UNIVERZITET U BEOGRADU

SAOBRAĆAJNI FAKULTET

Pavle D. Bugarčić

UNAPREĐENJE PROTOKOLA RUTIRANJA ZA DINAMIČKE BEŽIČNE *AD HOC* MREŽE KORIŠĆENJEM MAŠINSKOG UČENJA

doktorska disertacija

Beograd, 2025.

UNIVERSITY OF BELGRADE

THE FACULTY OF TRANSPORTATION AND
TRAFFIC ENGINEERING

Pavle D. Bugarčić

ENHANCEMENT OF ROUTING PROTOCOLS FOR DYNAMIC WIRELESS AD HOC NETWORKS USING MACHINE LEARNING

Doctoral Dissertation

Belgrade, 2025.

Mentori:

dr Marija Malnar, redovni profesor,
Univerzitet u Beogradu – Saobraćajni fakultet;

dr Nenad Jevtić, vanredni profesor,
Univerzitet u Beogradu – Saobraćajni fakultet.

Članovi komisije:

dr Goran Marković, redovni profesor,
Univerzitet u Beogradu – Saobraćajni fakultet;

dr Mirjana Stojanović, redovni profesor,
Univerzitet u Beogradu – Saobraćajni fakultet;

dr Nataša Nešković, redovni profesor,
Univerzitet u Beogradu – Elektrotehnički fakultet;

dr Valentina Radojičić, redovni profesor,
Univerzitet u Beogradu – Saobraćajni fakultet;

dr Predrag Ivaniš, redovni profesor,
Univerzitet u Beogradu – Elektrotehnički fakultet.

Datum odbrane: _____

UNAPREĐENJE PROTOKOLA RUTIRANJA ZA DINAMIČKE BEŽIČNE *AD HOC* MREŽE KORIŠĆENJEM MAŠINSKOG UČENJA

Sažetak: Sa razvojem pametnih gradova i inteligentnih transportnih sistema, primena bežičnih *ad hoc* mreža (*Wireless Ad hoc Networks*, WANETs), a posebno bežičnih *ad hoc* mreža za vozila (*Vehicular Ad hoc Networks*, VANETs) i bežičnih *ad hoc* mreža za letelice (*Flying Ad hoc Networks*, FANETs) sve više dobija na značaju. VANET i FANET mreže mogu se svrstati u grupu dinamičkih WANET mreža. Njihova izuzetna dinamičnost dovodi do učestalih prekida veza između čvorova, što otežava proces izbora optimalne putanje za slanje paketa i dovodi do degradacije mrežnih performansi. Jedan od načina da se ovaj problem prevaziđe jeste primena veštačke inteligencije u protokolima rutiranja. Veoma značajna oblast veštačke inteligencije koja se sve više primenjuje u dinamičkim WANET mrežama je mašinsko učenje (*Machine Learning*, ML), a posebno se ističe tip ML pod nazivom učenje potkrepljivanjem (*Reinforcement Learning*, RL). Najčešće korišćeni RL algoritmi u protokolima rutiranja za dinamičke WANET mreže su Q-učenje (*Q-Learning*, QL) i duboko učenje potkrepljivanjem (*Deep Reinforcement Learning*, DRL). U okviru ove disertacije predstavljen je pregled najznačajnijih protokola rutiranja baziranih na RL za dinamičke WANET mreže, njihova klasifikacija i poređenje. Zatim je opisan novi protokol rutiranja za urbane VANET mreže baziran na QL (*QL-based Dynamic Routing Algorithm for urban VANETs*, Q-DRAV), koji uključivanjem relevantnih mrežnih parametara u RL proces uspeva značajno da poboljša ukupne mrežne performanse VANET mreža. Novi protokol smanjuje gubitke paketa, kašnjenje paketa i džiter, dok istovremeno povećava protok paketa pri promenljivoj gustini i brzini vozila u mreži. Simulaciona analiza i poređenje sa drugim protokolima rutiranja izvršeni su u NS-3 simulatoru.

Ključne reči: WANET, VANET, FANET, protokoli rutiranja, mašinsko učenje, učenje potkrepljivanjem, Q-učenje, simulaciona analiza, NS-3 simulator

Naučna oblast: Saobraćajno inženjerstvo

Uža naučna oblast: Eksploatacija telekomunikacionog saobraćaja i mreža

ENHANCEMENT OF ROUTING PROTOCOLS FOR DYNAMIC WIRELESS AD HOC NETWORKS USING MACHINE LEARNING

Abstract: With the development of smart cities and intelligent transportation systems, the application of wireless ad hoc networks (WANETs), particularly vehicular ad hoc networks (VANETs) and flying ad hoc networks (FANETs), is gaining significance. VANETs and FANETs could be classified as dynamic WANETs. The exceptional dynamism of these networks leads to frequent disconnections between network nodes, complicating the process of selecting optimal routes for packet transmission and resulting in a degradation of network performance. One way to overcome this problem is through the application of artificial intelligence in routing protocols. A particularly significant area of artificial intelligence increasingly applied in dynamic WANETs is machine learning (ML), with a notable emphasis on a type of ML known as reinforcement learning (RL). The most commonly used RL algorithms in routing protocols for dynamic WANETs are Q-learning (QL) and deep reinforcement learning (DRL). This dissertation presents an overview of the most significant RL-based routing protocols for dynamic WANETs, including their classification and comparison. Subsequently, a new QL-based dynamic routing algorithm for urban VANETs (Q-DRAV) is introduced. By incorporating relevant network parameters into the RL process, this protocol significantly improves the overall network performance of VANETs. The new protocol reduces packet loss, packet delay, and jitter while simultaneously increasing packet throughput under varying vehicle density and speed in the network. Simulation analysis and comparisons with other routing protocols will be conducted using the NS-3 simulator.

Keywords: WANET, VANET, FANET, routing protocols, machine learning, reinforcement learning, Q-learning, simulation analysis, NS-3 simulator

Scientific field: Traffic Engineering

Scientific subfield: Operation of telecommunication traffic and networks

Sadržaj

Spisak slika

Spisak tabela

1. Uvod

Savremeni saobraćajni trendovi sve više ukazuju na potrebu za razvojem pametnih gradova i inteligentnih transportnih sistema. Uporedo sa tim, rastu i zahtevi za pouzdanom i kvalitetnom komunikacijom između vozila koja su njihov neizostavni deo. Jedan od najefikasnijih načina za realizaciju ove komunikacije je kreiranje bežičnih *ad hoc* mreža za vozila (*Vehicular Ad hoc Networks,* VANETs), zbog čega ove mreže sve više dobijaju na značaju. VANET mreže predstavljaju specifičan tip mobilnih bežičnih *ad hoc* mreža (*Mobile Ad hoc Networks,* MANETs), kod kojih mrežne čvorove čine vozila koja žele da ostvare bežičnu komunikaciju. MANET mreže su deo šire grupacije bežičnih *ad hoc* mreža (*Wireless Ad hoc Networks,* WANETs). Sa razvojem bespilotnih letelica, sve veći značaj dobija još jedan tip MANET mreža, a to su bežične *ad hoc* mreže za letelice (*Flying Ad hoc Networks,* FANETs). Osnovna karakteristika svih *ad hoc* mreža je odsustvo bilo kakve fiksne komunikacione infrastrukture, već se veza između participirajućih čvorova formira samo u slučaju potrebe za slanjem podataka. U procesu određivanja optimalne putanje za slanje podataka učestvuju svi čvorovi u mreži, prateći odgovarajući protokol rutiranja. Zbog visoke dinamičnosti čvorova u VANET i FANET mrežama, možemo ih svrstati u zajedničku grupu dinamičkih WANET mreža. U okviru disertacije, glavni fokus je na unapređenju procesa rutiranja u VANET mrežama, za koje je predložen i novi protokol rutiranja. Takođe, uzevši u obzir njihovu bliskost sa VANET mrežama, FANET mreže su uključene u pregled literature, klasifikaciju i poređenje protokola rutiranja.

