Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Hálózati Rendszerek és Szolgáltatások Tanszék

Bauer László Dániel

AZ 5G HÁLÓZATI SZELETELÉS SZIMULÁCIÓJA ÉS LEHETŐSÉGEI

KONZULENS

Dr. Huszák Árpád

BUDAPEST, 2023

Tartalomjegyzék

Összefoglaló

Az utóbbi években az ötödik generációs mobilhálózatok térhódítása számos újszerű funkcionalitás bevezetését tette lehetővé, ezen lehetőségek jelentős része ugyanakkor a mai napig kihasználatlanul várja a szükséges fejlesztéseket és koncepciókat. Ezen korai fázisát élő vívmányok között tarthatjuk számon az 5G *hálózati szeletelést*, vagy elterjedtebb angol nevén *network slicing* technológiát. Az eljárás lényege, hogy a hálózatot „szeletekre" bontjuk, és az így kapott egységek forgalmát különálló követelmények szerint kezeljük. A három triviális szelet a „hagyományos" eMBB (enhanced Mobile Broadband, konvencionális felhasználói mobilinternet-szolgáltatás), a gépi adatforgalom-fókuszú mMTC (massive Machine-Type Communication, sűrűn és nagy mennyiségben elhelyezett IoT-eszközök kiszolgálása), valamint a leginkább kritikus URLLC (Ultra-Reliable Low Latency Communication, különösen nagy megbízhatóságot és garantáltan kis késleltetést igénylő alkalmazások adatforgalma). Mivel az egyes alkalmazási területek drasztikusan eltérő elvárásokat támasztanak a hálózattal szemben, szükségszerű a problémát részekre bontani, és párhuzamosan futó, mégis egyéni megoldásokat találni.

A dolgozat elsőként a *network slicing* koncepcióját mutatja be, elemezve a szabvány nyújtotta kereteket, a már megvalósított szolgáltatások jellemzőit és eredményeit, valamint a fejlesztési lehetőségeket és implementációs bizonytalanságokat. A koncepció gyakorlatba ültetésére és a szeletek közti erőforrás-elosztásra többféle technikai megoldás is lehetőséget ad, amelyek áttekintő és összehasonlító jelleggel kerülnek kifejtésre. A gépi tanulás módszerek a hálózatoptimalizáció területén is egyre népszerűbbek, így érdemes a hálózatszeletelés, illetve hálózati erőforrás menedzsment esetén is megvizsgálni hatékonyságukat.

A legnagyobb hangsúlyt a saját fejlesztésű szimulációs rendszerrel elért eredmények bemutatása kapja, amelyet a dolgozat a rendszer lényegre törő ismertetésével alapoz meg. A kutatási koncepció újszerűsége abban rejlik, hogy az egyes hálózati szeletek erőforrás-elosztását mesterséges intelligenciával, azon belül is mély megerősítéses tanulással vezéreljük. A vezérlés egy, a szimulátorban számos kísérletet végrehajtó ágens betanításán alapul, amely a hálózat működésének optimalitását célszerűen, függvény alapján megállapító jutalom maximalizálására törekszik. Az így kapott vezérlés valós idejű beavatkozásra képes, ráadásul nemcsak egy időpillanat statikus optimalitására törekszik, hanem a dinamikus, adott időintervallumot felölelő szimuláció hálózatának optimalizálását célozza.

A szimulációs rendszer a felhasználók helyzetét egy forgalomszimulátor, a SUMO (Simulation of Urban MObility) által generált pozíció-adatok formájában kapja meg, így nem fenyeget az a veszély, hogy az ágens adott, pontosan ismétlődő mobilitási mintákat tanulna meg. Az így, minden ciklusban változó járműforgalomra futtatott *network slicing* szimuláció eredményeit adatvizualizációs eljárásokkal szemléltetjük, majd következtetéseket vonunk le a megoldás hatékonyságára vonatkozóan. A vizsgálat fő fókuszát az adaptivitás, a konvergencia és a már ismert megoldásokhoz viszonyított hatékonyság jelenti. Mindemellett az eredmények függvényében javaslatot fogalmazunk meg arra vonatkozóan, hogy a vizsgált megközelítés milyen módon és mértékben illeszthető bele a valódi 5G hálózatok működésébe, kitérve a szimuláció korlátaira és az üzleti szempontok figyelembevételére egyaránt.

Abstract

In recent years, the emergence of 5th generation mobile networks has enabled the introduction of many new functionalities, but many of these opportunities remain untapped, awaiting the necessary developments and concepts. Among these yet to be leveraged areas is 5G *network slicing* technology. This approach involves dividing the network into "*slices*" and managing the traffic of the resulting units according to separate requirements. The three trivial *slice*s are the 'traditional' eMBB (Enhanced Mobile Broadband, conventional end-user cellular internet service), the machine-to-machine traffic-focused mMTC (Massive Machine-Type Communication, serving dense and high-volumes of IoT devices) and the most critical one, URLLC (Ultra-Reliable Low Latency Communication, traffic for applications requiring particularly high reliability and guaranteed levels of low latency). As each application area has drastically different requirements for the network, it is necessary to break the problem down into parts and find solutions that may be run in parallel, and yet provide individual service.

The paper first introduces the concept of *network slicing*, analyzing the framework provided by the standard, the characteristics and results of already implemented services, as well as the development opportunities and implementation uncertainties. Several technical solutions may be used to put the concept into practice and to allocate resources between *slices*, which are presented in an overview in a comparative manner. Machine learning methods are also becoming increasingly popular in the field of network optimization, so it is worthwhile to investigate their effectiveness in the case of *network slicing* and network resource management.

The main focus is on the presentation of results obtained with the self-developed simulation system, which are introduced by a concise description of the system. The novelty of the research concept lies in the fact that the resource allocation of each *network slice* is controlled by artificial intelligence, which, in this case, is an approach based on deep reinforcement learning. The control is built on the training of an agent, which performs a number of experiments in a simulator, with the goal of maximizing the reward that determines the optimality of the network operation in a sensible, function-based manner. The resulting control is able to intervene in real time, moreover, it does not only aim at the static optimality for a single timeframe, but also at the optimization of the dynamic simulated network over a given time interval.

The simulation system receives the position of the users in the form of position data generated by a traffic simulator called SUMO (Simulation of Urban MObility), so there is no risk of the agent learning specific, exactly repeated mobility patterns. The results of the *network slicing* simulation run in this way, for vehicle traffic that changes every cycle, are illustrated using data visualization techniques, and conclusions are drawn about the effectiveness of the solution. The main focus of the study is on adaptivity, convergence and effectiveness compared to existing solutions. In addition to this, depending on the results, proposal for how and to what extent the approach can be integrated into real 5G networks is given, addressing both the limitations of the simulation and the business requirements.

Támogatói köszönetnyilvánítás

Jelen dolgozat elkészültét, illetve az ehhez kapcsolódó kutatási tevékenységet a Nemzeti Kutatási, Fejlesztési és Innovációs Hivatal NKFI Alapja által felkínált Tématerületi Kiválósági Program (TKP) támogatta.

1 Bevezetés

Korunk hálózati kihívásainak jelentős része a felhasználói és üzleti igények korábban nem tapasztalható mértékű heterogenitásával kapcsolatos. Míg a negyedik generációs LTE alapvetően homogén igénymodellre épült, addig az 5G szimultán küzdelemre kényszerül az eltérő késleltetést és adatátviteli rátát érintő elvárásokkal. Értelemszerűen ebben az esetben módosított architektúrára van szükség, amelyet a szabványok kellően definiálnak, ám mindez önmagában még nem garantálja az erőforrások elosztásának optimalitását. Ez a kihívás jelenti az alapvető motivációt a determinisztikus eljárások helyett alkalmazható mesterséges intelligencián alapuló vezérlés kifejlesztéséhez és vizsgálatához.

Jelen TDK dolgozat első harmada az 5G rendszerekben bemutatott *hálózati szeletelés* jellemzését, valamint a kutatási téma motivációját hivatott bemutatni. A *network slicing* technológia elterjedtsége napjainkban még nem elegendő ahhoz, hogy mi magunk is kipróbálhassuk a gyakorlatban, ám a legtöbb jel arra mutat, hogy jövőbiztos eljárásról beszélhetünk. Az 5G hálózatokban debütáló hálózati szeletek koncepciója terén a hatodik generáció esetén további bővülés várható, minden eddiginél kritikusabb és nagyobb internetkapcsolati igényekkel működő felhasználási célok kiszolgálására. Az erőforrás-menedzsment a hálózatvezérlési feladatok közül is kiemelkedően nagy fejlődés előtt áll, kiváltképp az olyan technológiai újítások bevezetésével, mint a mesterséges intelligencia.

Az AI területén belül a gépi tanulás [1] feleltethető meg leginkább a hálózat-orkesztráció kívánalmaira alkotott megoldási koncepcióknak, mivel ez a típus képes olyan ágens tanítására, amely azonnali, valós idejű döntések meghozatalára specializálódik. A dolgozat terjedelmének optimalizálása miatt nem kerül bemutatásra a gépi tanulás összes válfaja, de a három alaptípusból választott megerősítéses tanulás [2] releváns előnyeire a későbbiekben kitérek. A felügyelt tanulás [3], amely során felcímkézett adathalmazzal történik a tanítás, valamint a címkék nélküli, rejtett mintázatok felfedezésére alkalmas felügyeletlen tanulás [4] az erőforrás-vezérlés céljára koncepcionális eltérések miatt kevésbé alkalmazható.

Mivel a fő kihívások egyike a valós hálózati adatok rendelkezésre állásának hiánya, feltétlenül szükségessé válik a szimuláció alkalmazása. Ezzel ugyancsak elkerülhetők az aktív beavatkozással járó kísérletezés negatív következményei, hiszen a valós mobilhálózatok vezérlésének esetleges szuboptimalitása azonnali instabilitást, rendelkezésre állásban tapasztalható kieséseket, illetve az ebből fakadó ügyfélelégedettség-csökkenést vonná magával. A modell megalkotása tökéletlen és kihívást jelentő feladat, ugyanakkor a megerősítéses tanulás ágensének kísérletező jellegű tanulási folyamatát az így kapott szimulátor remekül kiszolgálja.

A dolgozat második harmadában kitűzött cél interdiszciplinaritására tekintettel bemutatásra kerül az alkalmazott szimuláció architektúrája, amelyben nagy hangsúlyt kap a komponensek kapcsolatának bemutatása és indoklása. A rendszer alapját a szakdolgozatom [1] keretein belül megalkotott és bemutatott, hasonló célkitűzésű szoftveregyüttes képezi, ugyanakkor jelen esetben az optimalizáció alapvető koncepciója merőben eltérő. Az alkalmazott 5G szimulátor ugyancsak eltérő, amely nagyban hozzájárul az optimalitás megtalálásának lehetőségeihez.

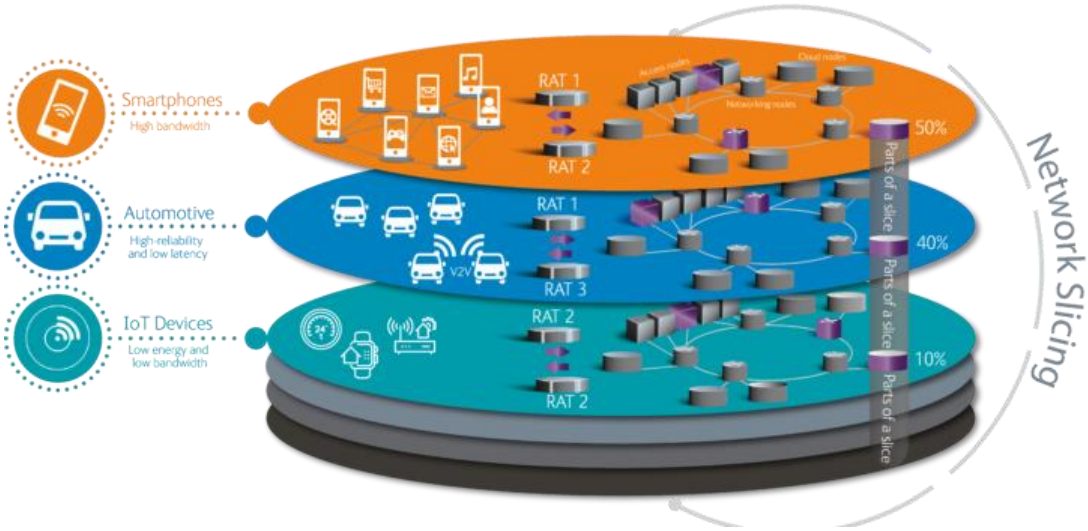
Végül a legfontosabb szekció következik, amely az új szimulációs rendszerben elért eredményeket taglalja. A következtetéseket adatvizualizációval támasztom alá, amelyek betekintést nyújtanak a gépi tanulási folyamatba, és összehasonlítják az annak eredményeként kapott ágens döntéshozatalának jellemzőit a mesterséges intelligencia nélküli optimalizációs eljárások képességeivel. Az eredmények értékelése során nagy hangsúlyt kap az 5G koncepcióiba való beilleszthetőség, és az ágens alkalmazásával elérhető előnyök gyakorlati haszna.

2 A hálózati szeletelés koncepciója

2.1 A szimultán jelentkező igények bemutatása

Az eddigiekben felvezetett kihívások egyike, hogy a heterogén hálózati felhasználói igények korát éljük. Ahogyan az az Intel vállalat összefoglalójából [6], valamint a [7] és [8] jelű 2022-es és 2021-es cikkből is kiderül, olyan újszerű ipari és lakossági felhasználási mintázatok válnak jellemzővé, mint az intelligens gyári technológiák, az önvezető járművek, a minden eddiginél sűrűbben elhelyezett IoT-eszközöket alkalmazó mezőgazdasági, ipari és városi infrastruktúra, vagy épp a triviálisan szükséges, de nem kiemelten kritikus szélessávú internetelérést igénylő virtuálisvalóság-szolgáltatások, online videojátékok és nagyfelbontású multimédia-adatfolyamok. Emellett maga a hálózat is jelentős önálló adatátviteli igénnyel rendelkezik, például az edge Computing és a növekvő adatforgalmú fronthaul megoldások elterjedésével. Utóbbiak pedig a mesterséges intelligencia eddig nem látott felhasználási lehetőségeit hozzák el a holnap hálózatába.

