokkokban láthatók. A közepes forgalom esetén tapasztalható hasonlóságok oka lehet, hogy a legáltalánosabb forgalmi rendhez voltak képesek leginkább alkalmazkodni a különböző jutalmazások szerint tanított ügynökök. CO_2 kibocsátás tekintetében alacsony forgalom esetén a különböző ügynökök hatékonysága nem egyértelműen hierarchikus, közel azonos átlagos szintet értek el, míg közepes forgalom esetén a sorban álló járművek minimalizálásával tanított ügynök működése a többihez képest magasabb kibocsátással járt. Erős forgalomú szituációban a legtöbb mérőszám esetén egyértelmű hierarchia mutatkozik, mely CO_2 kibocsátás esetén a következő: TMQ > Baseline > sorhossz minimalizálás > súlyozott utazási szám maximalizálás. A várakozási idő és helyben álló járművek számának esetében szintén a fenti hierarchia mutatkozik meg. Közepes forgalom esetén a Baseline modell azonban erősen felülmúlta az RL alapú modelleket. Amennyiben a tesztfolyamat alatti átlagsebességeket vizsgáljuk, közepes forgalomnál kezdenek egyértelműbb különbségek kialakulni, mely itt szintén a Baseline modellnek kedvez, sőt ezesetben magas forgalmi dinamika mellett is megőrizte vezető szerepét a súlyozott utazási szám maximalizálás mellett. Többé-kevésbé egyértelmű módon, de közepes forgalom mellett úgy tűnik a fix időtartamú forgalom irányítás nagyon eredményesen teljesít, ezzel részben meg is dőlt a fent megfogalmazott állításom, miszerint az általános forgalmi rendhez képesek leginkább alkalmazkodni az ügynökök. A fix időtartamú Baseline modell sikere azonban korántsem meglepő, mivel az útvonalak közt egyenlően elosztott zöld jelzések átlagos forgalom esetén az

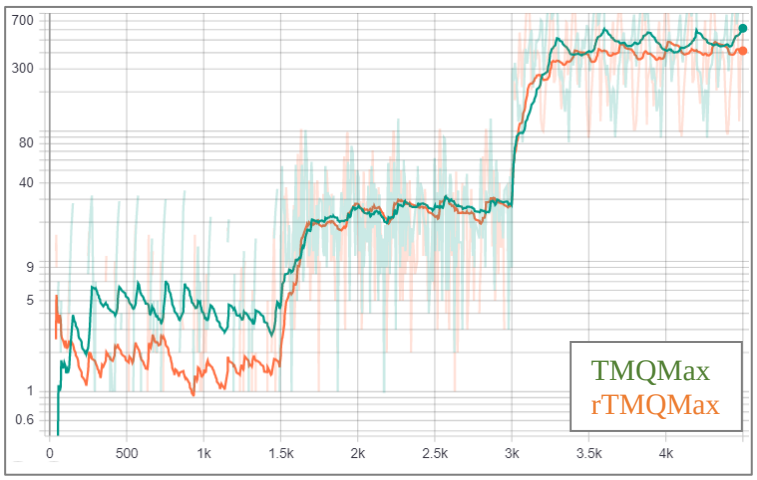
egyik legoptimálisabb módszernek tekinthető, így tehát nem meglepő, ha csak a szélsőségesebb dinamikák során képesek az ügynökök jobb módszert találni. A várakozási idők tekintetében egyértelmű különbség csak erős forgalom esetén jelentkezett, melynél a súlyozott utazási szám maximalizálás töredék átlagidőt produkált a többi ügynökhöz képest.

4.4 Hurokdetektorok nélküli modell

A hurokdetektorok nélküli jutalom számítás hatásosságának mérlegeléséhez a TMQ maximalizálás került módosításra. A jutalom mértéke a kereszteződés átengedőképessége és a sorbanálló járművek számának különbségeként adódik. Eredeti formája szerint az átengedőképességet az úthálózatba belépő járművek száma határozza meg. Stevens és Yeh (2016) tanulmányának nyomán a TMQ jutalmazás módosított formája került kialakításra, melyben az átengedőképességet a kereszteződést elhagyó járművek száma határozza meg, ebből eredően a reversed TMQ maximalizálás (rTMQ) nevet viseli. TMQ és rTMQ maximalizásás esetén a CO_2 és várakozási idők alakulását a 10. ábra és 11. ábra mutatja.