Iako celularne mreže (poput 5G i 6G) u poslednje vreme privlače značajnu pažnju zbog visokih protoka i malih kašnjenja koje obezbeđuju pri prenosu podataka, dinamičke WANET mreže i dalje nude brojne prednosti, čime ostaju veoma važan deo inteligentnih transportnih sistema. Najveća prednost ovih mreža leži u njihovoj jednostavnosti i relativno maloj ceni implementacije, s obzirom da nisu potrebna visoka inicijalna ulaganja u kompleksnu mrežnu infrastrukturu. Takođe, veliki problem 5G i 6G mreža je njihova i dalje ograničena rasprostranjenost, posebno u slabije razvijenim regionima, prvenstveno zbog visokih troškova implementacije. Stoga, u ovakvim uslovima uspostavljanje VANET i FANET mreža predstavlja najbolji način za ostvarivanje bežične komunikacije između vozila i bespilotnih letelica. Za kreiranje ovih mreža potrebno je samo obezbediti adekvatne komunikacione module u samim čvorovima (vozilima/bespilotnim letelicama), bez potrebe za ulaganjem u drugu komunikacionu infrastrukturu. Ove mreže mogu biti i komplementarne sa celularnim mrežama, tako da se komunikacija u onim oblastima i regionima koji nisu pokriveni odgovarajućom celularnom mrežom može odvijati formiranjem VANET ili FANET mreže.

Protokoli rutiranja u bežičnim komunikacionim mrežama imaju ulogu da obezbede optimalnu putanju za prenos podataka od izvora do odredišta, preko niza međučvorova. U slučaju visoko dinamičkih mreža, kao što su VANET i FANET mreže, taj zadatak postaje dosta složeniji. Osnovna karakteristika ovih mreža je relativno velika brzina kretanja čvorova, što prouzrokuje česte promene u topologiji mreže. To dovodi do čestih prekida linkova koji su deo trenutnih putanja, što značajno otežava održavanje optimalne putanje za slanje podataka. Rešenje ovog problema može biti u uključivanju veštačke inteligencije u proces izbora optimalne putanje. Najčešće korišćena oblast veštačke inteligencije u optimizaciji protokola rutiranja za dinamičke WANET mreže je mašinsko učenje (*Machine Learning,* ML), koje omogućava automatsko unapređenje performansi agenta

učenja kroz svoje iskustvo. ML je moguće podeliti na tri osnovna tipa: nadgledano učenje (*Supervised Learning*, SL), nenadgledano učenje (*Unsupervised Learning*, UL) i učenje potkrepljivanjem (*Reinforcement Learning*, RL) (Sutton i Barto, 2018). Cilj SL je da predvidi vrednosti izlaza (ciljne promenljive) na osnovu definisanih ulaza (uticajnih faktora). Ovo podrazumeva unapred poznat skup mogućih vrednosti ciljne promenljive. SL se uglavnom koristi kod klasifikacije i regresione analize (predviđanja budućih vrednosti ciljne promenljive). UL podrazumeva učenje ponašanja modela, bez znanja o mogućim vrednostima izlaza (ciljne promenljive). Najčešće se koristi za klasterovanje podataka, odnosno grupisanje podataka na osnovu njihovih karakteristika. Tip mašinskog učenja koji najviše odgovara zadacima rutiranja u dinamičkim mrežama je RL. Ovaj pristup podrazumeva stalnu interakciju agenta učenja sa okruženjem putem preduzimanja određenih akcija, na koje okruženje odgovara dodeljivanjem nagrade (ili kazne) za preduzetu akciju, kao i pružanjem povratnih informacija o novom stanju okruženja nakon preduzete akcije. Na ovaj način okruženje potkrepljuje agenta znanjem o korisnosti akcija koje preduzima. Agent tokom vremena pokušava da maksimizira nagradu kroz optimizaciju izbora mogućih akcija.

Tradicionalni protokoli rutiranja za MANET mreže, poput *Ad hoc on-demand distance vector* (AODV) protokola (Perkins i ostali, 2003), ne uspevaju pravovremeno da reaguju na brze promene u VANET i FANET mrežama. Ovo dovodi do kasne promene putanje koja se koristi za prenos podataka u slučaju prekida nekog linka na toj putanji, što rezultira pogoršanjem ukupnih mrežnih performansi. Brojni autori su pokušali da prevaziđu ovaj problem prilagođavanjem postojećih protokola rutiranja dinamičkoj prirodi ovih mreža, kao i definisanjem odgovarajućih metrika rutiranja sa relativno ograničenim uspehom (De Assis i ostali, 2023; Malnar i Jevtić, 2022; Mubarek i ostali, 2018; Saini i Sharma, 2020).

Poslednjih godina, ovaj problem se nastoji efikasnije rešiti primenom RL u procesu izbora optimalne putanje za slanje podataka. Jedan od prvih, a ujedno i veoma značajan primer, je *Q-learning* AODV (QLAODV) protokol (C. Wu i ostali, 2010), koji primenjuje RL za pronalaženje optimalne putanje za prosleđivanje paketa u VANET mrežama, uzimajući u obzir mobilnost vozila i opterećenje propusnog opsega. Problem kod primene ovog protokola su prvenstveno visoki gubici paketa, posebno u slučaju gustih mreža, a zatim i značajna kašnjenja paketa i veliki džiter (*jitter*), čak i kod mreža sa manjim brojem čvorova. Još jedan uspešan protokol koji koristi RL za rutiranje u VANET mrežama je *Adaptive routing protocol based on reinforcement learning* (ARPRL) (J. Wu i ostali, 2018), koji pri izboru optimalne putanje uzima u obzir stabilnost linka, brzine kretanja vozila i gubitak paketa. Najveći nedostatak ovog protokola su loše mrežne performanse u približno stacionarnim mrežama, koje su česta pojava u gustim gradskim jezgrima kada se vozila sporo kreću usled saobraćajnih gužvi. Takođe, problem su i visoka kašnjenja paketa i džiter kod gradskih mreža manjih dimenzija.

Proces rutiranja pomoću RL su na različite načine pokušavali da unaprede i drugi autori. Neki protokoli za rutiranje podataka koriste ne samo mrežu vozila, već i spoljnu infrastrukturu (Jiang i ostali, 2023; B. Liu i ostali, 2023; Yang i Yoo, 2024). Jedan od ključnih aspekata u procesu optimizacije rutiranja u određenim istraživanjima je bezbednost (B. Liu i ostali, 2023; Sarker i ostali, 2023), dok neki istraživači rutiranje zasnivaju na pozicijama raskrsnica (Jiang i ostali, 2023; Rui i ostali, 2023; Yang i Yoo, 2024). Na efikasnost procesa rutiranja u svim ovim protokolima pretežno utiče izbor relevantnih uticajnih faktora koji će kroz RL biti uključeni u proces odabira optimalne putanje za slanje podataka. U različitim istraživanjima korišćeni su raznovrsni uticajni faktori, u zavisnosti od osnovnog cilja optimizacije procesa rutiranja. Uglavnom protokoli uzimaju istovremeno u obzir više uticajnih faktora. Neki od najčešćih su pouzdanost i kvalitet veze ka potencijalnom sledećem čvoru (hopu) na putanji (Jafarzadeh i ostali, 2022; C. Wu i ostali, 2018; J. Wu i ostali, 2018; J. Wu i ostali, 2020; Yang i Yoo, 2024; D. Zhang, T. Zhang i Liu, 2019), stabilnost putanje (C. Wu i ostali, 2010; J. Wu i ostali, 2018), broj hopova koji su potrebni da bi