Mindezek nemcsak bitráta terén támasztanak változó és különböző elvárásokat, hanem a késleltetés és ennek időbeli ingadozása, a rendelkezésre állás, a bithiba-arány és az adatátvitel folytonossága is eltérő módon és mértékben optimális számukra. Ebből adódóan az igények kiszolgálására is különféle eljárások válnak szükségessé, amelyek az 5G *hálózati szeletelés* esetén három alaptípusba sorolhatók, amelyeket az 1. ábra szemléltet. Technikai szempontból ennél több *slice* is megvalósítható, és üzletileg potenciálisan sikeres koncepció lehet a szeleteket önálló szolgáltatásként értékesíteni, de jelen dolgozat szempontjából a három alaptípus bőven elegendő, az ezekre tett megállapítások kvalitatív tekintetben eltérő számú *slice* esetén is helytállóak volnának.



1. ábra: Az 5G hálózati szeletelés logikai felépítése

Forrás: https://www.viavisolutions.com/en-us/sites/default/files/network_slicing.png

Az enhanced Mobile Broadband (eMBB) [9] a leginkább konvencionális szolgáltatást nyújtja. Célja a nagy bitrátát és elfogadhatóan kis késleltetést igénylő, de ezekre nézve időben nem kritikus felhasználási területek kiszolgálása. Ezekre jellemző példa a 4K-nál nagyobb felbontású videó-streaming, az AR és VR videojátékok, a valós idejű videóanalitika és a telemedicina kevésbé kritikus válfajai. Az eMBB tehát jellemzően kiemelkedően magas adatátviteli sebességet kínál, de nem számol akkora erőforrás-tartalékolással és rendelkezésre állási követelményekkel, amelyek alkalmassá tennék a kritikus alkalmazási területekre.

A massive Machine-Type Communication (mMTC) [10] az IoT-eszközökre optimalizált *network slicing*-megoldás, amely a késleltetésre és a bitrátára egyáltalán nem szenzitív. Ugyanakkor az akkumulátorról működő berendezések ritka aktív fázisai miatt a rendelkezésre állásra, az efféle eszközök térbeli elhelyezkedésének sűrűsége okán pedig a szolgáltatás koncentrációjára annál érzékenyebb halmazt alkot. Az mMTC jellemzően az okosváros- és okosépület-infrastruktúra [11], az intelligens közlekedési rendszerek, a gyárvezérlés, a mezőgazdaság és egyes egészségügyi szolgáltatások hasznára válhat a leginkább.

A betűrendi sorrendben utolsó, ám elvárásai tekintetében elsőként említhető Ultra-Reliable Low-Latency Communication (URLLC) [12] állítja a legnagyobb kihívás elé az 5G hálózatokat, ugyanis ezesetben valódi garanciát kell nyújtani a megbízhatóságra, a rendelkezésre állásra és az adatátviteli paraméterek időbeli állandóságára. Mindössze néhány milliszekundum késleltetést és már-már eMBB-szintű sebességet biztosít az olyan kritikus alkalmazásokhoz, mint az orvosi szenzorok, a távdiagnosztika, az [13] jelű forrás alapján távműtétek, a valós idejű VR alapú képközvetítés, az önvezető járművek, a [14] cikk szerint a smart grid, a távolról végzett szabályozási rendszerek és a hálózatba kapcsolt kritikus gyártásfolyamatok.

2.2 A network slicing architektúrája

A három alapvető szolgáltatási típus, illetve az ezekből származtatott további verziók merőben eltérő felhasználói értéket nyújtanak, megvalósításuk ugyanakkor nagy mértékben megegyezik. A virtualizált, szoftveres hálózati entitások korát éljük [16], azon belül is a skálázhatóság és a natívan felhőalapú rendszerek térhódítása zajlik napjainkban. Ennek értelmében különösen fontos szempont, hogy az egyes komponensek milyen életciklussal kezelhetők, és milyen módon járulnak hozzá a hálózat dinamikus elaszticitásához. Elsőként tehát nem a hálózati szeleteket megvalósító entitások architektúráját, hanem a szeletek felépítésének logikai sorrendjét ismertetem.

Az Ericsson end-to-end architekturális és orkesztrációs összefoglalója [17] a *network slicing* életciklus-menedzsmentjét 8 csoportba osztja, függetlenül attól, hogy előre konfigurált, dinamikusan változtatható, vagy API használatával szabadon létrehozható szeletről beszélünk. A három kategória értelmezhető megvalósíthatóság nehézsége szerinti növekvő sorrendként is, de ezzel párhuzamosan a megoldástól várható bevétel is növekedhet. Mindez azonban az életciklus szempontjából kevésbé fontos, így az üzleti koncepció tárgyához sorolva érdemes bemutatni.

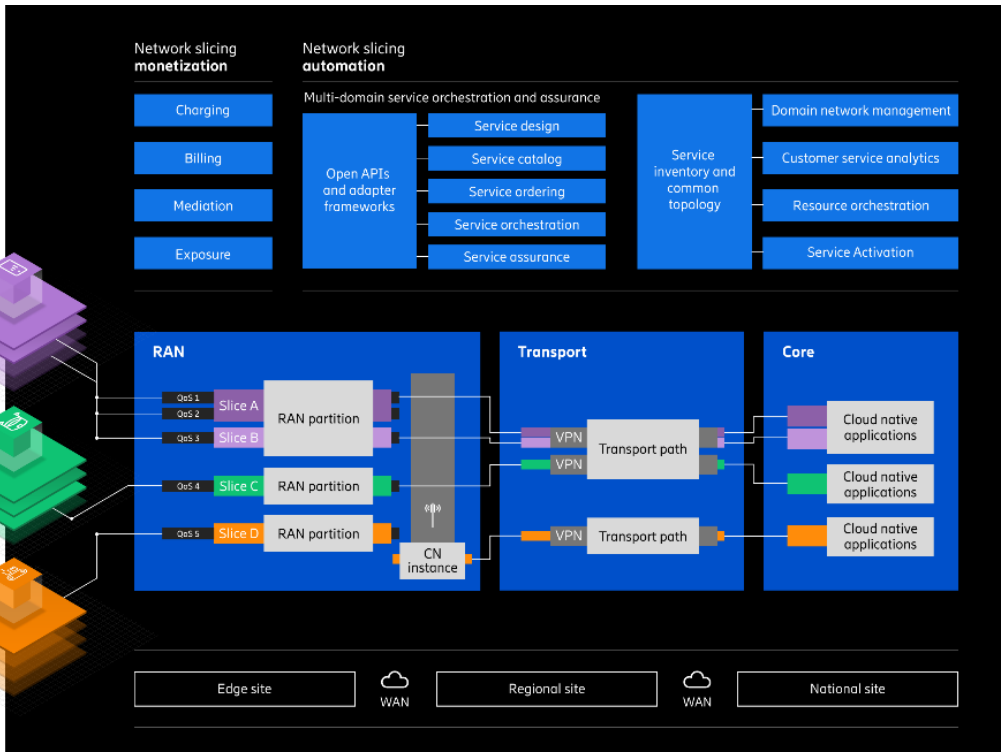
A nyolc lépés a *slice* létrehozásával kezdődik, amely a három opció egyike által triggerelt folyamat. Ezután következik a partnerek konfigurálása, amely során a releváns paraméterek beállításra kerülnek. Harmadik lépésként a hálózati eszközök és az előfizetők hozzáadása történik, amely megalapozza a szelet által létrejövő adatkapcsolat szerkezetét. Negyedik teendőként a szolgáltatás megrendelése következik, amelynek a kereskedelmi katalógus teremt alapot. Ebből következik a monetizációs és hálózat-policy kontroll elindítása, amely üzleti modellek palettájából enged választást az ügyfélnek.

Az utolsó lépéshármas a *slice* által nyújtott szolgáltatásminőségi garanciákat alapozza meg, amely világba triviálisan a QoS kezelésével lépünk be. Ez a három megközelítés típusától függően lehet előre konfigurált *slice*-okból, katalógusból, vagy épp API-n keresztül történő választás. Egy szomszédos absztrakciós szintre lépve a hálózati szelet LCM (Local Congestion Mitigation) entitása következik, amely a skálázható felhőalapú rendszerekhez hasonlóan dinamikus szeletpéldány-létrehozást is vezérelhet. Végül, de korántsem utolsó sorban a hibakezelés és a teljesítménymonitorozás elindítása következik, amely a szolgáltatás pontosabb testreszabását teszi lehetővé.

Igencsak érdekes kihívás egy ilyen, end-to-end szinten önálló architektúra megvalósítása egy valódi hálózatban. Noha léteznek olyan megoldások, amelyek akár LTE rendszerek esetén is a *hálózati szeletelést* közelítő funkcionalitást valósítanak meg, mégis alapvetően standalone 5G hálózat esetén aknázható ki leginkább a slicing. A kompatibilitás elősegítése üzleti szempontból is fontos, de a [17] jelű forrás rendszerszintű áttekintéséből is látható módon a szeletek korántsem monolit entitások. A rádióhálózatban éppúgy szükséges definiálni egy, az adott *slice*-hoz tartozó komponenst, mint a transzport és a maghálózati szinten.

A felhasználókhoz legközelebb elhelyezkedő RAN feladata a rádiókapcsolat fizikai attribútumainak dinamikus optimalizációja a szelet típusától, az átviteli közeg és a hálózat aktuális állapotától, valamint a terheléstől függően. A megvalósítandó szolgáltatás függvényében lehetséges dedikált hardver használata is, ugyanakkor ez korántsem elengedhetetlen. A rádiós erőforrások menedzsmentje mellett az egy hardveren futó különböző szeletpéldányok logikai szeparációjáról is gondoskodnunk kell.

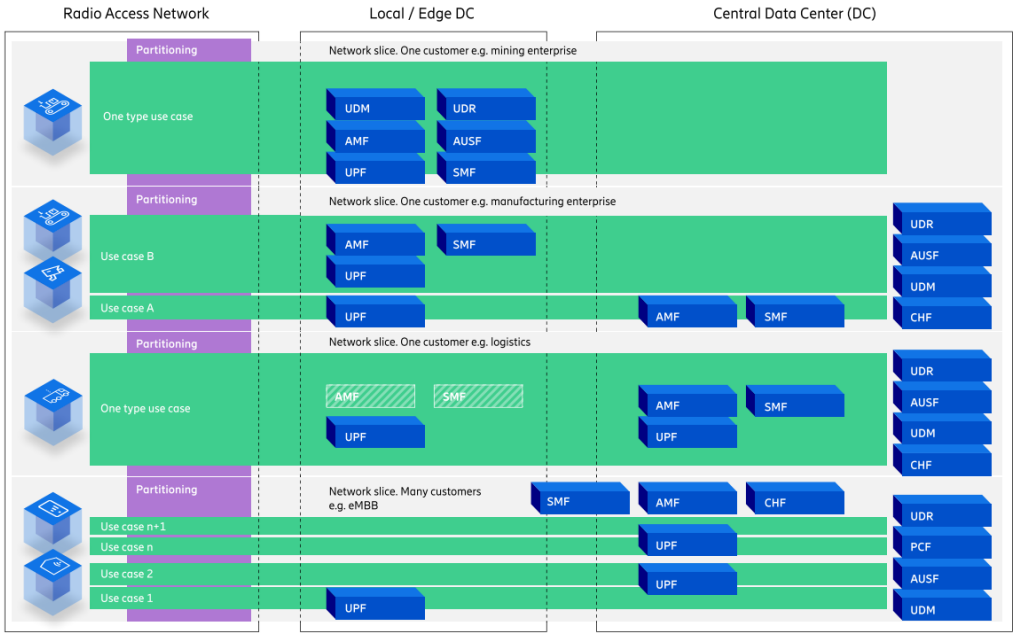
A transzport rétegben VPN alkalmazásával akár azonos útválasztás mellett is garantálható a logikai szeparáció, de ezesetben különös gondot kell fordítani az erőforrások prioritáshoz hű allokációjára, amely jelen dolgozat témájának integrális részét képezi. A maghálózattal szemben ez a réteg kritikus a különböző domének sorrendjének és együttműködésének optimalitására, amelyet nyílt API esetén lehet a leginkább garantálni.



2. ábra: Az 5G hálózati szeletelés architekturális áttekintése

Forrás: https://www.ericsson.com/49c9b9/assets/local/networks-slicing/docs/ericsson-network-slicing-infographic.pdf

A legösszetettebb architektúrát a maghálózatban találhatjuk, mivel a lehetőségek legnagyobb halmaza itt tárul a technológiát alkalmazni vágyók elé. Az Ericsson összefoglalója például rögtön ötféle megközelítést ajánl (3. ábra), amelyek közül az adott felhasználási cél alapos ismeretében lehet választani. Fontos leszögezni, hogy a *network slicing* esetén lehetővé válik az olyan alapvető hálózati entitások többszörözése és szimultán működtetése, mint a User Plane Function (UPF), az Access and Mobility Management Function (AMF), vagy épp a Session Management Function (SMF). Ezáltal tisztán látható, hogy a szeletelés az adatfolyamok és szolgáltatások tényleges szeparációjának magas szintjét valósítja meg.