10. ábra: CO_2 kibocsátás TMQ és rTMQ esetén

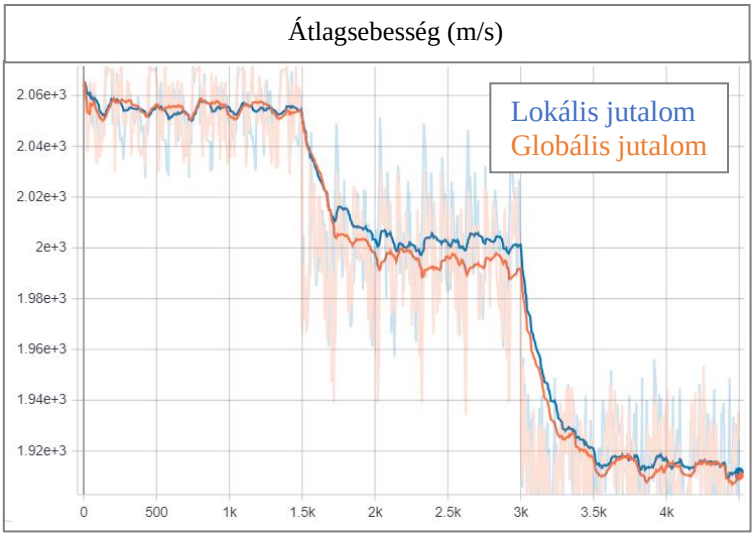


11. ábra: Várakozási idő TMQ és rTMQ esetén

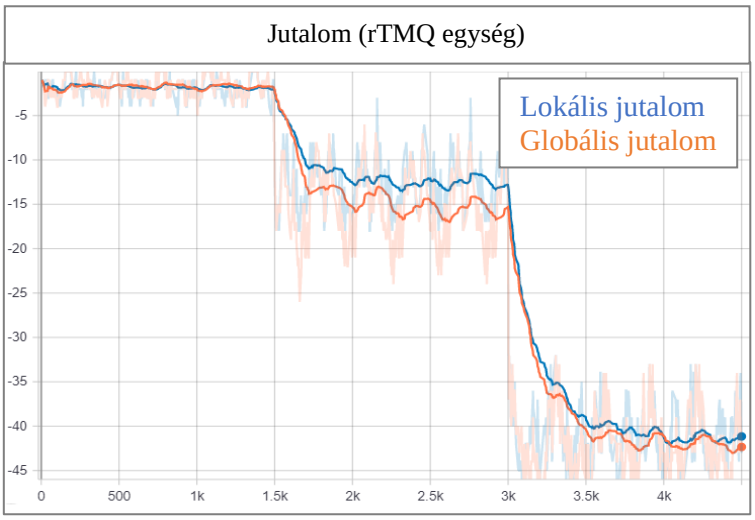
A CO_2 kibocsátás közepes forgalom esetén közel azonos mértékű volt, azonban a gyenge valamint az erős forgalmi dinamika mellett az rTMQ alapú ügynök működése valamivel alacsonyabb összesített CO_2 gázt termelt. A várakozási idő tekintetében is hasonló a forgalmi dinamika hatása, alacsony és erős forgalom mellett már látványosabban alacsonyabb várakozási idő volt tapasztalható. A dolgozat elején közölt hipotézisem a jutalom elsajátítási sikerességre is irányult, mely alacsony és közepes forgalomnál nem jelent meg számottevően, viszont erős forgalomnál már látható volt a javulás. Ez igaz az átlagsebességre is, a forgalom mértéke és a hatékonyságban mérhető előny között egyenes az arányosság. Az eredmények alapján arra lehet következtetni, hogy az újfajta végpont számítás valóban előnyt jelent az ügynök tanítása során, jobban teljesít minden vizsgált szempont szerint, így a kényelmesebb és dinamikusabban a valóságra szabható megoldás jól alkalmazható lehet.

4.5 Lokális és globális jutalom

Globális jutalom számolás esetén az utazási szám maximalizálás szerinti jutalmazási függvény során volt érezhető a legnagyobb javulás, lokális társával ellentétben felülmúlta a Baseline modellt több vizsgált paraméter tekintetében is, azonban így is minden más lokális alapú modellnél gyengébben teljesített. A súlyozott utazási szám maximalizálás, rTMQ maximalizálás és sorbanálló járművek minimalizálásán alapuló jutalmak esetében gyengébben teljesítettek a globális modellek, TMQ maximalizálás esetén pedig a globális és lokális jutalmazás egymással ellentétes helyzetekben teljesített jól. Következtetésként levonható, hogy a globális jutalomszámítás, vagyis a teljes úthálózat figyelembevételéből nem származik egyértelmű előny, inkább zavaró hatásnak tekinthető az általam használt beállítások és változók tekintetében, ugyanakkor az adott környezetben rosszul teljesítő jutalmazások javítására alkalmas lehet. Az alábbi diagramon az rTMQ alapú jutalmazással tanított modellek tesztelése során mért átlagsebesség és jutalom szerzés mértéke látható.