paket stigao do odredišta (C. Wu i ostali, 2018; Yang i Yoo, 2024; D. Zhang, T. Zhang i Liu, 2019), dostupni propusni opseg veze (Jiang i ostali, 2023; C. Wu i ostali, 2010; D. Zhang, T. Zhang i Liu, 2019), ostvareni protok paketa (D. Zhang, Yu i Yang, 2019), kašnjenje paketa (Yang i Yoo, 2024), rastojanje do odredišta (Jafarzadeh i ostali, 2022; Jiang i ostali, 2021; Jiang i ostali, 2023; B. Liu i ostali, 2023), brzina čvorova (Jiang i ostali, 2023; Saravanan i Ganeshkumar, 2020; Sarker i ostali, 2023; J. Wu i ostali, 2018), nivo snage signala na prijemu (Jiang i ostali, 2023), itd. Često je veoma bitno da li je sledeći čvor ujedno i odredišni, odnosno da li sledeći čvor zna putanju do odredišnog čvora (B. Liu i ostali, 2023; Luo i ostali 2022; Rui i ostali, 2023; C. Wu i ostali, 2010; C. Wu i ostali, 2018). Ako je cilj protokola optimizacija potrošnje energije, gubitak energije će biti ključni faktor pri određivanju vrednosti funkcije nagrade (Ye i ostali, 2021). S druge strane, ako je fokus na zaštiti od neželjenih i malicioznih uticaja, važan uticajni faktor će biti bezbednosni aspekt potencijalnog sledećeg čvora na putanji do odredišta (Sarker i ostali, 2023).

Pažljivim pregledom i analizom trenutnih dostignuća u primeni RL u protokolima rutiranja za dinamičke WANET mreže, može se zaključiti da se korišćenjem ove tehnike značajno unapređuju ukupne mrežne performanse i nadmašuju rezultati postignuti primenom tradicionalnih protokola rutiranja. Ipak, s obzirom na brojne mogućnosti koje RL pruža, još uvek postoji značajan potencijal za dalji napredak u optimizaciji procesa rutiranja i razvoj novih algoritama rutiranja baziranih na RL.

1.1. Predmet, cilj i metode istraživanja

Izbor optimalne putanje za slanje podataka u dinamičkim WANET mrežama predstavlja kompleksan zadatak, s obzirom na stalne promene u mrežnoj topologiji i brzinu kretanja mrežnih čvorova. Jedan od pristupa za prevazilaženje ovog problema je korišćenje RL u protokolima rutiranja, koji omogućava stalno praćenje promena u mrežnom okruženju, čime se proces rutiranja prilagođava dinamičkoj prirodi ovih mreža. Fokus disertacije je na istraživanju mogućnosti primene RL u protokolima rutiranja za dinamičke WANET mreže, kao i na daljem unapređenju procesa rutiranja u ovim mrežama korišćenjem ove tehnike.

U skladu s tim, najpre je sproveden detaljan pregled aktuelnih rezultata primene RL u protokolima rutiranja za dinamičke WANET mreže. Uzimajući u obzir uočene prednosti i nedostatke aktuelnih protokola, predložen je novi protokol rutiranja za VANET mreže zasnovan na RL, koji unapređuje mrežne performanse u odnosu na svoje prethodnike. Poređenje mrežnih performansi izvršeno je sprovođenjem simulacione analize u pogodno odabranim simulacionim scenarijima. Na osnovu dobijenih rezultata simulacija i njihove adekvatne analize, doneti su zaključci o validnosti predloženog protokola. U skladu sa prethodno navedenim, predmet istraživanja može se razložiti na sledeće:

- analiza problema rutiranja u dinamičkim WANET mrežama;
- pregled i analiza primene RL tehnike u protokolima rutiranja za dinamičke WANET mreže, sa ciljem unapređenja ukupnih mrežnih performansi;
- klasifikacija i poređenje aktuelnih protokola rutiranja baziranih na RL za dinamičke WANET mreže;
- izbor postojećih protokola rutiranja baziranih na RL sa kojima će se porediti novi protokol rutiranja;
- razvoj novog protokola rutiranja za VANET mreže baziranog na RL;
- analiza simulacionog okruženja za testiranje protokola;

- implementacija novog i odabranih postojećih protokola baziranih na RL u simulaciono okruženje;
- poređenje primene novog i odabranih postojećih protokola baziranih na RL u pažljivo odabranim simulacionim uslovima;
- analiza rezultata simulacija i davanje smernica za dalja istraživanja.

Osnovni cilj istraživanja je unapređenje mrežnih performansi dinamičkih WANET mreža kroz poboljšanje procesa rutiranja paketa korišćenjem RL tehnike. Da bi se postigao ovaj cilj, bilo je prvo potrebno sprovesti detaljnu klasifikaciju i poređenje aktuelnih rezultata primene RL u protokolima rutiranja za dinamičke WANET mreže. U ovom kontekstu, razmotrene su primene RL kako u VANET, tako i u FANET mrežama, jer se mnoga rešenja iz protokola rutiranja za FANET mreže mogu primeniti i u VANET mrežama. Posebna pažnja posvećena je tipu RL koji se primenjuje, a najčešće korišćeni tip je Q-učenje (*Q-Learning*, QL) (Arafat i Moh, 2021; Da Costa i ostali, 2021; Hosseinzadeh i ostali, 2023; S. Jiang i ostali, 2021; Jiang i ostali, 2023; X. Liu i ostali, 2023; Lolai i ostali, 2022; Luo i ostali, 2022; Rui i ostali, 2023; Sarker i ostali, 2023; Sliwa i ostali, 2021; Yang i Yoo, 2024; W. Zhang i ostali, 2021; itd.). Drugi najzastupljeniji tip RL za unapređenje procesa rutiranja u dinamičkim WANET mrežama je duboko učenje potkrepljivanjem (*Deep Reinforcement Learning*, DRL) (Ahmed i ostali, 2023; Ayub i ostali, 2022; B. Liu i ostali, 2023; Song i ostali, 2024; Upadhyay i ostali, 2023; Ye i ostali, 2021; itd.). Ostali tipovi RL koji se primenjuju u protokolima su duelno duboko učenje potkrepljivanjem (*Dueling Deep Reinforcement Learning*, DDRL) (D. Zhang, Yu i Yang, 2019), RL zasnovan na modelu okruženja (*Model-Based RL*, MBRL) (Jafarzadeh i ostali, 2022) i SARSA algoritam (Bi i ostali, 2020). Takođe, uzete su u obzir i druge tehnike koje se eventualno koriste u protokolima u kombinaciji sa RL, kao što su softverski definisane mreže (*Software-Defined Networking*, SDN) (D. Zhang, Yu i Yang, 2019; D. Zhang i ostali, 2020), *blockchain* (BC) (D. Zhang, Yu i Yang, 2019), kao i *fuzzy* logika (FL) (An i ostali, 2018; Jafarzadeh i ostali, 2022; Jiang i ostali, 2021; C. Wu i ostali, 2018; Yang i ostali, 2020; W. Zhang i ostali, 2021).