3. ábra: Az Ericsson által ajánlott deployment opciók az 5G hálózati szeletelés maghálózati implementációjához

Forrás: https://www.ericsson.com/49c9b9/assets/local/networks-slicing/docs/ericsson-network-slicing-infographic.pdf

A javaslatok közül a legegyszerűbb a bányászati vállalat hálózati igényeire ajánlott megoldás, ahol mindössze egy felhasználási cél kiszolgálása szükséges, így a hálózati entitások többszörözése szeleten belül nem érné meg. Rögtön szembetűnik ugyanakkor a második dimenzió, amely az edge Computing és a teljesen központosított megoldás közötti skálán helyezi el az opciókat. Az első javaslat például teljes egészében az edge területre korlátozódó architektúrát vázol fel, amely lokális lefedettségű privát üzemi hálózatok esetén lehet igazán kifizetődő.

A választási lehetőségeket tovább vizsgálva kiemelhetjük az eMBB típusba leginkább sorolható felhasználási területek halmazát, amely erős központosítást javasol, de a partíciók száma is igen magas. Ezesetben az Ericsson mindössze az UPF-et helyezné az edge területre, és ezt is csak céltól függően opcionálisan. Ezzel szemben az SMF az edge és a központi terület között helyezkedik el, amely erős összeköttetést valósíthat meg a két réteg között. Ugyancsak említésre méltó az UPF példányok nagyobb számossága is a központi adatcentrum szintjén, amely további szeparációt jelent az egyes partíciók számára, akár *slice*-on belül is.

A tipikusan gyári környezetben alkalmazható ajánlásokra is érdemes kitérni, ugyanis a közeljövő iparát a hálózati kapcsolatok minőségére és kvantitatív jellemzőire vonatkoztatott drasztikus igénynövekedés jellemzi. Ezesetben az eMBB-hez hasonlóan felhasználási területenként elkülönített UPF-et alkalmazhatunk, de ennek helye sokkal inkább az edge, mintsem a data center. Ennek okai között említhetjük az ipari szabályozórendszerek beavatkozási késleltetésére vonatkozó követelmények szigorúságát, az adatok kiértékelésének azonnaliságát és az egyes gyártási folyamatok nagy mértékű jellegi eltéréseit. Világosan látható tehát, hogy a *network slicing* architektúrája igen sokféle lehet, és kíváncsian várhatjuk a gyártók újabb megoldásait is.

2.3 A hálózati szeletelés teljesítményértékelési szempontjai

Mind a jelen dolgozathoz elkészített megerősítéses tanulásos rendszer, mint a valós szeletelést megvalósító rendszerek esetén kiemelten fontos definiálni az olyan értékeket, amelyek visszajelzéssel szolgálnak a rendszer működésének teljesítményéről. Ahhoz, hogy a hálózat működését optimalizálhassuk, paraméterek halmazával reprezentatív leírást kell adnunk arról. Ehhez az igen speciális témához a Varsói Műszaki Egyetem kutatóinak dolgozatát [18] hívtam segítségül, amely az 5G *hálózati szeleteléssel* kapcsolatos KPI (Key Performance Indicator) adatok témáját dolgozza fel.

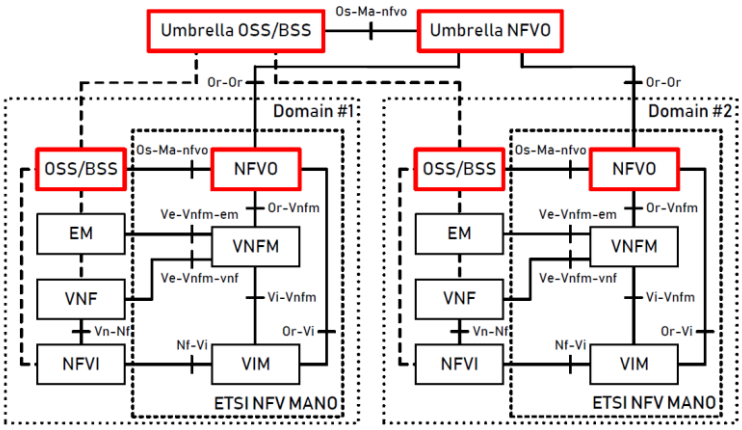
Az end-to-end működést jellemző KPI értékek igen gyakran szoros összefüggésben állnak a QoE (Quality of Experience) elvárások teljesülésével [19]. Ennek a megközelítésnek a nehézsége abban rejlik, hogy míg a QoS értékek pontosabban definiáltak, addig a QoE egy magasabb szintű, ügyfélelégedettséget jellemző koncepció. Éppen ezért számos esetben a hálózati szolgáltatók inkább a rendszer működési paramétereit preferálják mérőszámként, mivel ez objektívebb, könnyebben megvalósítható választás. Noha a szabványosítási szervezetek számos KPI-t definiálnak, gyakran az egyes szolgáltatók saját megközelítéseket választanak.

Az 5G rendszerek KPI értékeit jellegük alapján besorolhatjuk elérhetőségi, integritási vagy kihasználtsági csoportba. Az ehhez hasonlóan generikus hálózati teljesítményindikátorok mellett a *network slicing* megköveteli az ellenállósági és biztonsági, az erőforrás-elaszticitási, illetve az alkalmazásspecifikus KPI értékek definiálását is. Mivel az eddigi relatíve statikus architektúrájú hálózatok helyett dinamikusan skálázódó megoldásról beszélünk, kiemelten fontos az erőforrások adaptivitásának vizsgálata is.

A lengyel kutatók három alapvető KPI-ajánlást fogalmaznak meg, amelyek közül először a szeletre vonatkozó run-time KPI kerül bemutatásra. Ennek lényege, hogy az alul- és túlméretezett erőforrásokat hasonlóan súlyos problémaként kezelve, de ezeket csatlakozási, számítási teljesítménybeli és memóriahasználati szempontból vizsgálva állítunk elő szeletenként és hálózati összesítő szinten értékelhető KPI-okat. Mindez viszonylag részletes, objektíven és erőforrás-hatékonyan mérhető betekintést enged a rendszer állapotába, amely alapján arra következtetek, hogy az erőforráskezelés automatizációja ezen a szinten valósulhat meg, bizonyos szubjektív eredetű, statisztikai értelemben objektívvé konvertálható QoE-alapú KPI-ok kiegészítő felhasználásával.

A második mérőszám-csoport a szelet életciklusára vonatkozik, amely az adott *slice* létrehozásával és újrakonfigurálásával kapcsolatos időbeli értékeket takar. Megkülönböztetnek deployment time-ot, amely a *slice* létrehozására irányuló trigger és a használatra kész szelet felépülése között eltelt idő, valamint az erre vonatkozó skálázhatósági együtthatót, amely ugyanazon szelettípus nagy példányszámú deploymentje esetén szükséges idő alakulását értékeli a *slice*-szám függvényében. Emellett újrakonfigurálási idő bevezetését is szorgalmazzák, mivel a dinamikusan vezérelt, ügyféligény-központú szeletvezérlés esetén ez az elégedettséget komolyan befolyásolhatja. Végül értelemszerűen a szeletek leállási idejét is figyelembe veszik, amely a deploymenthez hasonlóan a trigger és az érintett hálózati szelet teljes megszűnése közti időt jelenti.

A harmadik kategória a multi-domain slicing KPI-ok csoportja, amely az adminisztratív, technológiai és/vagy orkesztrációs értelemben egyszerre több doménen átívelő szeletelést hivatott jellemezni. Az ügyfelet jellemzően nem az érdekli, hogy a szeletek egyes komponensei milyen dinamizmussal írhatók le, sokkal inkább az ezek együttműködésével kialakuló rendszerre tekintenek szolgáltatásként. Emiatt szükséges bevezetni az end-to-end működést jellemző többdoménes KPI értékeket, amelyek két esetre bonthatók. Az egyszerűbb esetben egyetlen orkesztrátorra alapuló rendszert írunk le, amelyek a publikáció tanulsága szerint az egydoménes eset KPI-értékeivel megegyező módon jellemezhetők. A probléma leginkább a többorkesztrátoros rendszerek esetén jelentkezhet, amelyek esetén a helyi OSS/BSS szintjén történő KPI-számításra tesznek javaslatot, amelyet aztán az orkesztrátor elérhetővé tesz a domén szintjén is. Ezekből pedig összegző eljárással kapható a multi-domain KPI-halmaz (4. ábra).



4. ábra: A multi-domain network slicing javasolt KPI-kalkulációs architektúrája

Forrás:[18], Fig. 1

2.4 Network slice, mint szolgáltatás

Mint minden technológiai vívmány, az 5G *hálózati szeletelés* is kizárólag akkor lehet sikeres, ha üzleti szempontból is nyereséget lehet elérni általa. Az utóbbi évek hálózati trendjei lehetővé tették olyan API-ok kifejlesztését, amelyekkel minden eddiginél kevesebb emberi beavatkozással lehetséges skálázható megoldásokat igénybe venni. Mindez az internetszolgáltatók üzleti ügyfeleinek olyan lehetőségeket adhat, amelyeket leginkább ahhoz lehet hasonlítani, mintha az ügyfél saját hálózatot üzemeltetne. Például egy élő közvetítést szolgáltató ügyfél dedikált hálózati kapacitással és policy vezérléssel rendelkezhet, akár dinamikusan változó módon is. Ez pedig olyan erős minőségi és kvantitatív garanciát ad az ügyfeleknek, amelyek a *network slicing* előtti időkben nem voltak lehetségesek.

Ahogyan az a [15] jelű cikkből is kiderül, koncepciót a NaaS, azaz Network-as-a-Service kifejezéssel illeti a szakirodalom. Ugyancsak az Ericsson [17] forrására támaszkodva betekintést nyerhetünk a NaaS világába, amely üzletileg igen sikeres megközelítésnek ígérkezik, ugyanakkor újszerű kihívások garmadájával növeli a fejlesztési folyamatok komplexitását. Amikor ez a szolgáltatás elterjed, és az ügyfelek kezébe szokatlanul nagy kontroll kerül a hálózat paramétereinek menedzsmentje terén, stabil keretrendszerre van szükség, hogy egymás adatfolyamát semmiképp se zavarhassák meg. Emellett a szeletek száma is ugrásszerű növekedésnek indulhat, amely komplex orkesztrációs kihívásokat teremt a hálózati szolgáltató oldalán. Emellett a számlázási ciklusok és a számlázás egységeinek újradefiniálása is szükségessé válhat, mint ahogyan az a másodpercnél is nagyobb felbontású erőforrás-igénybevételen alapuló számlázás esetén a felhőalapú szolgáltatások világában is tetten érhető.

A nehézségek ellenére az üzleti szereplők jellemzően érdemesnek találják a NaaS-fejlesztésekbe történő befektetést, ugyanis többmilliárd dolláros nyereségre számíthatunk ezen egyetlen koncepció által is. Fontos ugyanakkor, hogy a *network slicing* nem egy hirtelen bevezethető üzleti és technológiai megközelítés, sokkal inkább egy útként kell tekintenünk rá. Ahogy a cikk is említi, érdemes lépésről lépésre haladni, és a hálózati infrastruktúra gyártóitól az internetszolgáltatókon át az ügyfelekig ívelő kooperáció keretein belül gondozni a NaaS szolgáltatások projektjeit.

2.5 Erőforrás-optimalizáció az edge computing világában

Ahhoz, hogy megérthessük, miért is lehet igazán kifizetődő megerősítéses tanulás betanított ágensére bízni a *hálózati szeletelés* erőforrásvezérlését, minimális betekintést kell nyernünk az edge computing világába, azon belül is leginkább a Network edge kategóriába. Ehhez az Ericsson legfrissebb, 2023-as összefoglalóját [20] hívom segítségül. A hálózat felhasználókhoz közel eső „peremére" helyezett számítási kapacitás előnyei alapvetően három kategóriába sorolhatók, melyek a gyártás, a videojáték és szórakoztató tartalmak szolgáltatása, valamint az egészségügyi szolgáltatások. Mindez erős hasonlóságot mutat a *network slicing* motivációjául szolgáló témakörökkel.

Az összefoglaló tanulsága alapján az edge három részre osztható: gateway edge, private edge és network edge. A gateway edge az ügyfél által kezelt terület, amely relatíve kis teljesítményű számítási erőforrásokat takar a telephelyen, vagy akár a mobil entitásokon belül. Utóbbiak lehetnek intelligens járművek is, amelyek földrajzi helyzetüket gyorsan és változó irányban módosítják, így a hálózatban elfoglalt logikai helyzetük nem feltétlenül következik fizikai helyzetükből. A privát edge jellemzően jóval nagyobb kapacitású erőforráshalmaz, de szintén az ügyfél által kezelt terület sajátossága. Ezzel szemben a network edge [21] a szolgáltató oldalán helyezkedik el, de a hálózatban a felhasználókhoz minél közelebb. Ez a terület a leginkább releváns a szeletelés erőforrásvezérlése szempontjából.