12. ábra: Átlagsebesség lokális és globális jutalomnál



16. ábra: Jutalom lokális és globális jutalmazásnál

Több mérőszám esetén is a közepes forgalom mellett volt tapasztalható a legnagyobb eltérés, holott a korábban bemutatott lokális jutalmazások ennél a dinamikánál teljesítettek alapból is gyengébben a fix időtartamú forgalomirányításnál.

4.6 Ügynök kihelyezés

Meglehetősen nehéz összehasonlítani az IQL és ügynök kihelyezéses jutalmazási párokat, mivel a legtöbb esetben teljesen máshogy teljesítenek a különböző szituációkban. Sorra véve a jutalmazási függvényeket a következő megállapítások tehetők. A súlyozott utazási szám maximalizálás alapú modell esetén a kihelyezett ügynök magasabb CO_2 kibocsátást és várakozó járműszámot eredményezett, ugyanakkor a várakozási időben kevésbé találhatók kilengések az IQL párral szemben. TMQ és rTMQ maximalizálás és sorbanálló jármű minimalizálás esetén a

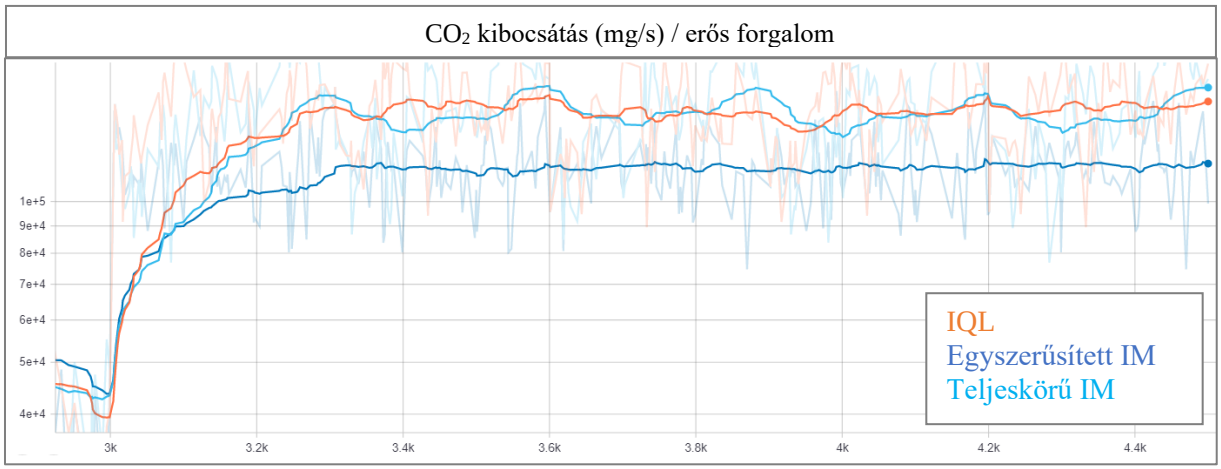
kihelyezett ügynök a legtöbb paraméter tekintetében jobb eredményt ért el a teljes tesztfolyamat alatt, a forgalom dinamizmusa pedig erősen befolyásolta ezt a különbséget. A teszt eredményeinek konklúziója, hogy mindenképp érdemes az egy csomópontban tanított modelleket is figyelemmel kísérni, mivel nem jelenthető ki egyértelműen, hogy megfelelő fejlesztések és kutatás mellett eredményesebb működést érjenek el, mint a párhuzamosan tanuló ügynökök. Továbbiakban érdemes lehet a neurális háló módosításával és transfer learning alkalmazásával is próbálkozni, mely során a kihelyezett ügynökök adott ideig hangolhatják tanulás segítségével a háló súlyait a több csomópontos úthálózaton.

4.7 Információmegosztás mértékének hatása

A dolgozatban három különböző mértékű információmegosztáson alapuló, rTMQ jutalmazással tanított DeMARL modell került összehasonlításra, melyek a következők:

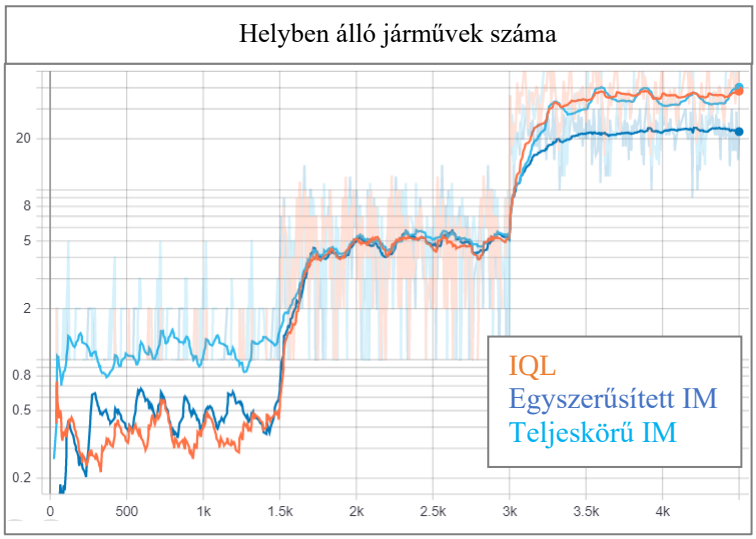
- információmegosztást nélkülöző IQL modell
- egyszerűsített információmegosztással működő modell, melynél az adott időpillanati lámpafázis, az abban eltöltött idő, valamint a detektorok által látott járművek összesített átlagsebessége és száma a megosztás alapja
- teljeskörű információmegosztású DeMARL melynél a lámpafázis, az abban eltöltött idő, valamint sávonként elkülönítve a detektorok által látott járművek átlagsebessége és száma kerül megosztásra.

A tesztfázis eredményei alapján a következő mintázatok voltak megfigyelhetők: A CO_2 kibocsátás tekintetében alacsony forgalomnál a teljeskörű IM járt a legmagasabb kibocsátással, közepes forgalomnál nem egyértelmű a korreláció, erős forgalomnál pedig egyértelműen az egyszerűsített IM teljesített legjobban, ahogy ez a következő diagramon látható.

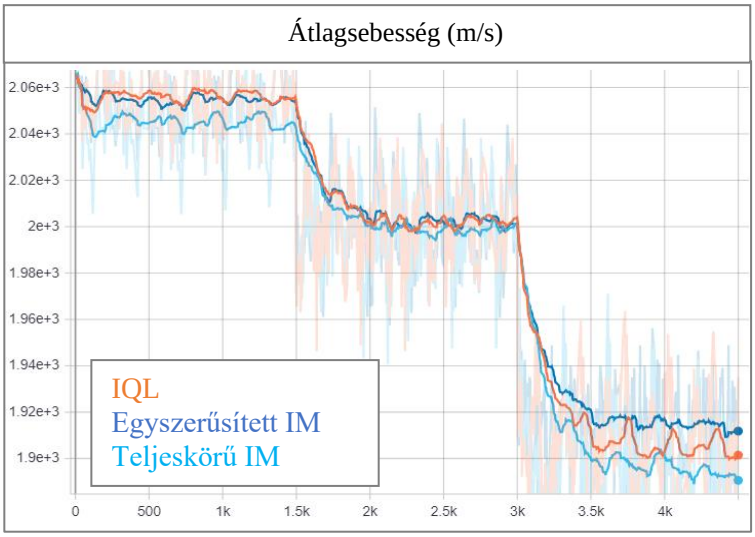


17. ábra: CO_2 kibocsátás erős forgalomnál

A 15-ös ábrán látható helyben álló járművek száma és a várakozási idő között a modellek teljesítménybeli hierarchiája között azonosság látható, alacsony forgalom esetén a teljeskörű IM teljesített legrosszabbul, közepesnél nem határozható meg egyértelműen, erős forgalomnál pedig a CO_2 kibocsátáshoz hasonlóan az egyszerűsített IM modell volt a legsikeresebb. Az átlagsebesség tekintetében annyi eltérés fedezhető csak fel az előző tendenciától, hogy kevésbé nagymértékű erős forgalom esetén az egyszerűsített IM modell előnye, a teljeskörű IM alapú modell pedig a teljes tesztszakasz során egyértelműen alul maradt, ahogy az a 16-os ábrán is látható.