Nakon toga, na osnovu uočenih prednosti i nedostataka aktuelnih istraživanja, predložen je i razvijen novi protokol rutiranja za VANET mreže koji za izbor optimalne putanje za slanje paketa koristi RL baziran algoritam. Novi protokol je razvijen sa ciljem da unapredi mrežne performanse u poređenju sa ranijim protokolima rutiranja, uzimajući u obzir relevantne uticajne faktore. Mrežne performanse koje je bilo potrebno unaprediti su procenat izgubljenih paketa (*Packet Loss Ratio*, PLR), ostvareni aplikacioni protok (*application throughput*), kašnjenje paketa s kraja na kraj mreže (*End-to-End Delay*, E2ED) i džiter (varijacija kašnjenja paketa). Poređenje protokola je moguće izvršiti u različitim mrežnim simulatorima, a u ovom istraživanju korišćen je *Network Simulator 3* (NS-3), koji je jedan od najpoznatijih mrežnih simulatora otvorenog koda. Simulatori otvorenog koda su pogodni za ovu svrhu jer omogućavaju istraživačima da prilagođavaju postojeće modele, kao i da implementiraju i testiraju potpuno nove modele. Zato je najpre bilo neophodno upoznati se sa strukturom ovog simulatora, zatim implementirati odgovarajuće protokole rutiranja u simulaciono okruženje, izvršiti intenzivne simulacije i na kraju prikupiti i adekvatno obraditi rezultate simulacija. Na osnovu ovih rezultata izvršena je analiza i poređenje mrežnih performansi postignutih primenom odabranih postojećih protokola i novog protokola rutiranja. U skladu sa svim navedenim, osnovni cilj istraživanja može se razložiti na više naučnih podciljeva:

- klasifikovati aktuelne protokole rutiranja bazirane na RL za dinamičke WANET mreže, sa posebnim osvrtom na tip RL koji se koristi i primenu drugih tehnika u procesu rutiranja;
- izvršiti detaljno poređenje aktuelnih protokola rutiranja baziranih na RL za dinamičke WANET mreže, na osnovu uticajnih faktora u RL procesu, posmatranih mrežnih performansi i korišćenog simulacionog alata za testiranje protokola;

- izabrati odgovarajuće postojeće protokole rutiranja bazirane na RL sa kojima je potrebno uporediti novi protokol rutiranja;
- razviti novi protokol rutiranja za VANET mreže zasnovan na RL, koji bi doprineo unapređenju ključnih mrežnih performansi;
- implementirati novi protokol i odabrane postojeće protokole zasnovane na RL u NS-3 simulaciono okruženje;
- sprovesti intenzivne simulacije s ciljem testiranja novog i postojećih protokola;
- statistički obraditi i analizirati rezultate simulacija;
- doneti adekvatne zaključke i izneti predloge za dalja istraživanja.

U okviru istraživanja korišćene su naučnoistraživačke metode simulacione analize i statističke analize. Simulaciona analiza je sprovedena korišćenjem NS-3 simulatora, dok je Simulator urbane mobilnosti (*Simulation of Urban MObility*, SUMO) poslužio kao pomoćni alat za generisanje modela mobilnosti vozila u simulacijama. Statistička obrada rezultata simulacija, procena vrednosti mrežnih parametara i njihova komparativna analiza izvršeni su korišćenjem programa *Microsof Excel*.

1.2. Osnovne hipoteze

Osnovna polazna hipoteza disertacije je da performanse dinamičkih WANET mreža značajno zavise od protokola rutiranja koji se koristi za određivanje optimalne putanje za slanje paketa podataka u mreži. Protokol treba da prati dinamičku prirodu ovih mreža i da se prilagođava stalnim promenama u mrežnoj topologiji. Tradicionalni protokoli rutiranja nisu uspešno zadovoljili ovaj zahtev, pa je potrebno razviti nove modele koji prate promene u mrežnom okruženju i uzimaju ih u obzir pri izboru optimalne putanje. Iz ove osnovne hipoteze proizilaze sledeće četiri pomoćne hipoteze:

- Korišćenjem RL moguće je značajno unaprediti proces rutiranja u dinamičkim WANET mrežama. RL podrazumeva kontinuirano praćenje promena u mrežnom okruženju kroz interakciju agenta učenja sa okruženjem, što omogućava prilagođavanje protokola rutiranja ovim promenama. Na taj način se mogu postići značajno bolje mrežne performanse u poređenju sa tradicionalnim protokolima rutiranja.
- Odabirom relevantnih uticajnih faktora za RL proces, moguće je dodatno unaprediti mrežne performanse u dinamičkim WANET mrežama. Ovo se prvenstveno odnosi na ključne mrežne performanse, kao što su PLR, ostvareni aplikacioni protok, E2ED i džiter.
- Simulaciona analiza predstavlja pouzdan i ekonomičan način testiranja primene protokola rutiranja u dinamičkim WANET mrežama.
- NS-3 simulator je jedan od najpouzdanijih i najčešće korišćenih simulacionih alata za testiranje prethodno publikovanih istraživanja u ovoj oblasti.

1.3. Pregled sadržaja disertacije

Disertacija je strukturirana tako da prati postavljene ciljeve istraživanja. Nakon uvodnog dela, sadrži sledeća poglavlja:

- U drugom poglavlju detaljnije su opisane bežične WANET mreže, predstavljeni su izazovi sa kojima se susreću protokoli rutiranja u ovim mrežama i pojašnjen je tradicionalni pristup rutiranju podataka u ovim mrežama. Takođe, izvršena je klasifikacija tradicionalnih protokola rutiranja i objašnjen je princip funkcionisanja AODV, *Dynamic source routing*

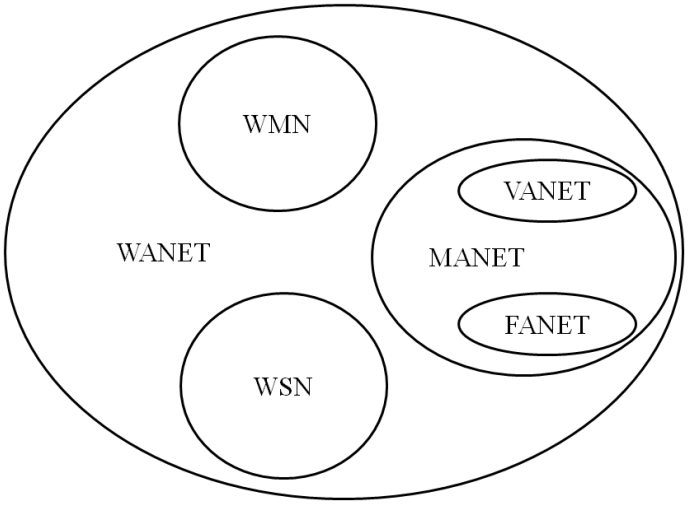
(DSR) (Johnson i ostali, 2007) i *Destination-sequenced distance-vector* (DSDV) (Perkins i Bhagwat, 1994) protokola, koji predstavljaju neke od najznačajnijih tradicionalnih protokola rutiranja.