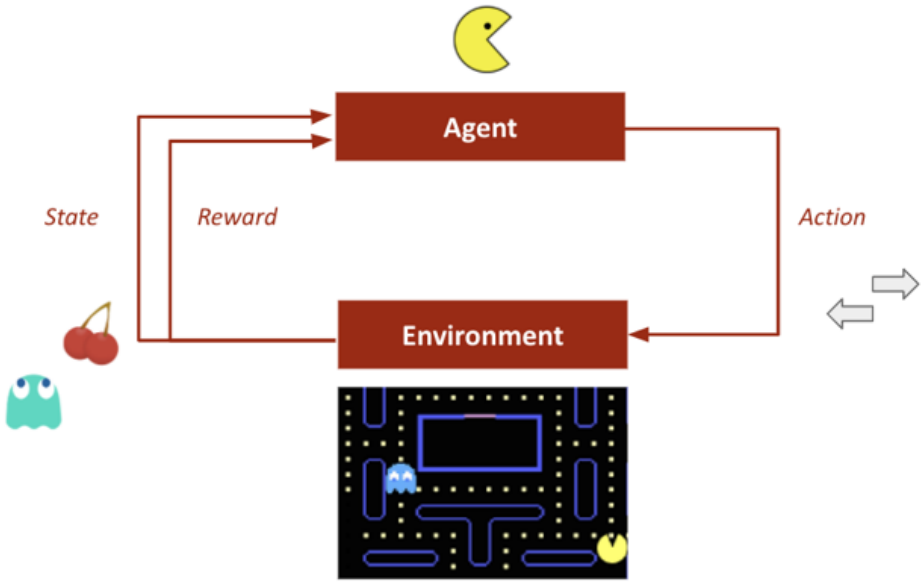
A network edge lényegében a meglévő központosított hálózati erőforrások kiegészítéseként értelmezhető, amely közelebb hozza az ügyfelekhez az adatfeldolgozást. Az Ericsson összefoglalója szerint az automatizált hálózati infrastruktúra megköveteli a sztenderd integrációt, amely több gyártó eszközeivel általában nehézséget jelent. Az orkesztráció terén ugyanakkor kiemelik, hogy erősen korlátozott erőforrásokkal rendelkező Network edge-megoldások kínálkoznak, amelyből következően nagy gondot kell fordítani arra, hogy mire használjuk az itt működő számítási kapacitást.

Az összefoglaló ajánlása megegyezik az általam választott koncepcióval, miszerint AI megoldást kell találni az erőforrás-allokáció menedzsmentjére. Ehhez ugyanakkor nem vehetünk igénybe kellően nagy számítási kapacitást a network edge-ben, mivel ez egyébként is terhelt, korlátozott infrastruktúra. Emiatt válik igazán érdekessé a nagy számítási kapacitást igénylő optimalizáció előzetes betanításra történő használata, és a betanult ágens kis erőforrásigényű, valós idejű beavatkozásainak előnye.

3 A megerősítéses tanulás koncepciója

Amint az a [2] jelű cikkből is kiderül, az AI megoldások, ezen belül is a megerősítéses tanulás egyik fő célja a teljesen autonóm módon működő ágensek előállítása, amelyek a környezetüket aktívan alakítva sajátítják el az optimális működést heurisztikus módon kísérletezve, majd a hibákból gyorsan és hatékonyan tanulva. Mindez sokkal kézzelfoghatóbb és körülhatárolhatóbb eredménnyel kecsegtet, mint a közvéleményt évek óta lázban tartó mesterséges általános intelligencia, amely többnyire szubjektív követelményeknek próbál megfelelni.

A RL-modell lényegét szemlélteti az 5. ábra. A koncepció lelke egy ágens, amelyet a kontextusban kissé szokatlanul hangzó ügynök kifejezéssel is illethetünk. A folyamat diszkrét lépésekre oszlik, amelyek mindegyikében az ügynök ismeri a rendszer állapotát, és a jutalom értékét, amelyet az adott állapotba jutással érdemelt ki. Ennek függvényében akcióval befolyásolja a környezetet, amelynek hatására az új állapotba kerül, és új jutalomértéket ad az ágensnek. A jutalomérték megválasztása az egyik legnagyobb RL-kihívás, mivel ennek az értéknek kellően reprezentálnia kell a környezet állapotának és az ágens működésének optimumközelségét. Az állapottér bejárása kezdetén az ügynök kevésbé veszi figyelembe a jutalmat, ugyanakkor a kísérletezés fázisait követően a tapasztalatokra építkezve a jutalom értékét maximalizálni igyekszik.



5. ábra: A megerősítéses tanulás funkcionális modellje

Forrás: https://miro.medium.com/max/1100/0*hq337YNm2dj2FgFL.webp

4 Megerősítéses tanulás alkalmazása hálózati szeletelés optimalizálására

A dolgozat fő célkitűzése, hogy megerősítéses tanulás (reinforcement learning, röviden RL) [2] alkalmazásával dinamikus erőforrás-menedzselést valósítson meg. A hálózati szeletekhez társított erőforrásokat egy ügynök vezérli, amely a hálózat pillanatnyi állapota és a felhasználói igények alapján törekszik az optimális döntések végrehajtására. A valós időben történő beavatkozás képessége a RL-ágensek azon tulajdonsága, amely igazán sikeressé teszi a koncepciót ipari és hálózatmenedzsmenttel kapcsolatos feladatok ellátása terén.

Mivel valós hálózatokban, többezer felhasználóval a gyakorlatban képtelenség volna kísérletet végezni, triviális következtetésként adódott, hogy szimulátorra van szükség. A hálózati szeletelés szimulációjára számos szoftver áll rendelkezésre, amelyek közül a szakdolgozatomban [1] a [22] jelű cikk szerzői által is használt, blokkolásalapú és észszerűen egyszerűsített modellt alkalmazó szimulátort, a SliceSimet választottam. Mivel ez kimondottan az egyes szeletek allokált bitrátáival dolgozik, kézenfekvő opciót jelentett igen hasonló célkitűzéseimhez. Ezt megfelelő javításokkal és átalakításokkal egy megerősítéses tanulási algoritmus környezetévé alakítottam, majd blokkoláson alapuló, azt minimalizálni igyekvő jutalomfüggvénnyel használtam fel.

A legfőbb probléma ezzel a modellel az volt, hogy a SliceSim [22] névre hallgató program nem biztosított kellő konzisztenciát, és a szeletek súlyozásának vezérlése bemenetként nem mutatott megfelelően erős korrelációt a kimenetként értelmezett, blokkolási rátától függő jutalomértékkel. Emiatt a tanítás sem volt zökkenőmentes, és az ágens döntései csupán lokális optimumállapotokat értek el. Ez jellemzően az egyik hálózati szelet kapacitásának végletekig történő minimalizálását, és a két további slice értékeinek kiegyenlítődését eredményezte, igen távol sodorva ezzel a szimulált hálózat működését a valós 5G rendszerek elvárt funkcionalitásától.

További problémaként jelentkezett, hogy nem állt rendelkezésre referenciaként tekinthető működés, mivel a SliceSim cikkhez felhasznált verziója nem állt rendelkezésre. Emellett az érintett repozitórium utolsó módosításából ítélve a projekt jó ideje nem állt fejlesztés alatt, és a személyes ismeretség, vagy szervezeti összeköttetés hiánya megnehezítette volna a kiegészítő funkcionalitás sikeres igénylését. Így tehát a szakdolgozatomban még a rendszer lelkét alkotó SliceSim lecserélése kínálta a leginkább logikus megoldást a kutatás elmélyítéséhez. A TMIT tanszék kutatóival, Abdulhalim Fayaddal és Dr. Cinkler Tiborral való együttműködés eredményeként lehetőségem nyílt az általuk fejlesztett statikus 5G hálózati szeletelési szimulátor használatára, amely átláthatósága mellett a fejlesztő állandó támogatását élvezi. Ez a szimulátor képes a kutatás szempontjából releváns paramétereket konzisztensen és megbízhatóan előállítani, nagyban elősegítve ezzel a kitűzött célok elérését.

4.1 Motiváció és célkitűzések

A szimulációs és megerősítéses tanulási rendszerrel szemben támasztott kritériumok közül az egyik legfontosabb az volt, hogy az egyes hálózati szeletek blokkolási statisztikái elfogadható mértékű konzisztenciát mutassanak a rendszer állapotával. Utóbbit a szeletekhez allokált kapacitások alkotják, 100%-ra normált összegű értékekkel. Ez egy relatíve egyszerű reprezentáció, de mivel a kapacitások allokációja és a kérések blokkolása szempontjából ez hűen leírja a rendszert, az ágens tanításának hatékonyságát további elemek állapottérbe vonásával csökkentenénk. Ha az egy későbbi fejezetben taglalt jutalomfüggvényt helyesen határozzuk meg, akkor a konzisztencia a tanításba is átvihető, és valódi optimalizáció valósulhat meg.

Hasonlóan fontos elvárás, hogy a szimuláció képes legyen blokkolási statisztikák mellett optimalizációs eljárásra is, és annak eredménye felhasználható legyen a jutalom kiszámításához. Mindez persze nagyban megnövelheti a szükséges számítási kapacitást, de cserébe az ágens kis erőforrásigényű működése közelítheti az összetett számításssorozatok által meghatározott optimumot. Ez a tulajdonság jelenti a dolgozat egyik legfőbb motivációját, hiszen az eddigi fejezetekben taglalt módon erősen korlátozott erőforrásokkal rendelkező network edge infrastruktúra egy ágenssel könnyűszerrel elbírna, de egy komplex és időigényes optimalizációs eljárással aligha. A [23] jelű kutatás is hasonló motivációval történt, ráadásul ugyancsak matematikai optimumkeresés alapján tanított megerősítésestanulás-ágenst alkalmaz. Ebből fakadóan a koncepció létjogosultsága és lehetőségei igazoltnak tekinthetők, így érdemes jelen problémára is alkalmazni azt.

Célkitűzésként tehát tekinthető egy olyan rendszer létrehozása, amely az 5G hálózati szeletelést oly módon szimulálja, hogy az az allokált kapacitásokkal konzisztens blokkolási arányokat eredményezzen. A következő cél az, hogy ebből az adatállományból megerősítéses tanulás menjen végbe, amelynek alapját a blokkoláson alapuló optimumszámítás eredménye jelenti. Az eredményekre vonatkozó célkitűzés része olyan adatvizualizáció megalkotása, amely betekintést nyújt a tanítási folyamatba, illetve a blokkolás és a jutalom értékének lépések függvényében történő alakulásába. Az így kapott információ feldolgozása pedig a kiértékelésből áll, amely során a megközelítés hatékonyságára, a kapott rendszer optimalitására, illetve a mindezt eredményező folyamat főbb jellemzőire vonatkozó kijelentések fogalmazódnak meg.

4.2 A rendszer architektúrája

A szimulációs rendszer logikailag három részre osztható, amelyek közül az első kettő tartozik a szimuláció feladatkörébe, a harmadik pedig maga a megerősítéses tanulás kódegyüttese. Elsőként a felhasználók térbeli elhelyezése szükséges, amelyeket user equipmentként tekintve futtatható az 5G szimuláció. Ennek a folyamatnak inputként a felhasználók hozzávetőleges száma szolgálhat, de célkitűzésként tekinthető a pszeudo-random működés, elkerülendő a hajszálpontosan ismétlődő mintázatok megjelenését, amely a tanítási folyamatot rossz irányba viheti. Az általam megvalósított rendszer architektúrája a 6. ábra által értelmezhető, amelynek komponenseit röviden ismertetem.

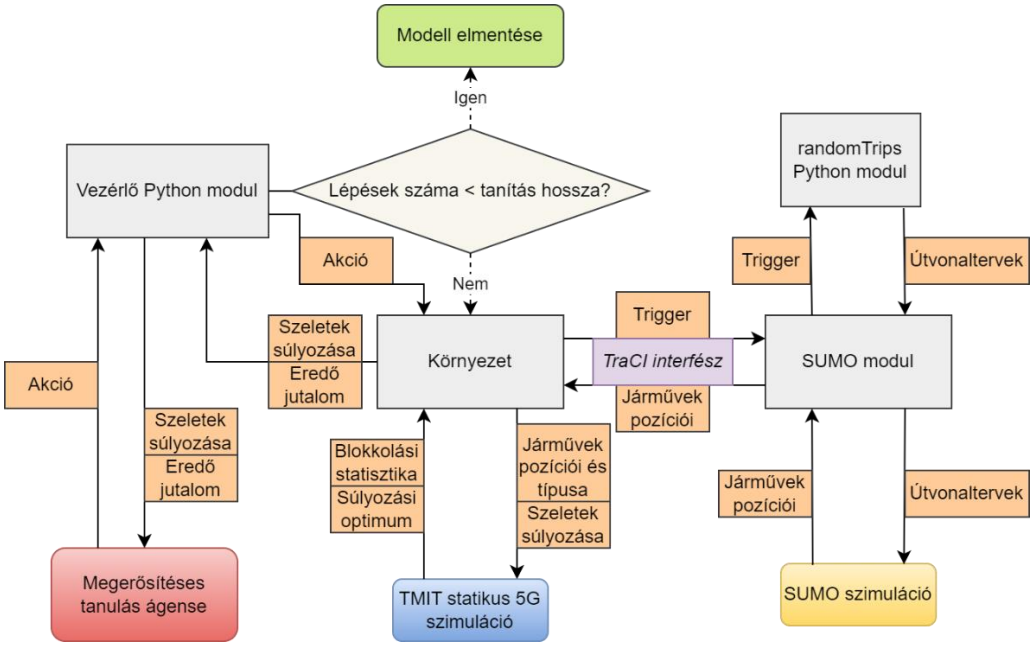
A célnak leginkább a SUMO [24], azaz Simulation of Urban MObility szoftver felel meg, amely megadott paraméterekkel képes járműforgalmat generálni. Fontos megjegyezni, hogy a mobilhálózatok felhasználói nem kizárólag járműberendezések, vagy járművel utazó személyek eszközei lehetnek, hanem gyalogosok és stacionárius berendezések is. Emiatt a SUMO rendszerben gyalogosok útvonalai is szimuláció tárgyát képezik, valamint a térképet elhagyó járművek parkoló helyzetbe kerülnek. A pszeudo-random működés a [25] jelű összefoglalóban is ismertetett randomTrips.py modul felhasználásával valósul meg.

Miután a véletlenszerű útvonalak elkészülnek különféle járműtípusokra egyenként, lefut a szimuláció, amely jelen dolgozat esetén egy 100 másodpercnek megfeleltethető, 100 lépéses folyamat. Az egyes felhasználók pozíciói egy 1000x1000 méretű kétdimenziós koordináta-rendszerre értelmezve időlépésenként egy-egy csv formátumú fájlba kerülnek, amelyet az 5G szimulátor beolvas. Ezen a ponton elérkeztünk a rendszer kutatási szempontból egyik leglényegesebb eleméhez, az 5G hálózati szeletelés szimulációjához.