18. ábra: Várakozó járművek száma az információmegosztás mértéke szerint

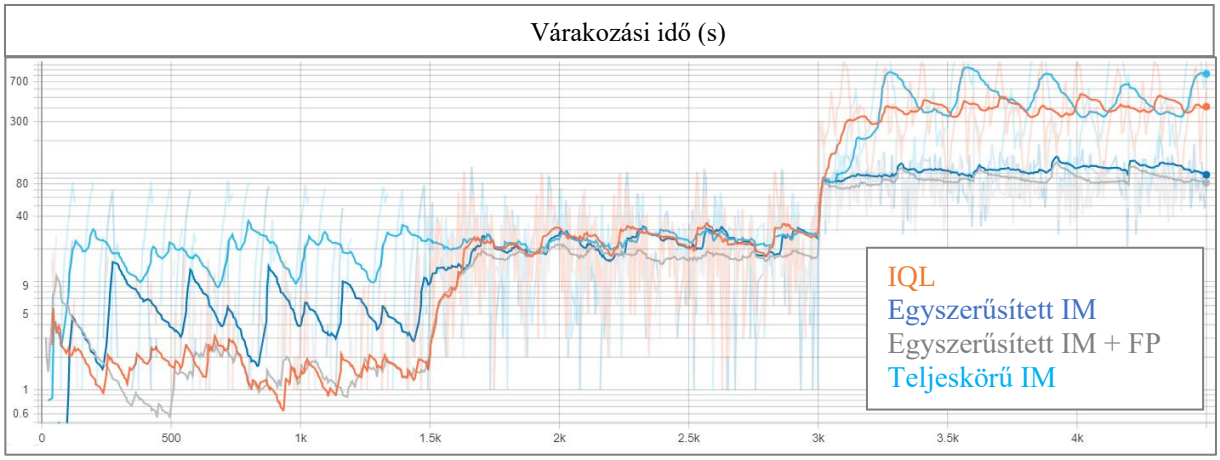


19. ábra: Átlagsebesség az információmegosztás mértéke szerint

Jogos lehet a kérdés, hogy milyen okból kifolyólag teljesít rosszabbul az az ügynök típus, amely a környezetről több információval rendelkezik. Több tanulmány is arra a következtetésre jutott, hogy a túlzottan komplex, vagy elenyésző hasznosságú információval való bővítés rontja a modell tanulási konvergenciáját. Ez gyökerezhet a már tárgyalt felfedezési egyensúly felbomlásában, valamint abban is, hogy olyan sokféle állapot létezik, hogy egy bizonyos része nem bejárható. Az összehasonlítás konklúziója tehát, hogy a vizsgált mérőszámok alapján az egyszerűsített információmegosztás előnyt jelent az IQL modellekhez képest, azonban érdemes figyelmet szentelni a megosztott információ méretére és fontosságára, mivel a javulási tendencia nem lineáris módon emelkedik a bevont információ méretével.

4.8 Fingerprint

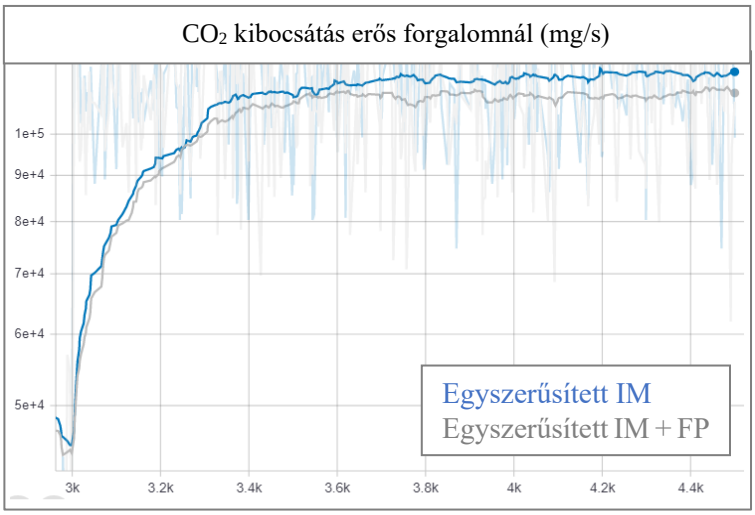
Az egyszerűsített információmegosztással és rTMQ jutalmazással tanított ügynök alacsony és erős forgalom esetén magabiztosan felülmúlta a fix időtartamú forgalomirányítást szinte minden vizsgált paraméter szerint, közepes forgalomnál pedig hasonlóan teljesít ahhoz. Ennek okát korábban azzal magyaráztam, hogy valóban az egyenlően elosztott lámpafázis lehet az optimális megoldás. Az irodalmi áttekintőben részleteztem a FingerPrint, non-stacionárius környezet javítására használt módszert. Egyértelműen kijelenthető az eredmények alapján, hogy miden mérőszám és minden forgalmi dinamika esetén vagy azonos, vagy magasabb hatékonyság érhető el segítségével, negatív hatása a dolgozat körülményei között nem volt tapasztalható. Alább a várakozási időt tartalmazó grafikon látható, kiegészítve az rTMQ maximalizálás alapú, egyszerűsített információmegosztással és FingerPrint-el (FP) tanított ügynök eredményeivel.