- U trećem poglavlju najpre su definisani osnovni principi funkcionisanja RL tehnike, predstavljeni su i objašnjeni najznačajniji RL algoritmi, kao i primeri njihove primene u optimizaciji protokola rutiranja za dinamičke WANET mreže. Zatim je izvršen pregled i klasifikacija aktuelnih protokola rutiranja baziranih na RL za dinamičke WANET mreže, kao i kratak opis principa funkcionisanja po jednog protokola iz svake izvedene klase. Nakon toga izvršeno je detaljno poređenje aktuelnih protokola i sumirani su zaključci izvršene klasifikacije i poređenja. Na kraju je objašnjen princip funkcionisanja QLAODV i ARPRL protokola, koji su izabrani za reprezentativne predstavnike RL baziranih protokola rutiranja.
- U četvrtom poglavlju predložen je i detaljno predstavljen novi dinamički protokol rutiranja na bazi RL za urbane VANET mreže (*QL-based dynamic routing algorithm for urban VANETs*, Q-DRAV) (Bugarčić i ostali, 2024). Ovde su opisani metodologija razvoja, detaljna matematička formulacija i princip funkcionisanja novog protokola.
- U petom poglavlju opisano je simulaciono okruženje, u kome je sprovedena simulaciona analiza i poređenje novog i odabranih postojećih protokola rutiranja. Najpre je izvršeno poređenje osnovnih karakteristika najpoznatijih mrežnih simulatora, a zatim su opisani osnovni koncepti i elementi NS-3 simulatora za modelovanje *ad hoc* mreža. Na kraju poglavlja ukratko je opisan SUMO simulator, kao jedan od najznačajnijih simulatora mobilnosti čvorova.
- U šestom poglavlju dat je opis simulacionih parametara i scenarija u kojima su testirani protokoli. Zatim su predstavljeni i upoređeni rezultati simulacione analize i testiranja primene novog Q-DRAV protokola, odabranih QLAODV i ARPRL protokola baziranih na RL i tradicionalnog AODV protokola rutiranja. Protokoli su upoređeni na osnovu PLR, ostvarenog aplikacionog protoka, E2ED i džitera.
- U sedmom poglavlju data su zaključna razmatranja, sumirani su rezultati disertacije, praktični i naučni doprinosi. Takođe, definisane su dalje mogućnosti razvoja oblasti primene RL u protokolima rutiranja za dinamičke WANET mreže.

2. RUTIRANJE U DINAMIČKIM WANET MREŽAMA

Mreže koje nemaju fiksnu komunikacionu infrastrukturu i centralni kontrolni uređaj nazivaju se *ad hoc* mreže. Veza između čvorova formira se "*ad hoc*", odnosno samo u slučaju potrebe za razmenom podataka. Čvorovima je omogućeno da direktno komuniciraju jedni sa drugima, stvarajući tako dinamičku i decentralizovanu mrežu. WANET mreže predstavljaju *ad hoc* mreže kod kojih se komunikacija između čvorova odvija isključivo bežičnim putem. Osnovne karakteristike su im decentralizovana arhitektura, fleksibilnost i rekonfigurabilnost. Decentralizovanost se ogleda u odsustvu centralnog kontrolnog uređaja (na primer rutera), tako što svi čvorovi u mreži ravnopravno učestvuju u određivanju putanje kojom će se podaci slati. Za ove mreže važi da su veoma fleksibilne, što znači da se relativno brzo mogu prilagoditi promenama u mrežnoj topologiji, koje su najčešće posledica kretanja čvorova u mreži. Takođe, ukoliko dođe do prekida mrežnih linkova, ove mreže imaju mogućnost samostalnog rekonfigurisanja putanja kojima se šalju podaci.

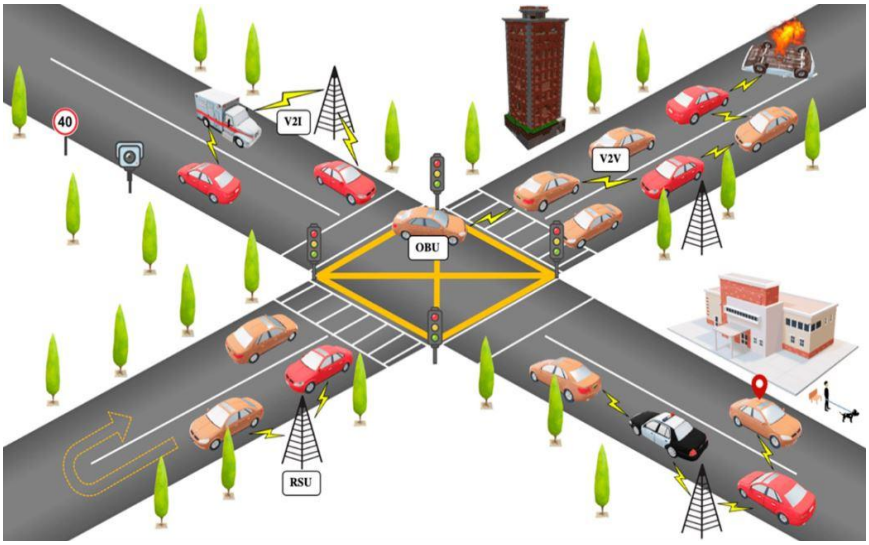
WANET mrežama pripadaju bežične *mesh* mreže (*Wireless Mesh Networks*, WMN), bežične senzorske mreže (*Wireless Sensor Networks*, WSN) i MANET mreže, kao što je ilustrovano na slici 2.1. MANET mreže su poseban tip WANET mreža kod kojih se čvorovi mogu slobodno kretati. Ubrzani razvoj inteligentnih transportnih sistema doveo je do sve veće popularnosti VANET i FANET mreža, koje predstavljaju posebne kategorije MANET mreža kod kojih mrežne čvorove čine vozila i bespilotne letelice (*Unmanned Aerial Vehicles*, UAVs), respektivno.



Slika 2.1. Tipovi WANET mreža

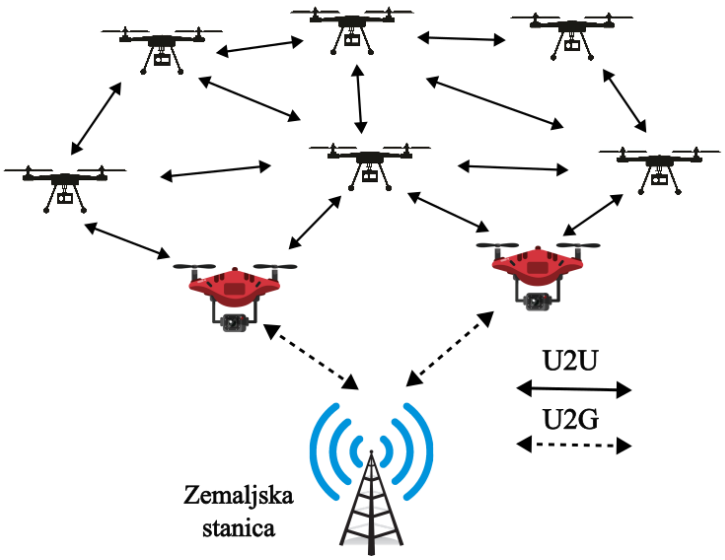
Jedna od karakteristika VANET mreža je da se vozila mogu kretati samo unapred definisanim ograničenim skupom putanja, kao što su gradske saobraćajnice, autoputevi, magistralni putevi, itd. VANET mreže omogućavaju različite vrste komunikacija koje uključuju vozila, ali su dve najvažnije: komunikacija između vozila i infrastrukture (*Vehicle-to-Infrastructure*, V2I), koja se odvija između vozila i fiksnih pristupnih tačaka nazvanih *roadside units* (RSUs) i međusobna komunikacija između vozila (*Vehicle-to-Vehicle*, V2V), gde svi čvorovi čine potpuno mobilnu

mrežu, razmenjujući informacije direktno jedni s drugima bez potrebe za centralnom pristupnom tačkom. Vozila komuniciraju sa drugim vozilima i infrastrukturom putem ugrađenih *on-board unit* (OBU) uređaja. Na slici 2.2 prikazana je ilustracija jedne urbane VANET mreže, koja omogućava V2I i V2V komunikaciju.