Az 5G szimulátor, amely jelen rendszer részét képezi, Abdulhalim Fayad (doktorandusz, Budapesti Műszaki és Gazdaságtudományi Egyetem - Villamosmérnöki és Informatikai Kar, Távközlési és Médiainformatikai Tanszék) munkája, és az általa jegyzett [26] jelű dolgozatban is felhasználásra kerül. A szoftver koncepcionálisan abban különbözik a török kutatásban [22] felhasznált SliceSim-től, hogy egy futtatása során egyetlen időpillanatra optimalizálja a szeletek súlyozását, ugyancsak a blokkolás alapján. Emiatt a programot minden egyes ágenstanítási lépés során 100 alkalommal szükséges futtatni, ily módon minden időpillanatra előállítani az optimumot.

Annak érdekében, hogy a SUMO és a TMIT által fejlesztett szimulátor felhasználható legyen a megerősítéses tanulás számára, a szimulátort be kellett illeszteni Gym struktúrába, amely a környezet szabványos leírását valósítja meg. Ennek elengedhetetlen feltétele, hogy a környezet leírásához szükséges adatokat ki lehessen nyerni a szimulátorokból, így például a jutalomfüggvény vagy épp az állapot-lekérdezés metódusát is implementálni kellett.

A szimulációs lépések futását követően a mély megerősítéses tanulásé a főszerep. Az eljárás lényege, hogy a környezet állapota, és a kapott jutalom értéke alapján az ágens döntést hoz a következő lépésről, és elküldi döntését a környezetnek. Ez a folyamat iteratívan ismétlődik a beállított epizódhossz eléréséig, amikor is alaphelyzetbe áll a rendszer, és új epizód kezdődik. Mindez a beállított tanításhossz eléréséig ismétlődik, amely elérése esetén az ágens modellje elmentésre kerül, ahogyan azt a 6. ábra is szemlélteti. Az így kapott, megközelítőleg 50-100 kB tárhelyet foglaló modell tartalmazza a *mély* tulajdonságot adó többrétegű neurális háló súlyozását, valamint az ágens alkalmazása során végzett döntéshozatalhoz szükséges egyéb adatállományokat is. Ez a modell tehát az ágens tanulása során gyűjtött tapasztalataiként tekinthető.



6. ábra: A megvalósított megerősítéses tanulás alapú keretrendszer architektúrájának áttekintő folyamatábrája

A következő, és egyben utolsó lépés az elmentett modell betöltése, majd az ágens működésének megfigyelése ugyanazon környezetben. Ekkor az elvárást a jutalomérték maximalizálása jelenti. Ha a jutalmat az adott célnak megfelelően állítottuk be, akkor egyéb paraméterek optimalizációját is megfigyelhetjük, amely jelen esetben leginkább a blokkolási ráta minimalizációját jelenti, az egyes szeletek kritikusságának súlyozását figyelembevéve. Utóbbi paramétereket 15, 10 és 75 értékűnek határoztam meg, azonos sorrendben az eMBB, mMTC és URLLC szeletekre. Ezek az értékek természetesen csak közelítik az egyes szeletek kritikusságának mértékét, amelynek meghatározása önálló kutatás tárgyát is képezhetné, azonban a rendszer lehetőségeinek vizsgálatára véleményem szerint ezen értékek is megfelelnek.

4.3 Az optimalizáció háttere

A felhasznált szeletelés-szimulátor optimalizációjának alapját az IBM CPLEX [27] adja, amely a megadott matematikai elvárások alapján ad megoldást a problémára. Ez a viszonylag komplex szoftvercsomag elfogadható számítási kapacitásigénnyel rendelkezik, és a gyakorlatban is jól használhatónak bizonyult, például a [26] jelű kutatás során is. A termék már 1988 óta kereskedelmi forgalomban van, de az IBM vállalat folyamatos fejlesztései nyomán korunk drasztikusan megnövekedett számítási kapacitású számítógépeinek lehetőségeit is kellően kihasználja.

Az optimalizáció matematikai szempontból az úgynevezett hátizsák-problémával [28] (angol nyelven elterjedt elnevezéssel knapsack problem) modellezhető. A maximalizáció témakörébe sorolható probléma lényege, hogy egy adott kapacitású „zsákba” adott halmazból kell elemeket helyezni, ahol minden elem súlya és értéke adott. A cél az, hogy a kiválasztott elemeket összesítve a lehető legnagyobb összértéket kapjuk, méghozzá anélkül, hogy a „zsák” kapacitását túllépnénk az elemek súlyával. Esetünkben természetesen nem zsákról, hanem a bázisállomások átbocsátó képességéről beszélünk megszorításként, az értéket pedig az allokált felhasználók száma jelenti. Mindez természetesen fontossági sorrendbe kerül az optimumtól való eltérés slice-kritikussággal való súlyozása által, de ezt a jutalomról szóló fejezetben fejtem ki részletesebben.

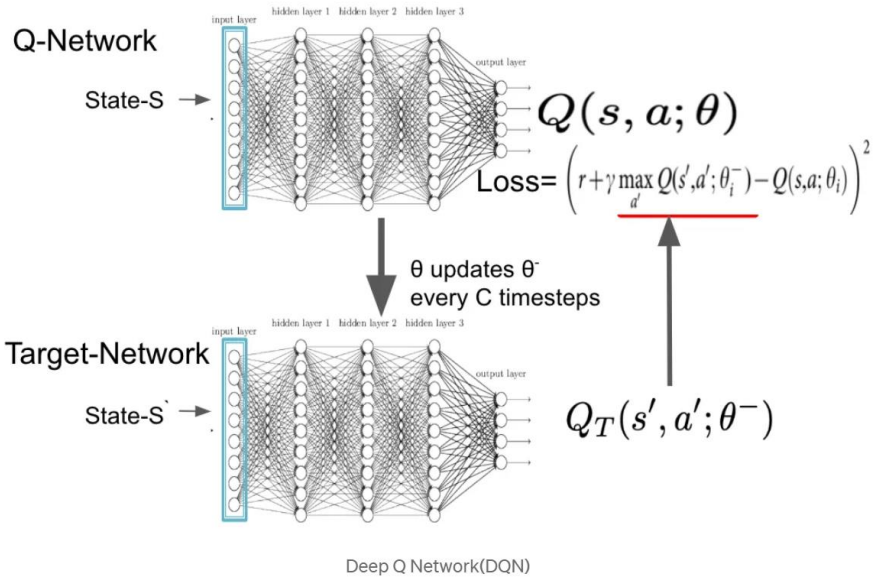
A knapsack-probléma három csoportra oszlik. Ezek közül az egyik a töredékes hátizsák-probléma, amely során nemcsak a kiválasztott elemeket határozhatjuk meg szabadon, hanem azokat darabolhatjuk is. Hasonló megközelítés az SLA-ban foglaltak szerint megvalósítható lenne mobilhálózatok erőforrásvezérlése terén is, de jelen dolgozat nem ezt az opciót alkalmazza. Emellett beszélhetünk korlátos verzióról is, amely során maximalizáljuk a kiválasztható elemek számát. Véleményem szerint ez a legkevésbé releváns jelen felhasználási cél szemszögéből.

Végül, de korántsem utolsó sorban a 0/1 hátizsák-probléma is a témakörbe tartozik. Ekkor nem darabolhatjuk az elemeket, de szabadon választhatunk közülük. Ha egy elemet kiválasztunk, azt a „zsákba” tesszük, ha pedig nem, akkor részben sem adódik hozzá a „zsák” elemeinek értékéhez. Ez a modell írja le legpontosabban a célkitűzéssel kapcsolatos kihívást. A legfőbb nehézség mindezzel kapcsolatban az, hogy NP-nehéz feladatot jelent, amelynek nehézségét a jelű cikk is megfelelően szemlélteti. Ahogy a szerzők is teszik, ezekre a problémákra általában közelítő algoritmusokat alkalmazunk.

4.4 Felhasznált RL-szoftverkönyvtárak és eljárások

Mivel jelen kutatásnak nem célja a megerősítéses tanulás keretrendszerének önálló kidolgozása, és sokkal inkább felhasználni érdemes ezt, logikus opciót jelent a lényegében iparági sztenderdnek számító OpenAI Gym [30] rendszer. Noha a szoftver fejlesztését az Elon Musk, Sam Altman és 4 kollégájuk által 2015-ben alapított OpenAI vállalattól átvette a Farama Foundation, és a termék neve Gymnasiumra módosult, ebben a dolgozatban a még az előbbi vállalat által fejlesztett verziót használtam. Ez a rendszer lényegében egy számos referencia-környezetet tartalmazó, API-alapú megoldás, amelyben többféle megerősítéses tanulási algoritmust próbálhatunk ki. Ezesetben a környezetet maga a szimulációs rendszer alkotja, az algoritmus pedig az eddigi tapasztalataim alapján leginkább hatékony és alkalmas DQN (Deep Q- Network) [31].

A DQN algoritmus a Q-learning [32] alapjaira épül, de a deep szóból következtethető módon mély tanulási eljárás, azaz többrétegű, „mély” neurális hálót alkalmaz Q-tábla helyett. Az eljárás lényege, hogy a hagyományos Q-learninghez hasonlóan kiszámított Q-értékeket neurális háló bemeneteként kezeljük, majd ezt adott számú lépésenként adjuk át a célhálózatnak (7. ábra). Ezzel elkerülhető a rövidtávú zavarok hatásának túlzott befolyása az ágens tanulási folyamatára, amely különösen hasznos egy hasonlóan összetett rendszerben.



7. ábra: a DQN algoritmus neurálisháló-szerkezete

Forrás: Renu Khandelwal,

https://arshren.medium.com/deep-q-learning-a-deep-reinforcement-learning-algorithm-f1366cf1b53d

A Gym eszközt nem közvetlenül használtam fel, hanem segítségül hívtam a Stable-baselines3 [33] keretrendszert. Utóbbi lényege, hogy nyílt forráskódú megoldást nyújtson a mély megerősítéses tanulásos algoritmusok terén, Python nyelven. A programozási nyelv azért is fontos, mert az általam írt kód ugyancsak Python nyelvű, ezáltal meglehetősen könnyen együttműködésre bírható a Stable-baselines3 kódjával is. Ez az implementáció a környezet változtatása nélkül nyújt lehetőséget többféle RL-algoritmus kipróbálására, amely igencsak tanulságos lehet, ha a cél az optimumot leginkább és leggyorsabban közelítő eljárás kiválasztása. Léteznek ennél jóval komplexebb szoftverek is, de egyfelől a kutatás jelen szakaszában véleményem szerint nem indokolt további beállításokkal kísérletezni, másrészt pedig, ha mégis változtatni volna szükséges, a sztenderd Gym-környezetként implementált szimulátor kifejezetten könnyen, lényegi módosítás nélkül felhasználható összetettebb rendszerekben is.

4.5 A megerősítéses tanulás jutalomfüggvénye és állapottere

Az eddigi fejezetekben taglalt rendszer működésének optimalitása még nem elegendő a sikeres ágenstanításhoz, ugyanis a környezet állapotának optimumát csak úgy találhatjuk meg, ha megfelelő matematikai reprezentációt alkotunk ehhez. Egy olyan függvény megalkotása a cél, amely annál nagyobb értéket vesz fel, minél kedvezőbb a rendszer állapota. Noha a szakirodalom jutalomként (reward) említi ezt az értéket, valójában a szóhoz kapcsolt pozitív érzelmek nem tükrözik a működés lényegét. Az ágens a jutalom értékét maximalizálni igyekszik, de ahhoz, hogy az optimumtól távol eső állapotok elkerülését megtanulja, nagy abszolútértékű negatív jutalmat kell kapnia. A jutalom tehát köznapi kifejezéssel élve büntetés is lehet, így érdemesebb matematikai reprezentációt értelmezni.

Az általam fejlesztett környezet optimumát akkor éri el, ha a lehető legkevesebb kérés kerül blokkolásra, tehát a legtöbb felhasználói eszközt kapcsoljuk a hálózatba. Ekkor tehát a blokkolási valószínűség alacsony. Mindezzel ugyanakkor nem elégszem meg, hiszen késleltetési követelmények éppúgy jelen vannak a hálózati szeletek vezérlése során, mint bitrátára vonatkozó elvárások. Ezt úgy reprezentálom, hogy az egyes szeletekhez kritikussági súlyozásokat rendelek, ahol a nagyobb érték a kritikusabb szeletet jelenti. Ez megközelítőleg fordítottan arányos a késleltetésre vonatkozó toleranciával. Jelen esetben nem volt célom a prioritások pontos arányainak reprezentációja, de mivel az URLLC slice a leginkább kritikus, így 75-ös értéket kapott, amely jóval nagyobb az eMBB 15 és az mMTC 10 értékénél.

A szimuláció során egy ágenstanítási lépés 100 jármű- és hálózatszimulációs időlépésnek felel meg, mivel az ágens 100 másodperces szimulációs fázisokat tekint egy megfigyelési és beavatkozási ciklusként. Ezt a későbbiekben szükségessé válhat megváltoztatni, ugyanakkor ilyen módon egy napi forgalom, vagy egyéb időbeli periodicitás is szimulálható. Ugyanakkor az ágens döntéshozatalához igen kis számítási kapacitásra van szükség, ezért a jövőben akár egyetlen másodperces ciklusonként történő beavatkozást is elképzelhetőnek tartok. A továbbfejlesztési lehetőségekre vonatkozó fejezetben erre részletesebben is kitérek.