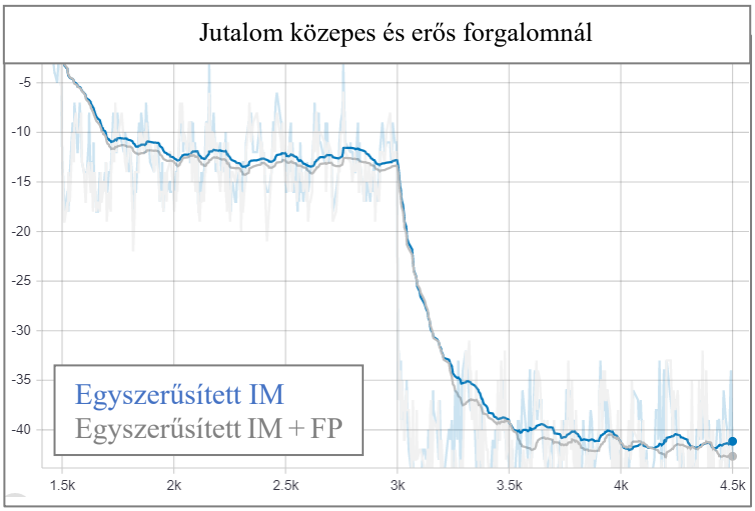
20. ábra: Várakozási idő FingerPrint-nél

A trajektórián látható, hogy a tesztelés teljes fázisában alacsonyabb várakozási idő jellemezte a FingerPrint kiegészítéssel tanított modellt, valamint szembetűnő az is, hogy a FingerPrint nélküli változathoz képest alacsony forgalom esetén nagy mértékű javulás történt, szintén az egész tesztfázis során eredményesebb működésre volt képes.

Nagymértékű volt a javulás többek között a CO_2 kibocsátás terén, erős forgalmi szituációban, mely a fenti diagramon lett feltüntetve. A mérőszámok által jelzett előrelépés ellenére furcsamód a jutalom mértéke közepes és erős forgalomnál kisebb volt, mint a FingerPrint nélküli ügynöknél.



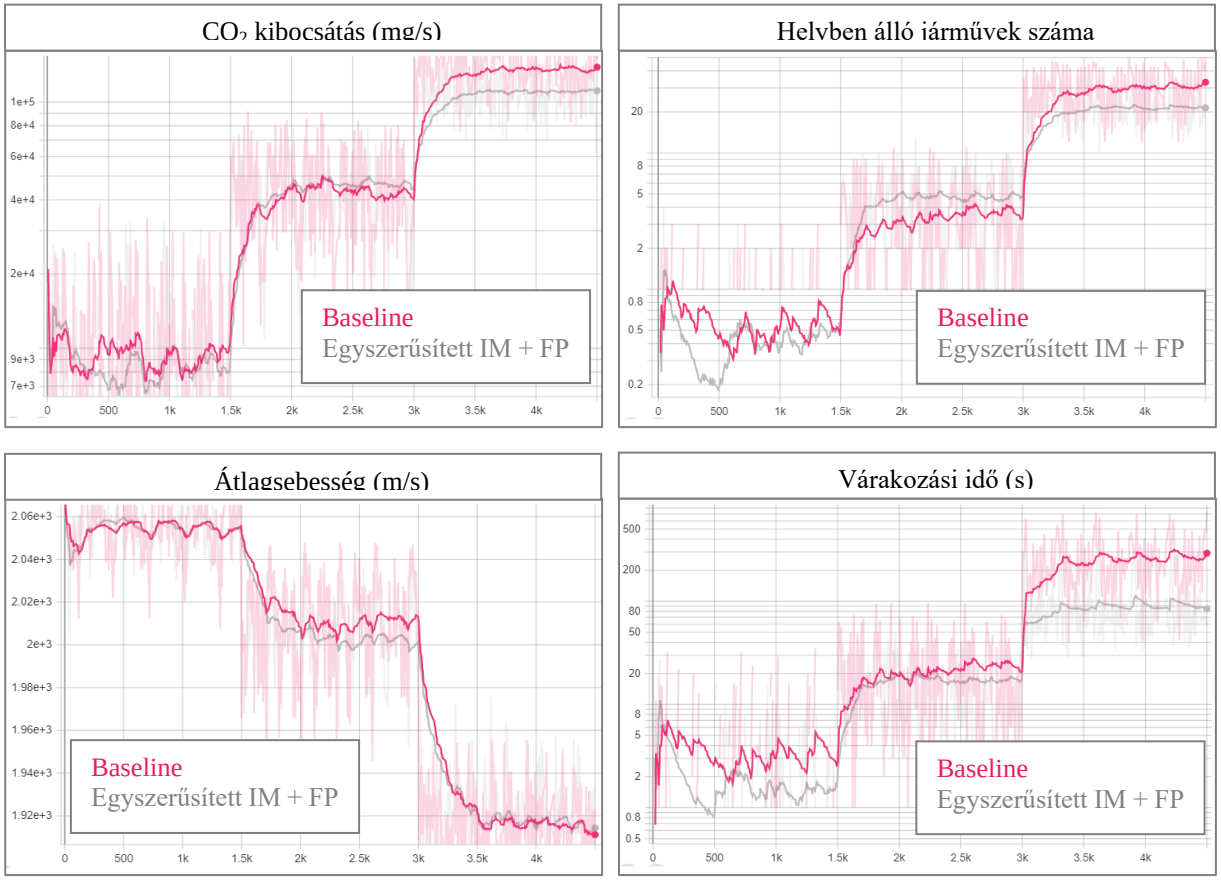
21. ábra: CO_2 kibocsátás erős forgalomnál FingerPrint esetén