Slika 2.2. Ilustracija urbane VANET mreže (izvor: Khan i ostali, 2019)

Sa druge strane, FANET mrežu čini grupa bespilotnih letelica koje imaju mogućnost da međusobno komuniciraju bežičnim putem. Pored toga, neizostavan deo FANET mreže je i zemaljska stanica (*ground station*), preko koje letelice ostvaruju komunikaciju sa zemaljskim komunikacionim sistemima. Najčešće su letelice organizovane hijerarhijski, što znači da samo određeni broj njih može da ostvari komunikaciju sa zemaljskom stanicom (*UAV to Ground station*, U2G), dok sve ostale imaju samo mogućnost međusobne komunikacije (*UAV to UAV*, U2U). Za razliku od VANET mreža kod kojih se vozila kreću u dvodimenzionalnom prostoru unapred definisanim saobraćajnicama, čvorovi u FANET mrežama se kreću u trodimenzionalnom slobodnom prostoru i imaju neograničeno mnogo mogućih putanja. Na slici 2.3 predstavljena je jedna jednostavna FANET mreža, koja se sastoji iz nekoliko bespilotnih letelica i zemaljske stanice. Crvenom bojom su označene bespilotne letelice višeg hijerarhiskog nivoa koje imaju mogućnost U2G komunikacije.



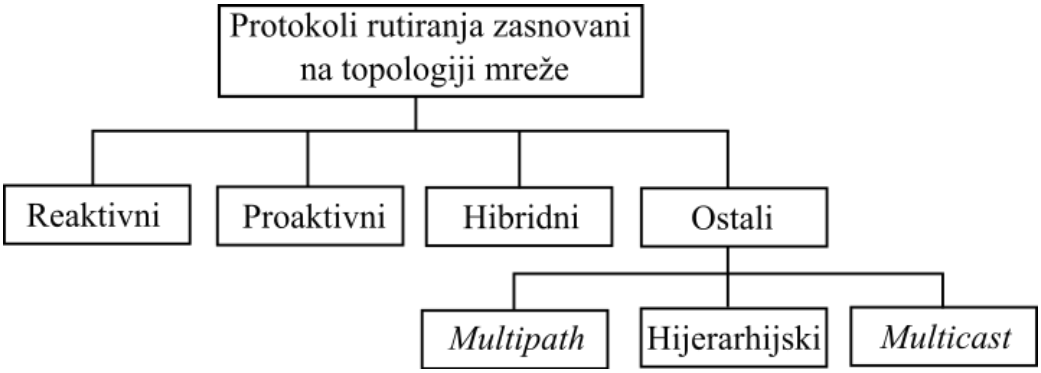
Slika 2.3. Ilustracija FANET mreže

Zbog velike dinamičnosti čvorova, VANET i FANET mreže spadaju u grupu dinamičkih WANET mreža. Osnovna osobina ovih mreže je velika pokretljivost mrežnih čvorova, što prouzrokuje stalne promene u mrežnoj topologiji i česte prekide mrežnih linkova. Iz tog razloga, glavni izazov sa kojim se susreću protokoli rutiranja za ove mreže je održavanje povezanosti čvorova u mreži. U slučaju prekida mrežnog linka kojim se šalju podaci, potrebno je što pre pronaći novu putanju, koja ne uključuje prekinuti link, kako ne bi došlo do velikih gubitaka paketa i kašnjenja u isporuci paketa ka odredištu. Dodatni izazovi sa kojima se susreću FANET mreže su mala gustina mrežnih čvorova, ograničen izvor energije, kao i ograničeni skladišni i procesorski kapaciteti. Sve ovo treba uzeti u obzir pri izboru putanje kojom će se slati podaci, tako da protokoli rutiranja pred sobom nemaju nimalo jednostavan zadatak.

2.1. Tradicionalni protokoli rutiranja

Za određivanje optimalne putanje za slanje paketa u VANET i FANET mrežama, najpre su korišćeni tradicionalni protokoli rutiranja za MANET mreže. Tradicionalne protokole je moguće podeliti na protokole zasnovene na topologiji mreže, protokole zasnovane na poziciji čvorova i kombinovane protokole. U MANET mrežama, najpopularniji su protokoli zasnovani na topologiji mreže i njih je moguće podeliti na sledeće tipove protokola (slika 2.4):

- reaktivni,
- proaktivni,
- hibridni,
- ostali (*multipath*, hijerarhijski i *multicast*).



Slika 2.4. Klasifikacija protokola rutiranja zasnovanih na topologiji mreže

Reaktivni protokoli rutiranja funkcionišu tako što pronalaze putanje između čvorova isključivo na zahtev, odnosno kada postoji potreba za slanjem paketa. Najpoznatiji reaktivni protokoli rutiranja su: AODV, DSR, *Link quality source routing protocol* (LQSR) (Daves i ostali, 2004), *Associativity-based routing* (ABR) (Toh, 1997), *Signal stability-based adaptive routing* (SSBR) (Dube i ostali, 1997) i *The ant-colony based routing algorithm* (ARA) (Gunes i ostali, 2002).

Za razliku od reaktivnih, kod proaktivnih protokola rutiranja čvorovi u svakom trenutku znaju putanje do svih ostalih čvorova u mreži. Ove putanje čvorovi čuvaju u svojim tabelama rutiranja i periodično ih šalju ostalim čvorovima u mreži. Najpoznatiji proaktivni protokoli rutiranja su: DSDV, *Wireless routing protocol* (WRP) (Murthy i Garcia-Luna-Aceves, 1996), *Optimized link state routing* (OLSR) (Clausen i Jacquet, 2003), *Topology dissemination based on reverse-path forwarding* (TBRPF) (Ogier i ostali, 2004) i *Source-tree adaptive routing* (STAR) (Garcia-Luna-Aceves i Spohn, 1999).

Hibridni protokoli rutiranja predstavljaju kombinaciju reaktivnih i proaktivnih protokola. Za udaljene čvorove i retko korišćene putanje primenjuje se reaktivni pristup, dok se za obližnje čvorove i često korišćene putanje koristi proaktivni pristup. Neki od najznačajnijih hibridnih protokola rutiranja su: *Hybrid wireless mesh protocol* (HWMP) (Conner i ostali, 2006), *Zone routing protocol* (ZRP) (Hass i ostali, 2002), *Fisheye state routing multipath* (FSR) (Pei i ostali, 2000), *Hybrid ant colony optimization* (HOPNET) (Wang i ostali, 2009) i *Ad hoc networking with swarm intelligence* (ANSI) (Rajagopalan i Shen, 2006). Postoje i hibridni protokoli koji koriste oba pristupa, po mogućstvu dinamično radi prilagođavanja protokola rutiranja specifičnim uslovima prenosa.