A választás mindezen kritériumok alapján egyetlen jármű- és hálózatszimulációs időlépésre vonatkozóan az alábbi összefüggésre esett:

$$R(t) = \sum_{s} -W(s) * |O(s,t) * A(s,t)|$$

ahol:

- *t*: az adott időlépés
- *s*: az adott hálózati szelet
- *R*(*t*): a jutalom értéke az adott időlépésre vonatkoztatva
- *W*(*s*): az adott szelet kritikussági súlyozása
- *O*(*s*,*t*): az optimális erőforrás-allokáció az adott időlépésre és szeletre vonatkoztatva
- *A*(*s*,*t*): a tényleges, ágens által elrendelt erőforrás-allokáció az adott időlépésre és szeletre vonatkoztatva

Az összefüggés logikai alapja, hogy annál nagyobb mértékben csökkenjen a jutalom értéke, minél nagyobb mértékben tér el a valódi, ágens által választott szeletkapacitás-allokáció az optimálistól. Emellett a kritikusság alapján történő súlyozást is tetten érhetjük a W(s) súlyok formájában. Ez az összefüggés rábírhatja az ágenst arra, hogy tanulási folyamata során az olyan állapotokat találja kívánatosnak, amelyeket a hátizsák-probléma megoldásai indikálnak. Fontos azonban megjegyezni, hogy mivel az egyes szeletek felhasználóinak száma nem tükrözi a valós állapotokat, a slice-ok allokációs aránya nem feltétlenül reprezentatív valós hálózatokra vetítve. Emellett a szeletek túlzott erőforráskiosztását is büntetéssel sújtjuk, amely energiagazdálkodási okokból értelmes stratégia lehet, de valószínűleg egy negyedik „unallocated” kategóriával is érdemes lenne bővíteni a szeletek halmazát, és megállapításokat tenni a kísérlet hatékonyságára vonatkozóan, összehasonlítva azt jelen dolgozat eredményeivel.

Az állapottér a [5] szakdolgozatban viszonylag összetett, kilencelemű adatállomány volt, amely a szabad kapacitást és a felhasználók szeletenként értelmezett számát is magában hordozta. Mindez nem hozta meg a kellő hatékonyságot és optimumkeresést, ezért a kutatás ezen szakaszában egyszerűsítettem az állapotteret. A jelenlegi állapottér háromelemű, és a szeletek erőforrás-allokációit tartalmazza. Ez remélhetőleg egyszerűbb tanítási folyamatot, és gyorsabb konvergenciát fog eredményezni, amely eddigi tapasztalataim alapján nagy előnyt jelenthet.

Az akciók, amelyeket az ágens elvégezhet a környezetben, a [22] cikk ajánlásai alapján alakultak ki, ahol *c* a lépések felbontását hivatott kontrollálni. A *c* változó értékét 2 százalékban határoztam meg, amelynek optimalitását az állapottér bejárása során igazolhatjuk vissza. Így az akciótér alábbi módon értelmezett:

eMBB változási ráta [%]	**mMTC változási ráta [%]**	**URLLC változási ráta [%]**
0	**0**	**0**
+2*c*	**-*c***	**-*c***
-2*c*	**+*c***	**+*c***
-*c*	**+2*c***	**-*c***
+*c*	**-2*c***	**+*c***
-*c*	**-*c***	**+2*c***
+*c*	**+*c***	**-2*c***

1. táblázat: A hételemű diszkrét akciótér elemei a vonatkozó cikk [22] alapján

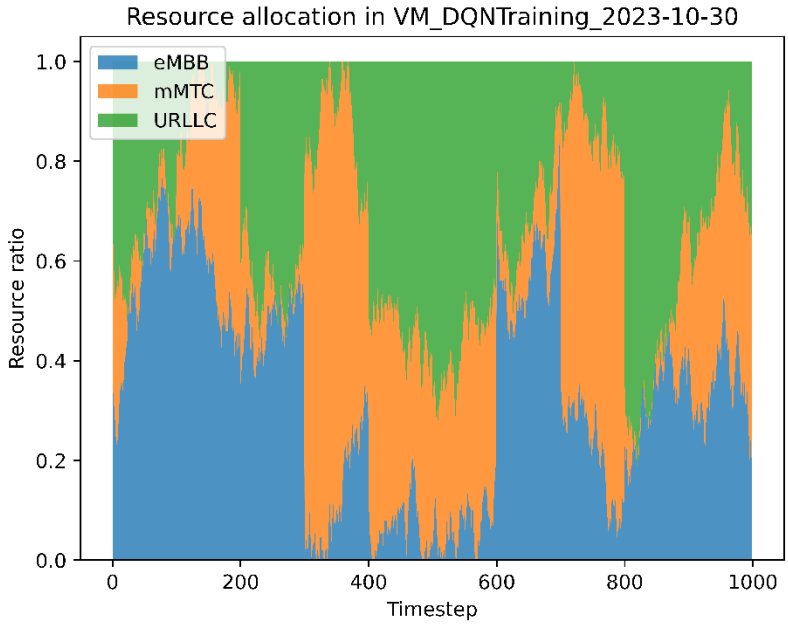
5 Az eredmények értékelése

5.1 A tanítás folyamata

Az ágens tanításának hosszát eddigi tapasztalataim [5] alapján 1000 tanítási lépésben határoztam meg, mivel a szakdolgozatomban használt rendszer jellemzően az 500 lépés feletti tartományban mutatott kellően erős konvergenciát az ágens döntése tekintetében. Ezzel legnagyobb problémát az jelentette, hogy a tanítási folyamat rendkívül időigényes, mivel a szimulációt százezer alkalommal kell futtatni, és ezerszer kiértékelni. A teljes processzus hozzávetőleg 47 óráig tartott, amely során a célra használt viszonylag nagy számítási teljesítményű laptop CPU-kihasználtsága 60 és 100 százalék között ingadozott. Noha ez praktikai szempontból igencsak nehezen megoldható feladatot adott, pozitív hozadéka is van: remekül szemlélteti, hogy miért nem érdemes az NP-nehéz optimalizációs problémákat a kisteljesítményű és jellemzően specializált network edge-beli eszközökön megoldani.

A tanítási fázis során érdemes azt is megfigyelni, hogy az esetleges rövidtávú jutalomérték-kilengések mennyire befolyásolják az ágens tanulását. Amennyibe ez jelentős, átlagolást vezethetünk be, amely a különösen változékony környezetek esetén előnyös lehet. Noha a DQN önmagában is képes a rövid zavarok kiszűrésére a célhálózat akkumulált értékekkel történő feltöltése által, mégis számos kutatás, köztük a [35] jelű cikk tanulsága szerint a környezet adottságaitól függően megfontolandó lehet extra átlagolással javítani a tanulás konzisztenciáját. Továbbá fontos arra vonatkozó következtetéseket levonni, hogy a 100 lépésenként bekövetkező epizódváltáskor történő véletlenszerű slice-arányok állapotából merre mozdul el az ügynök.

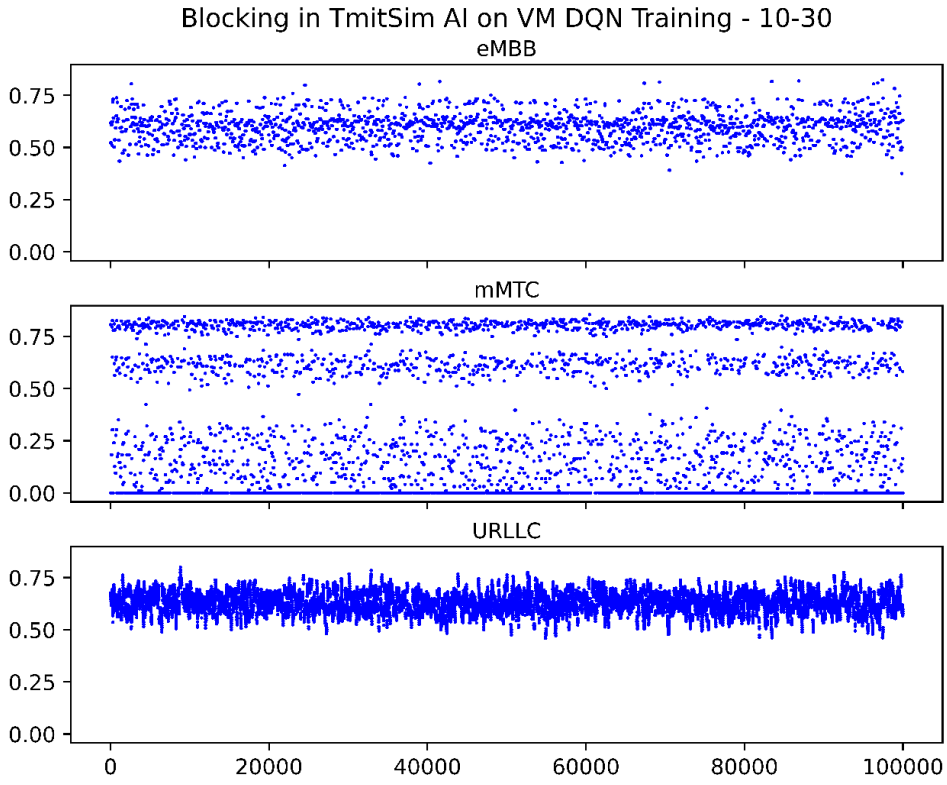
A tanítás értékelése során a legfőbb szempontok egyike, hogy az ágens bejárta-e a teljes állapotteret. Ez diszkrét állapot-értékek esetén véges számú állapotot takar, de a szimulátor (a Python float adattípusának korlátai erejéig) folytonos módon működik. Teljes bejárásról tehát nem beszélhetünk, de értékelnünk kell, hogy a bejárt állapotok megfelelően nagy felbontással lefedik-e az állapotteret. Ebből a célból a Matplotlib [34] kódkönyvtárt alkalmaztam, amely a Pythonban elérhető adatfeldolgozási képességekhez illesztve sztenderd és elterjedt megoldásnak számít. Ezen belül is a stackplot metódust hívtam segítségül, amely egymásra helyezve szemlélteti az egyes értékeket az időlépések alkotta közös abcissza tengelyen (8. ábra). Mivel a három allokációs arányszám értékének összege mindig 1, a grafikon könnyen olvasható, szemléletes betekintést nyújt a tanulási folyamatba.



8. ábra: az ágenstanítás allokációs arányai

A grafikonon jól látható, hogy az ágens az állapotteret kellően alaposan lefedi, mindhárom szelet allokációját minimalizálja és domináns szerepbe is helyezi. A 100 lépésenként történő újrakonfigurálás után is folytatódni látszanak a változási tendenciák, amely a DQN algoritmus jelen implementációjának hatékonyságára utal. A grafikon alapján tehát jó választásnak tűnik a DQN, akárcsak a kiválasztott lépésméretek. Mindemellett különös tendencia, hogy az mMTC szelet súlyozása az ábrán a vetélytársakra jellemzőnél látszólag kisebb területet foglal el.

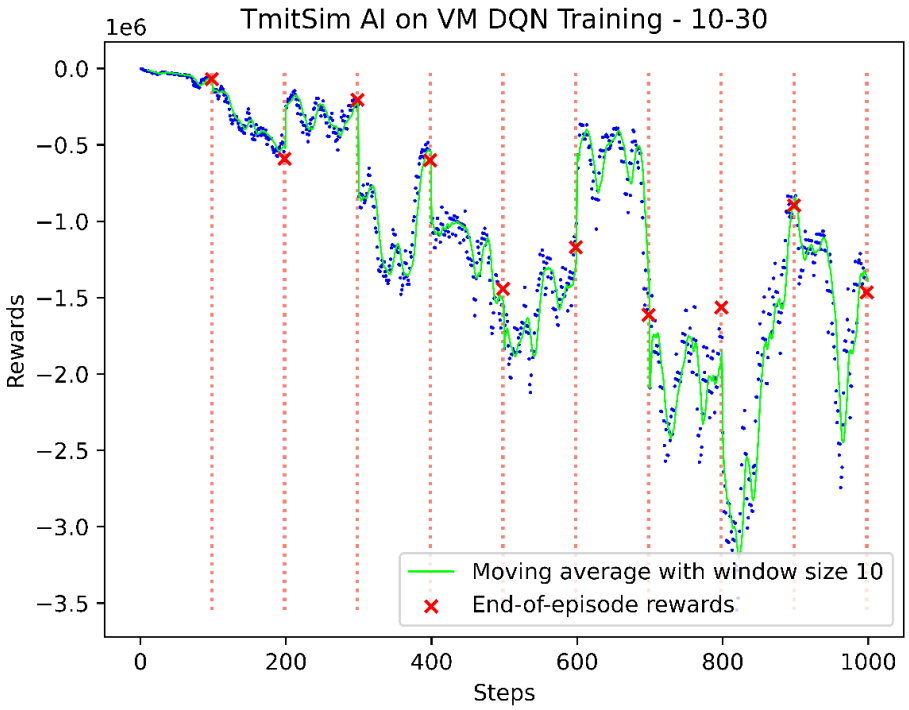
A jutalomfüggvény ugyan áttételesen tartalmazza a blokkolás alapján meghatározott optimumot, mégis szükségesnek tartom a blokkolási arányokat különálló grafikonon ábrázolni, amelyre ugyancsak a Matplotlibet hívtam segítségül. Ezesetben háromosztatú, egymás alatt elhelyezkedő koordináta-rendszereket hoztam létre, hogy a három szelet blokkolási statisztikáinak alakulását párhuzamosan követhessük (9. ábra). Mivel ez nem folytonos jellegű értéksorozatot ad, scatter, azaz ponthalmaz típusú ábrázolást választottam hozzá, a hálózatszimulációs időlépések százezres számosságával.