22. ábra: Jutalom közepes és erős forgalomnál FingerPrint esetén

Az alábbi négy diagramon a Baseline modellel való összevetése látható a FingerPrint-el kiegészített modellnek. Az adatok azt mutatják, hogy alacsony és magas forgalmi dinamika mellett minden vizsgált paraméter szerint maghaladta az ügynök a Baseline modell működését,

azonban a közepes forgalom továbbra is problémás maradt és a Baseline modell jobban képes allokálni a forgalmat ezekben az esetekben.



23. ábra: Eredmények összehasonlítása

5 Konklúzió

A dolgozat során széleskörűen került vizsgálatra több DQN alapú modell architektúra, hurokdetektorral, és teljes mértékben kamera alapján működő jutalmazási függvény, valamint több, tanulási konvergenciát növelő megoldás közlekedési csomópontok vezérlésére. Az eredmények alapján megállapítható, hogy a Dueling DQN architektúra beépítése kiemelten fontos az üresjáratok során történő lámpaciklus váltások elkerüléséhez, részben ebből kifolyólag a D3QN modell-struktúra tanulási dinamikája nagy előrelépést jelent az egyszerűbb DQN modellekhez képest. A D3QN modell tanítása az első szakaszban egy csomópontos úthálózaton történt, ahol egyértelműen a súlyozott utazási szám maximalizálás alapú ügynök működött a legoptimálisabb módon, míg az egy csomóponton tanított, majd több kereszteződésre kihelyezett szcenárióban az rTMQ alapon tanított ügynök volt a legsikeresebb. Általánosságban elmondható, hogy a kihelyezett ügynök modell megállja a helyét, és felveszi a versenyt a párhuzamos tanítási módszerrel, a fix időtartamú forgalomirányítás felülmúlására képes. A multi-agent környezetben történő tanítás során a választott mérőszámok azt mutatták, hogy a globális jutalomszerzés képes javítani az ügynök eredményein, azonban ez csak az alapból kevésbé sikeresen működő jutalmazások esetén igaz, egyéb esetben a lokális jutalmazás célravezetőbb lehet. A tesztelés során egyértelműen kiderült továbbá, hogy a túlzott mennyiségű, illetve nem megfelelő módon kiválasztott információ megosztása káros lehet a DeMARL környezetben, míg jól megválasztva a működés nagymértékű javulását eredményezi. A vizsgálatba vont FingerPrint megoldás mérlegelése volt a legegyértelműbb, mivel káros hatását semmilyen kombináció esetén nem lehetett mérni, míg több szempont szerint is javította a forgalom allokálási képességet. A teljes analízis során az rTMQ maximalizálással, lokális jutalmazási móddal tanított, egyszerűsített információmegosztással és FingerPrint-el kiegészített állapotterű ügynök bizonyult a leghatásosabbnak. A kezdeti hipotézis tesztelésére, miszerint a gépi tanulás alapú forgalomi irányítás képes felülmúlni a tradicionálisan használt fix időtartamú módszereket, grafikonon hasonlítom össze a két modell működését.

Az eredmények azt mutatták, hogy a hipotézist részben sikerült igazolni, alacsony és magas forgalmi dinamika mellett minden vizsgált paraméter szerint maghaladta az ügynök a Baseline modell működését, azonban a közepes forgalom továbbra is problémás maradt. Ennek okaként továbbra is az optimális, útszakaszok közt egyenlően elosztott fázisidőket sejtem. Szeretném felhívni a figyelmet a dolgozat azon gyengeségére, hogy az eredmények nehezen általánosíthatók, mivel a környezet meghatározása különösen nagymértékben befolyásolhatja azt. A dinamikus beavatkozás érdekében választott egy másodperces RL lépésköz ritkán

használt módszer, mivel a tanulási konvergenciát csökkentheti. Ezen felül érdemes kiemelni, hogy az eredmények alapján a teljes mértékben gépi tanulás alapú forgalomirányítás néhány esetben kiszámíthatatlanul viselkedhet, mely nem megengedhető biztonsági szempontból. Ennek ellenére a jelenleg használt módszerek bővítéseként, valamint jövőbeli fejlesztésével alternatívájaként is használható lehet.