Osim ove tri osnovne grupe protokola, postoje i protokoli rutiranja koje pored načina određivanja putanje (reaktivno, proaktivno ili hibridno) karakterišu druge osobine. U ovu grupu protokola ubrajaju se *multipath,* hijerarhijski i *multicast* protokoli rutiranja:

- *Multipath* protokole karakteriše istovremeno biranje višestrukih putanja od izvora do odredišta. Osnovna prednost ovakvog pristupa je veća pouzdanost, jer u slučaju otkaza jedne putanje čvor ima drugu putanju kojom može da nastavi slanje podataka. U ove protokole ubrajaju se: *Multipath security-aware QoS routing* (MuSeQoR) (Reddy i ostali, 2006), *Ad hoc on-demand multipath distance vector routing* (AOMDV) (Marina i Das, 2001), *Truthful multipath routing protocol* (TMRP) (Wang i ostali, 2008), *Scalable multipath on-demand routing* (SMORT) (Reddy i Raghavan, 2007), itd.
- Hijerarhijske protokole rutiranja karakteriše višeslojna organizacija čvorova u mreži. Čvorovi viših nivoa dobijaju dodatne funkcije, pored onih funkcija koje imaju čvorovi nižih nivoa. U hijerarhijske protokole rutiranja ubrajaju se: *Hierarchical state routing* (HSR) (Iwata i ostali, 1999), *Hierarchical landmark routing* (H-LANMAR) (Xu i ostali 2003), *Core-extraction distributed ad hoc routing* (CEDAR) (Sivakumar i ostali, 1999), itd.
- *Multicast* protokole rutiranja karakteriše mogućnost slanja podataka od jednog izvora do više odredišta. U ovu grupu protokola ubrajaju se: *Dynamic core based multicast routing* (DCMP) (Das i ostali, 2002), *Ad hoc multicast routing protocol* (AMRoute) (Xie i ostali, 2002), *Robust multicasting in ad hoc networks using trees* (ROMANT) (Vaishampayan i Garcia-Luna-Aceves, 2004), *QoS multicast routing protocol for clustering mobile ad hoc networks* (QMRPCAH) (Layuan i Chunlin, 2007), itd.

Imajući u vidu da se u VANET i FANET mrežama najčešće koriste reaktivni protokoli, kao i da su upravo oni bili polazna tačka ovog istraživanja, u nastavku poglavlja detaljnije su opisani najpoznatiji predstavnici ovih protokola – AODV i DSR. Posle reaktivnih, u dinamičkim WANET mrežama najzastupljeniji su proaktivni protokoli, tako da je detaljnije opisan i jedan od najpoznatijih protokola ovog tipa – DSDV.

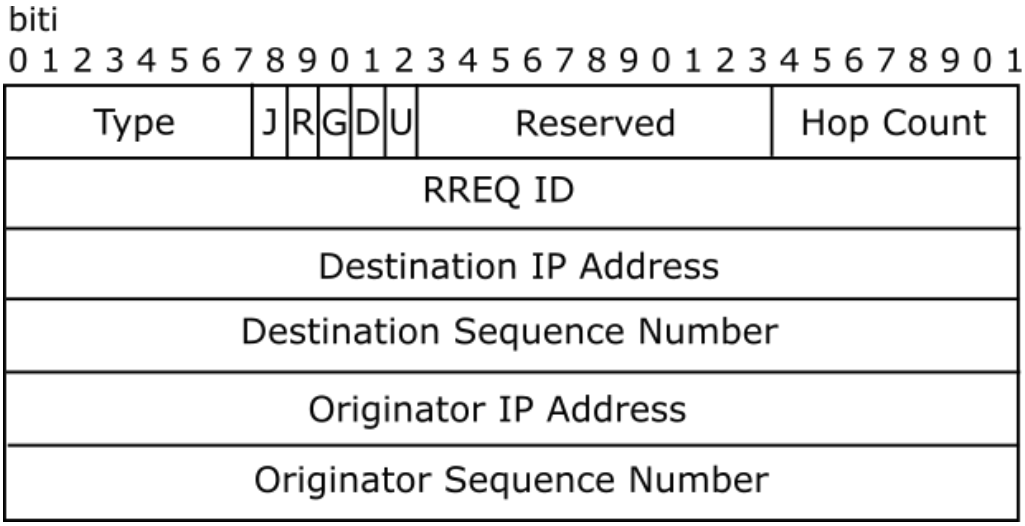
2.2. AODV protokol

AODV predstavlja reaktivni protokol rutiranja koji omogućava dinamično i autonomno *multihop* rutiranje između mobilnih čvorova koji učestvuju u uspostavljanju i održavanju *ad hoc* mreže. AODV omogućava mobilnim čvorovima da brzo dobiju putanje ka novim odredištima i ne zahteva da čvorovi održavaju putanje koje nisu aktivne u komunikaciji. Takođe, omogućava mobilnim čvorovima da pravovremeno reaguju na prekide veze i promene u mrežnoj topologiji. Jedna od karakteristika AODV protokola je korišćenje sekvencijalnog broja odredišta za svaku uspostavljenu putanju. Sekvencijalni broj odredišta kreira odredišni čvor kako bi ga uključio u svaku informaciju o putanji koja se šalje čvoru koji je poslao zahtev za putanjom. Korišćenje sekvencijalnih brojeva odredišta obezbeđuje izbegavanje petlji u rutiranju. U slučaju kada postoji više putanja do odredišta, čvor koji je poslao zahtev za putanju dužan je da izabere onu sa najvećim sekvencijalnim brojem.

Funkcionisanje AODV protokola se bazira na dva mehanizma – otkrivanje putanje (*route discovery*) i održavanje putanje (*route maintenance*). Za ove mehanizme protokol koristi *Route Request* (RREQ), *Route Reply* (RREP), *Route Error* (RERR) i *Hello* kontrolne pakete. Za razmenu ovih paketa koristi se *User datagram protocol* (UDP). S obzirom da će AODV protokol biti testiran u šestom poglavlju, njegova struktura i način funkcionisanja biće nešto detaljnije opisani.

2.2.1. Procedura otkrivanja putanje kod AODV protokola

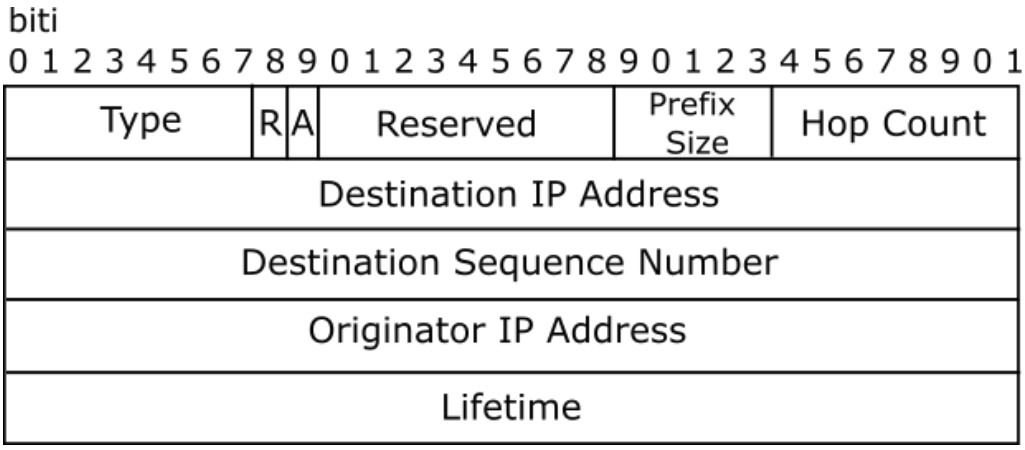
Procedura otkrivanja putanje u AODV protokolu pokreće se kada neki čvor želi da pošalje pakete podataka ka određenom odredišnom čvoru. U ovoj proceduri protokol koristi RREQ i RREP kontrolne pakete. Na slici 2.5 prikazana je struktura jednog RREQ paketa kod AODV protokola. Svaki paket se sastoji iz sledećih polja: *Type*, J, R, G, D, U, *Reserved*, *Hop Count*, RREQ ID, *Destination IP Address*, *Destination Sequence Number*, *Originator IP Address* i *Originator Sequence Number*.