9. ábra: A blokkolás alakulása a tanítás alatt

Jóllehet, ez a szakasz még csupán a tanulásról szól, a blokkolás ábráját mégis meglepőnek tartom. Látszik, hogy az optimalizáció ráférne a hálózatra, mivel igen sok kérés végződik blokkolással, mindhárom slice esetén. Az eMBB szelet blokkolása nagyjából egyenletesen alakul, bár optimizmusra ad okot az epizódok végéhez közeledve egy-egy alacsonyabb érték. Az mMTC esetén különös tendencia figyelhető meg a blokkolási ráta három, jól elkülöníthető zónára oszlása nyomán, de a 0 érték felülreprezentált. Az URLLC slice valószínűleg kevés felhasználója miatt kisebb súllyal esik latba a jutalomszámításnál, de valószínűleg az erre mutatkozó kis „érdeklődéssel” magyarázható az is, hogy a rendszer állapota nem gyakorolt számottevő hatást az URLLC blokkolási arányára.

A tanítás sikerességének vizsgálata alapvetően a betanított ágens döntésein végezhető el, de a tanítási processzus jutalomfüggésének alakulása szempontjából fontosnak tartom a kumulált jutalomérték alakulásának ábrázolását erre a szakaszra is (10. ábra). Ebből eldönthető, hogy milyen erős a kapcsolat a jutalom és az ágens beavatkozása között. Minél folytonosabb a grafikon, annál következetesebbnek tekinthető a rendszer. Emiatt ezesetben ponthalmaz grafikonon ábrázolom a konkrét értékeket, de emellett folytonosan ábrázolt mozgóablakos átlagértéket is hozzáadok.



10. ábra: A jutalom alakulása az ágens tanulása alatt

A jutalomérték nagy mértékű korrelációra enged következtetni a rendszer állapota és az ágens döntései között, így ez szintén pozitív fejleményként értékelendő. Ugyanez a grafikontípus a szakdolgozatom [5] tanítási fázisára vetítve jóval nagyobb véletlenszerűséget mutatott, így ehhez viszonyítva egyértelmű előrelépést jelent az új 5G-szimulátor és az átdolgozott jutalomfüggvény. Érdekes tendencia, hogy a jutalom a folyamat vége felé haladva jellemzően csökken, ugyanakkor az epizódvégi értékek, amelyeket az ágens a leginkább determinisztikusan kapott, az adott epizódra vonatkozó maximumértékek közelében helyezkednek el. Alapjában véve mindezen okoknál fogva a jutalom alakulását, és ezzel a tanítási fázist sikerként értékelem.

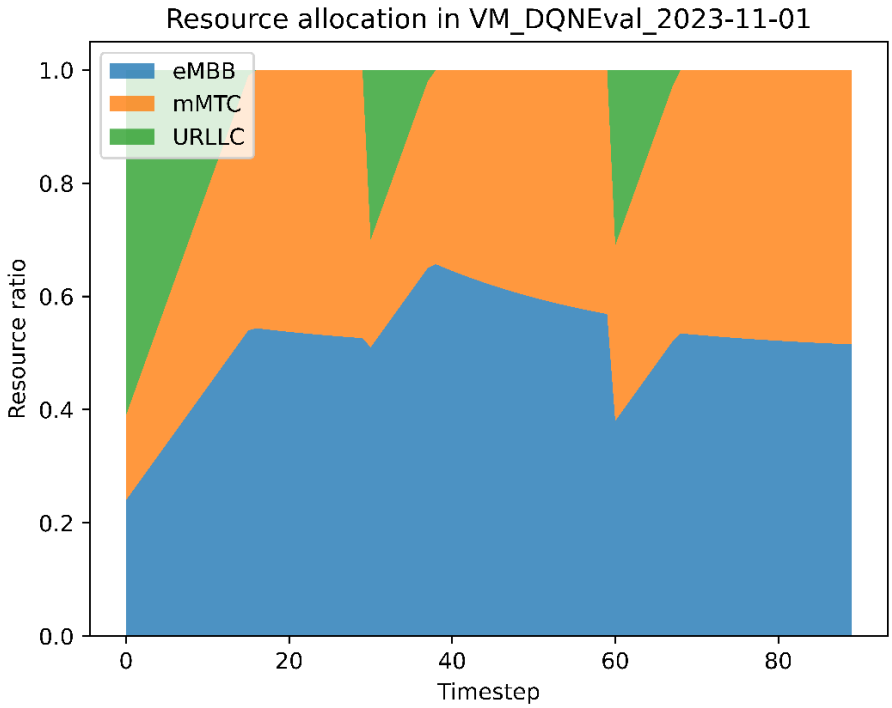
5.2 A betanított ágens viselkedése

A tanulás eredménye leginkább a már betanított ügynök lépésein érhető tetten. Az eredmények kiértékelése során fontos szempont, hogy az ágens mennyire konzisztens, és az általa elért állapot mennyire áll közel a valós optimumhelyzethez. Ebben a szakaszban már nem történik tanulás, és a DQN ügynök neurális hálója is változatlan marad. Amit viszont fontos megfigyelni, az a döntés bemenetfüggőségének mértéke, hiszen, ha az ágens döntései nem elég adaptívak, az arra enged következtetni, hogy valamilyen okból a dinamikus optimum megtalálása nehezen leküzdhető kihívást jelent az ágensnek.

Az OpenAI Gym rendszer lényeges mérőszámokat közöl az ágens kiértékelésének végén, mindezt epizódonként történő bontásban. Az egyik ezek közül a jutalomértékek átlaga, amelyből következtethetünk a rendszer működésének konzisztenciájára. A másik közölt érték a jutalomérték szórása, amely az átlagos értéktől való átlagos eltérést jelenti. Ha ez kellően alacsony, akkor stabil eredményről, nagy mértékű determinisztikus jellegről és kis mértékű véletlenszerűségről beszélhetünk. Az értékek a háromepizódos kiértékelés során az alábbiak voltak:

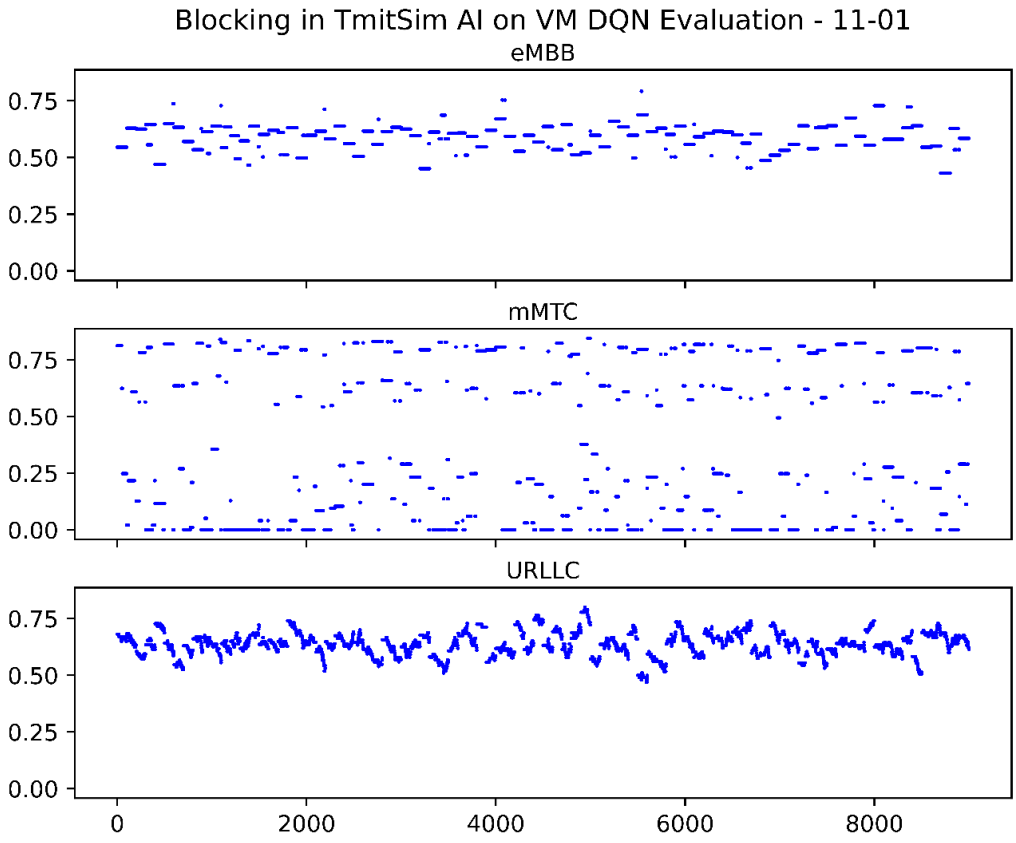
```
DQN agent evaluation complete!
Mean of rewards:  [-1222641.631755, -3602214.152946, -6245235.700217]
Standard deviation of rewards: [30, 30, 30]
```

Noha az értékek átlagának epizódokon átívelő csökkenése aggodalomra ad okot, az ágens konzisztenciáját a stabil 30-as szórásérték nagyban megerősíti. Így tehát az ügynök lépéseit következetesnek ítélem meg, tehát a tanítás elég hosszú volt ahhoz, hogy stabil működés alakuljon ki. A jutalomértékek átlaga ugyanakkor nem ad okot megelégedésre. A tanítási fázisban is alkalmazott adatvizualizációs eljárásokkal további vizsgálatra nyílik lehetőség (11., 12. és 13. ábra).



5. ábra: A betanított ágens hatása az allokációk arányára

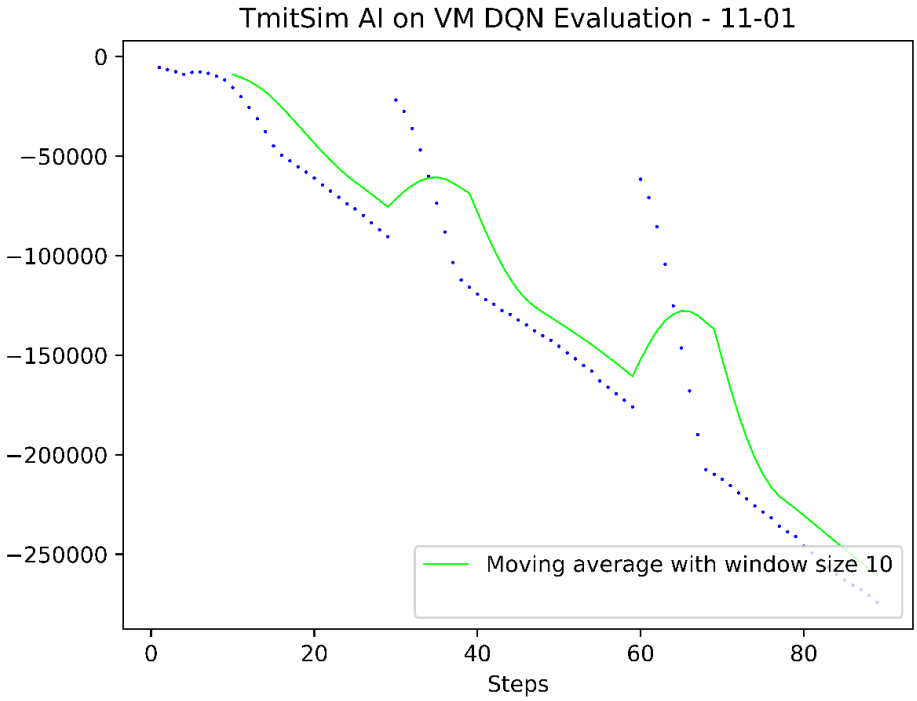
A jutalomértékek szórására alapozott megállapítás, miszerint az ágens konzisztens módon, de kifogásolható optimalitással működik, ezesetben is visszaigazolható. Tisztán látható, hogy mindhárom epizód során a cél az URLLC szelet minimalizációja, míg az eMBB és mMTC slice egyenlő arányban osztozhat a felszabadult erőforrásokon. Ez a szakdolgozatomban [5] tapasztalthoz hasonló probléma, amelynek okára jelen ábra alapján nem derül fény. Mindemellett az első epizód vége 30 időlépésnél történő elérkezése előtt az eMBB szelet allokációjának rovására az URLLC is pozitív kapacitáshoz jut. Összességében tehát egy valós hálózat üzemelésének optimumát a kapott eredmény nem közelíti megfelelően, ugyanakkor a konzisztencia megkérdőjelezhetetlen. A szuboptimalitás oka lehet a felhasználók számának és a kritikusság értékének szeletek közti egyenlőtlen megoszlása, amely értékek meghatározása további kutatómunka tárgyát képezheti.



12. ábra: A betanított ügynök által elért blokkolási értékek

Számomra ezen szakasz egyik legérdekesebb megállapítása végezhető el a blokkolási ábra alapján. Ezesetben a tanulási szakasszal ellentétben nagyon is éles tendencia figyelhető meg az URLLC slice esetén, ahol bizonyos periodicitással, hozzávetőleg 40 és 70 százalék közt alakul a blokkolások aránya. Ennek a ciklikusságnak valószínűleg az 5G szimulátor tanulmányozása hozhatja el a megfelelő magyarázatát. Az eMBB slice blokkolási arányai szintén viszonylag kevéssé függnek az allokációk pillanatnyi állapotától.

Az mMTC szelet esetén ugyancsak három szekcióra oszlik a blokkolási grafikon, de a három slice közül ez a legkedvezőbb eset. Mindez önálló magyarázatra szorul. Amint arra Abdulhalim Fayad, a [26] jelű cikk szerzője is felhívta a figyelmem, az mMTC kliensek rendelkeznek a legkisebb bitráta-elvárással. Emiatt amikor a többi szelet klienseinek „nagyravágyóbb” igényei a kapacitás korlátai miatt nem elégíthetők ki, az mMTC felhasználók értékes sávszélességhez jutva nagyobb számban szerzik meg a nagy kapacitásigényű készülékek sikertelen csatlakozása nyomán felszabaduló erőforrásokat. Ez igen fontos megállapítás, amely további optimalizációs lehetőségeket hordoz magában.