6 Jegyzékek

6.1 Irodalomjegyzék

Abdoos, M., Mozayani, N., & Bazzan, A. (2011). Traffic light control in non-stationary environments based on multi agent q-learning. *14th International IEEE Conference on Intelligent Transportation Systems.*

Araghi, S., Khosravi, A., Johnstone, M., & Creighton, D. (2013). Q-learning method for controlling traffic signal phase time in a single intersection. *16th International Conference on Intelligent Transportation Systems.*

Arel, I., Liu, C., Urbanik, T., & Kohls, A. (2010). Reinforcement learning-based multi-agent system for network traffic signal control. *T Intelligent Transportation Systems, 4*(2), 128-135.

Busoniu, L., Babuska, R., & De Schutter, B. (2008). Busoniu, L., Babuska, R., & De Schutter, B. (2008). A Comprehensive Survey of Multiagent Reinforcement Learning. *Transactions on Systems, Man, and Cybernetics*, 156–172.

Chu, T., Wang, J., Codecà, L., & Li, Z. (2019). Multi-Agent Deep Reinforcement Learning for Large-scale Traffic Signal Control. *IEEE Transactions on Intelligent Transportation Systems*.

Cools, S.-B., Gershenson, C., & DâzHooghe., B. (2013). Self-organizing traffic lights: A realistic simulation. *Advances in applied self-organizing*, old.: 45-55.

Foerster, J., Nardelli, N., Farquhar, G., & Afouras, T. (2018). Stabilising Experience Replay for Deep Multi-Agent Reinforcement Learning.

Gartner, N. H., Stamatiadis, C., & Tarnoff, P. J. (1994). Development of Advanced Traffic Signal Control Strategies for Intelligent Transportation Systems: Multilevel Design. *TRANSPORTATION RESEARCH RECORD*, 98-105.

Hajbabaie, A., & Benekohal, R. (2013). Traffic Signal Timing Optimization Choosing the Objective Function. *Journal of the Transportation Research Board*.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *International Conference on Computer Vision*, 1026-1034.

Koonce, P., Rodegerdts, L., Lee, K., Quayle, S., Beaird, S., Braud, C., . . . Urbanik, T. (2008). *Traffic Signal Timing Manual.* United States. Federal Highway Administration.

Lin, L. (1992). Reinforcement learning for robots using neural networks. *Computer Science*.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing Atari with Deep Reinforcement Learning.

Mousavi, S., Schukat, M., & Howley, E. (2017). Traffic light control using deep policy-gradient and value-function-based reinforcement learning. *IET Intelligent Transport Systems, 11*(7), 417-423.

Nguyen, T. T., Nguyen, N. D., & Nahavandi, S. (2019). Deep Reinforcement Learning for Multi-Agent Systems: A. *Institute for Intelligent Systems Research and Innovation*.

Onion, A., Sullivan, M., & Mullen, M. (2009). First electric traffic signal installed. HISTORY: A&E Television Networks.

Roess, R., Prassas, E., & McShane, W. (2004). *Traffic engineering.* Prentice Hall.

Stevens, J., & Yeh, C. (2016). Reinforcement Learning for Traffic Optimization.

Sutton, R. (1988). Learning to predict by the methods of temporal differences. *Machine Learning, 3*, 9–44.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning, second edition.* MIT Press.

Tan, & Ming. (1993). Multi-agent reinforcement learning: Independent vs. cooperative agents. *Proceedings of the tenth international conference on machine learning,*, 330–337.

van Hasselt, H., Guez, A., & Silver, D. (2016). Deep reinforcement learning with double q-learning. *Conference on Artificial Intelligence.*

Varaiya, P. (2013). The max-pressure controller for arbitrary networks of signalized intersections. In *Advances in Dynamic Network Modeling in Complex Transportation Systems* (old.: 27-66). Springer.

Wang, Z., Schaul, T., Hessel, M., van Hasselt, H., Lanctot, M., & de Freitas, N. (2015). Dueling network architectures for deep reinforcement learning. arXiv.

Watkins, C. (1989). *Learning from Delayed Rewards.* Cambridge University.

Watkins, C., & Dayan, P. (1992). Technical Note: Q-Learning. *Machine Learning, 8*, 279-292.

Zhang, R., Ishikawa, A., Wang, W., B., S., & Tonguz, O. (2020). *Using Reinforcement Learning with Partial Vehicle Detection for Intelligent Traffic Signal Control.* Pittsburgh: Machine Learning Department, Carnegie Mellon University.

Zhao, D., Dai, Y., & Zhang, Z. (2012). Computational intelligence in urban traffic signal control: A survey. *Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 485-494.