6. ábra: A betanított ágens hatása a jutalomérték alakulására

Végül a jutalomértékek alakulását vettem górcső alá. Ez az ábra az ügynök globális optimum keresésére tett erőfeszítéseinek sikertelenségét szemlélteti. Sajnálatos módon az ágens működése a kezdeti értékek meredek csökkenését eredményezte. Elképzelhetőnek tartom, hogy a jutalomfüggvény alakulásának feltérképezése során lokális optimumállapotokat találva a tanuló ágens ezen opciók felé konvergált, és ez megnehezítette a globális optimum előállítását.

Ezzel szemben arra, hogy a három epizód miért végződött sorrendben egyre alacsonyabb jutalomértékekkel, jelenleg nem találok magyarázatot. Megeshet, hogy egy hiba folytán a jutalomfüggvény az előző lépések jutalmát is bemeneti adatként dolgozza fel. A szándékosság ezesetben kizárható, de Python nyelvű fejlesztői tapasztalataim alapján bizonyos memóriakezelési problémák esetén ugyanazon memóriaterületeket veszi alapul akkor is, amikor az elvárt funkcionalitás ezt nem tartalmazná. A rendszer más pontján is találkoztam ezzel a jelenséggel, de ott a deepcopy beépített metódus segítségével elhárult a probléma. A fejlesztési lehetőségek között ezzel a hibajavítással is számolni kell. A konzisztencia és a konvergencia ugyanakkor ezesetben sem kérdőjeleződött meg.

5.3 Fejlesztési lehetőségek

Az eredmények értékelése során arra a következtetésre jutottam, hogy miután a jutalom következetes csökkenésének esetleges programozási okai kivizsgálásra és elhárításra kerülnek, érdemes a szimuláció szerkezetét átdolgozni. Mivel jelenleg 100 időlépés optimumtól való eltérésének átlagával számol az ágens, az értékek túlzott mértékben átlagolódhatnak, és a dinamikus, valós idejű beavatkozás előnye elveszhet az akkumuláció ködében. Emiatt érdemesnek találom a DQN ügynök tanítási lépéseit a járműforgalom- és hálózati szimulációéval egyenlő ütemezéssel végezni. Ez gyorsabb beavatkozásra, és az eddigi monotonitáshoz viszonyítva jóval heterogénebb, adaptívabb működésre adna lehetőséget az ágensnek.

A fejlesztés kivitelezhetősége szempontjából a szakdolgozatomban [5] is alkalmazott felhőalapú virtuális számítógép, vagy lehetőség szerint ennél jóval nagyobb számítási teljesítmény igénybevétele indokolt. A lokális számítógépen történő tanítási folyamat igencsak lassú, így nem kaphatok kellően gyors visszajelzést a fejlesztési szándékaim sikerességéről. Emellett természetesen a kód optimalizációjával is csökkenteni lehetne a futtatáshoz szükséges időt. A SUMO szimulációt időlegesen célszerű lehetne véletlenszerűen mozgó eszközök koordináta-generálására cserélni, ezzel is csökkentve a fajlagos számítási időt.

Miután az előbbiekben ismertetett problémák megoldásra kerülnek, érdemes lehet realisztikus városi forgalomszimulációt alapul venni a felhasználói eszközök reprezentációja céljából. A jelenlegi térkép ugyan Dublin város egy részletének pontos mása, a forgalom sűrűsége és tendenciái mégsem feleltethetők meg a valós viszonyoknak. Emellett Budapest egyes részletein kísérletezve a valódi bázisállomások pozícióit is használhatná a rendszer, amely közvetlen tanulságokat engedne levonni a térség hálózati szeletelésének lehetőségeivel kapcsolatban. Ezzel párhuzamosan a multi-agent megerősítéses tanulásra [36] váltás lehetőségét is fenntartom, mivel ezzel a bázisállomások egyedi optimumát lehetséges megtalálni.

Alapvetően sikerként tekinthetünk az 5G network slicing megerősítéses tanuláson alapuló erőforrás-optimalizációjára, hiszen az ágens a rendszer megfelelő finomításával realisztikus működésre bírható. A valós adatok felhasználásával eljöhet a realizált 5G-rendszerekben is alkalmazható RL- ügynökök kora, amelyek takarékosan bánnak a számítási erőforrásokkal ezen az egyébként is igen erőforrás-kritikus tématerületen.

Irodalomjegyzék

[1] Dhall, D., Kaur, R., Juneja, M. (2020). Machine Learning: A Review of the Algorithms and Its Applications. In: Singh, P., Kar, A., Singh, Y., Kolekar, M., Tanwar, S. (eds) Proceedings of ICRIC 2019 . Lecture Notes in Electrical Engineering, vol 597. Springer, Cham. https://doi.org/10.1007/978-3-030-29407-6_5

[2] K. Arulkumaran, M. P. Deisenroth, M. Brundage and A. A. Bharath, "Deep Reinforcement Learning: A Brief Survey," in IEEE Signal Processing Magazine, vol. 34, no. 6, pp. 26-38, Nov. 2017, doi: 10.1109/MSP.2017.2743240.

[3] Tammy Jiang, Jaimie L. Gradus, Anthony J. Rosellini, Supervised Machine Learning: A Brief Primer, Behavior Therapy, Volume 51, Issue 5, 2020, Pages 675-687, ISSN 0005-7894, https://doi.org/10.1016/j.beth.2020.05.002.

[4] H. U. Dike, Y. Zhou, K. K. Deveerasetty and Q. Wu, "Unsupervised Learning Based On Artificial Neural Network: A Review," 2018 IEEE International Conference on Cyborg and Bionic Systems (CBS), Shenzhen, China, 2018, pp. 322-327, doi: 10.1109/CBS.2018.8612259.

[5] Bauer László Dániel – „5G hálózati szeletelés optimalizációja megerősítéses tanulás segítségével”, BSc szakdolgozat, Budapesti Műszaki és Gazdaságtudományi Egyetem, Villamosmérnöki és Informatikai Kar, 2022

[6] Intel – Top Use Cases for 5G Technology, https://www.intel.com/content/www/us/en/wireless-network/5g-use-cases-applications.html, 2020

[7] J. Navarro-Ortiz, P. Romero-Diaz, S. Sendra, P. Ameigeiras, J. J. Ramos-Munoz and J. M. Lopez-Soler, "A Survey on 5G Usage Scenarios and Traffic Models," in IEEE Communications Surveys & Tutorials, vol. 22, no. 2, pp. 905-929, 2020

[8] O. O. Erunkulu, A. M. Zungeru, C. K. Lebekwe, M. Mosalaosi and J. M. Chuma, "5G Mobile Communication Applications: A Survey and Comparison of Use Cases," in IEEE Access, vol. 9, pp. 97251-97295, 2021

[9] B. S. Khan, S. Jangsher, A. Ahmed and A. Al-Dweik, "URLLC and eMBB in 5G Industrial IoT: A Survey," in IEEE Open Journal of the Communications Society, vol. 3, pp. 1134-1163, 2022

[10] C. Bockelmann et al., "Towards Massive Connectivity Support for Scalable mMTC Communications in 5G Networks," in IEEE Access, vol. 6, pp. 28969-28992, 2018

[11] H. Chen, L. Yuan and G. Jing, "5G Boosting Smart Cities Development," 2020 2nd International Conference on Artificial Intelligence and Advanced Manufacture (AIAM), Manchester, United Kingdom, 2020, pp. 154-157, doi: 10.1109/AIAM50918.2020.00038.

[12] Z. Li, M. A. Uusitalo, H. Shariatmadari and B. Singh, "5G URLLC: Design Challenges and System Concepts," 2018 15th International Symposium on Wireless Communication Systems (ISWCS), Lisbon, Portugal, 2018

[13] D. Zhang et al., "5G-Enabled Health Systems: Solutions, Challenges and Future Research Trends," 2019 ITU Kaleidoscope: ICT for Health: Networks, Standards and Innovation (ITU K), pp. 1-8, Atlanta, GA, USA, 2019

[14] R. Liu, X. Hai, S. Du, L. Zeng, J. Bai and J. Liu, "Application of 5G network slicing technology in smart grid," 2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE), Nanchang, China, 2021, pp. 740-743, doi: 10.1109/ICBAIE52039.2021.9389979.

[15] Qadir, J., Ahmed, N., Yousaf, F. Z., & Taqweem, A., "Network as a service: the new vista of opportunities," arXiv preprint arXiv:1606.03060, 2016

[16] S. Abdelwahab, B. Hamdaoui, M. Guizani and T. Znati, "Network function virtualization in 5G," in IEEE Communications Magazine, vol. 54, no. 4, pp. 84-91, April 2016, doi: 10.1109/MCOM.2016.7452271.

[17] Ericsson – The essential building blocks of E2E network slicing, https://www.ericsson.com/assets/local/networks-slicing/docs/network-slicing-building-blocks.pdf , 2023

[18] S. Kukliński and L. Tomaszewski, "Key Performance Indicators for 5G network slicing," 2019 IEEE Conference on Network Softwarization (NetSoft), pp. 464-471, Paris, France, 2019

[19] R. D. Mardian, M. Suryanegara and K. Ramli, "Measuring Quality of Service (QoS) and Quality of Experience (QoE) on 5G Technology: A Review," 2019 IEEE International Conference on Innovative Research and Development (ICIRD), Jakarta, Indonesia, 2019, pp. 1-6, doi: 10.1109/ICIRD47319.2019.9074681.

[20] Ericsson – edge deployment strategies for communications service providers, https://www.ericsson.com/49f4a5/assets/local/edge-computing/doc/edge-computing-deployment-report-2023.pdf, 2023

[21] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta and D. Sabella, "On Multi-Access edge Computing: A Survey of the Emerging 5G Network edge Cloud Architecture and Orchestration," in IEEE Communications Surveys & Tutorials, vol. 19, no. 3, pp. 1657-1681, thirdquarter 2017, doi: 10.1109/COMST.2017.2705720.

[22] A. Aslan, G. Bal and C. Toker, "Dynamic Resource Management in Next Generation Networks with Dense User Traffic," 2020 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom), Odessa, Ukraine, 2020, pp. 1-6, doi: 10.1109/BlackSeaCom48709.2020.9235006.

[23] H. Zhou, K. Jiang, X. Liu, X. Li and V. C. M. Leung, "Deep Reinforcement Learning for Energy-Efficient Computation Offloading in Mobile-edge Computing," in IEEE Internet of Things Journal, vol. 9, no. 2, pp. 1517-1530, 15 Jan.15, 2022, doi: 10.1109/JIOT.2021.3091142.

[24] Behrisch, Michael and Bieker, Laura and Erdmann, Jakob and Krajzewicz, Daniel (2011) *SUMO – Simulation of Urban MObility: An Overview.* In: Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation. ThinkMind. SIMUL 2011, 23.-28. Okt. 2011, Barcelona. ISBN 978-1-61208-169-4.

[25] *Traffic Modeling with SUMO: a Tutorial*, arXiv:2304.05982 [cs.NI] (or arXiv:2304.05982v1 [cs.NI] for this version) https://doi.org/10.48550/arXiv.2304.05982

[26] Abdulhalim Fayad, Tibor Cinkler (2023). Optimal Slicing of mmWave Micro Base Stations for 5G and Beyond, Volume 3, Issue 3, pp. 99–108., https://doi.org/10.33969/J-NaNA.2023.030301.

[27] Garcia-López, J.M., Ilchenko, K., Nazarenko, O. (2017). Optimization Lab Sessions: Major Features and Applications of IBM CPLEX. In: Póvoa, A., Corominas, A., de Miranda, J. (eds) Optimization and Decision Support Systems for Supply Chains. Lecture Notes in Logistics. Springer, Cham. https://doi.org/10.1007/978-3-319-42421-7_10

[28] Harvey M. Salkin, Cornelis A. De Kluyver: The knapsack problem: A survey, Naval Research Logistics Quarterly, Volume22, Issue1, March 1975, Pages 127-144

[29] Dorit S. Hochba: Approximation Algorithms for NP-Hard Problems, ACM SIGACT News, Volume 28, Issue 2, June 1997, pp 40–52, https://doi.org/10.1145/261342.571216

[30] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, Wojciech Zaremba: OpenAI Gym, arXiv:1606.01540v1, https://doi.org/10.48550/arXiv.1606.01540, 2016

[31] J. Fernando Hernandez-Garcia, Richard S. Sutton: Understanding Multi-Step Deep Reinforcement Learning: A Systematic Study of the DQN Target, arXiv:1901.07510v2, https://doi.org/10.48550/arXiv.1901.07510, 2019

[32] Jesse Clifton, Eric Laber: Q-Learning: Theory and Applications, Annual Review of Statistics and Its Application 2020 7:1, 279-301

[33] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, Noah Dormann: Stable-baselines3: reliable reinforcement learning implementations, The Journal of Machine Learning Research, Volume 22, Issue 1, Article No.: 268, pp 12348–12355, 2021

[34] N. Ari and M. Ustazhanov, "Matplotlib in python," 2014 11th International Conference on Electronics, Computer and Computation (ICECCO), Abuja, Nigeria, 2014, pp. 1-6, doi: 10.1109/ICECCO.2014.6997585.

[35] Oron Anschel, Nir Baram, Nahum Shimkin: Averaged-DQN: Variance Reduction and Stabilization for Deep Reinforcement Learning, arXiv:1611.01929v4, https://doi.org/10.48550/arXiv.1611.01929, 2017

[36] L. Busoniu, R. Babuska and B. De Schutter, "A Comprehensive Survey of Multiagent Reinforcement Learning," in IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 38, no. 2, pp. 156-172, March 2008, doi: 10.1109/TSMCC.2007.913919.