Slika 2.5. Struktura RREQ paketa kod AODV protokola

Polje *Type* kod RREQ paketa uvek ima vrednost 1. Polja J, R, G, D i U koriste se za označavanje određenih stanja ili ponašanja tokom procesa pronalaženja putanje u mreži. *Join* (J) *flag* je rezervisan za multikast. *Repair* (R) *flag* označava da čvor koji šalje RREQ pokušava da popravi putanju do odredišta. *Gratuitous* RREP (G) *flag* ukazuje na to da li međučvor koji zna putanju do odredišta, pored RREP paketa ka izvornom čvoru, treba da pošalje *gratuitous* RREP odredišnom čvoru. *Destination only* (D) *flag* označava da samo odredište može odgovoriti na ovaj RREQ. *Unknown sequnence number* (U) *flag* ukazuje da sekvencijalni broj odredišta nije poznat. Polje *Reserved* se šalje sa vrednošću 0 i ignoriše se na prijemu. Polje *Hop Count* predstavlja brojač hopova (skokova) od pošiljaoca RREQ paketa do čvora koji je trenutno primio RREQ. Polje RREQ ID predstavlja sekvencijalni broj paketa, koji zajedno sa IP adresom čvora pošiljaoca jedinstveno identifikuje određeni RREQ. Polje *Destination IP Address* označava IP adresu odredišta do kog se traži putanja. Polje *Destination Sequence Number* označava poslednji sekvencijalni broj koji je pošiljalac RREQ paketa primio u prošlosti za bilo koju putanju ka odredištu. Polje *Originator IP Address* označava IP adresu pošiljaoca RREQ paketa. Polje *Originator Sequence Number* označava trenutni sekvencijalni broj koji se koristi na početku putanje prema pošiljaocu RREQ paketa.

Na slici 2.6 prikazana je struktura jednog RREP paketa kod AODV protokola. Ovi paketi sadrže sledeća polja: *Type*, R, A, *Reserved*, *Prefix Size*, *Hop Count*, *Destination IP Address*, *Destination Sequence Number*, *Originator IP Address* i *Lifetime*.



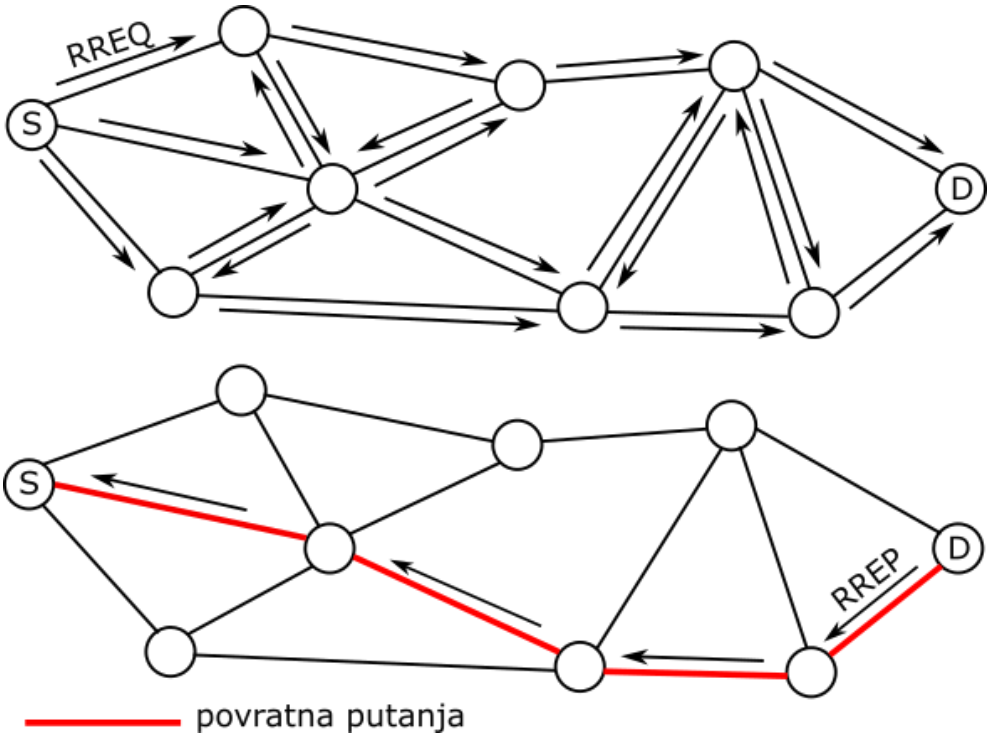
Slika 2.6. Struktura RREP paketa kod AODV protokola

Polje *Type* kod RREP paketa uvek ima vrednost 2. R *flag* ima istu funkciju kao kod RREQ paketa. *Acknowledgment* (A) *flag* označava da je potrebna potvrda o prijemu. Polje *Reserved* se postavlja na 0 i ignoriše se na prijemu. Ako je različito od nule, petobitno polje *Prefix Size* označava da se navedeni sledeći hop može koristiti za bilo koje čvorove sa istim prefiksom rutiranja (kao što je definisano poljem *Prefix Size*) kao traženo odredište. Polje *Hop Count* označva broj hopova od pošiljaoca do primaoca paketa. Polje *Destination IP Address* označava IP adresu odredišta do koga se obezbeđuje putanja. Polje *Destination Sequence Number* označava sekvencijalni broj odredišta. Polje *Originator IP Address* označava IP adresu čvora koji je inicirao slanje RREQ paketa. Polje *Lifetime* predstavlja vreme u milisekundama u okviru kog čvorovi koji primaju RREP smatraju da je putanja ispravna.

Kada izvorni čvor želi da pošalje pakete podataka odredišnom čvoru, prvo difuzno šalje RREQ paket svim čvorovima u mreži. Ovaj paket može primiti neki međučvor ili odredišni čvor. Ako paket primi međučvor, prvo kreira (ili ažurira, ukoliko već postoji) podatak o putanji do izvornog čvora u svojoj tabeli rutiranja. Nakon toga, međučvor proverava u svojoj tabeli rutiranja da li ima putanju do odredišnog čvora i, ukoliko je ima, upoređuje sekvencijalne brojeve odredišta iz primljenog RREQ paketa sa onim u svojoj tabeli rutiranja. Ako je sekvencijalni broj odredišta iz tabele rutiranja veći ili jednak sekvencijalnom broju iz primljenog RREQ paketa, međučvor generiše *unicast* RREP paket ka izvornom čvoru kako bi ga obavestio o putanji do odredišta. Međutim, ako je sekvencijalni broj odredišta iz RREQ paketa veći od sekvencijalnog broja iz tabele rutiranja, to znači da međučvor ima podatak o zastareloj putanji i prosleđuje RREQ dalje ka odredišnom čvoru.

Kada odredišni čvor primi RREQ paket, on takođe kreira (ili ažurira, ako već postoji) podatak o putanji do izvornog čvora u svojoj tabeli rutiranja. Ako je već primio isti RREQ paket od istog izvornog čvora (samo drugom putanjom), odbacuje ga i ne šalje nikakav odgovor izvornom čvoru. U suprotnom, generiše *unicast* RREP paket ka izvornom čvoru koristeći istu putanju kojom je primio RREQ paket. Kada međučvor primi RREP paket, najpre ažurira svoju tabelu rutiranja informacijom o putanji do odredišnog čvora, a zatim prosleđuje ovaj paket ka izvornom čvoru. Na kraju, kada izvorni čvor primi RREP paket, na taj način dobija informaciju o putanji do odredišnog čvora i može započeti slanje paketa podataka. Treba napomenuti da čvorovi u svojim tabelama rutiranja pamte samo adresu sledećeg čvora na putanji kojom šalju pakete.

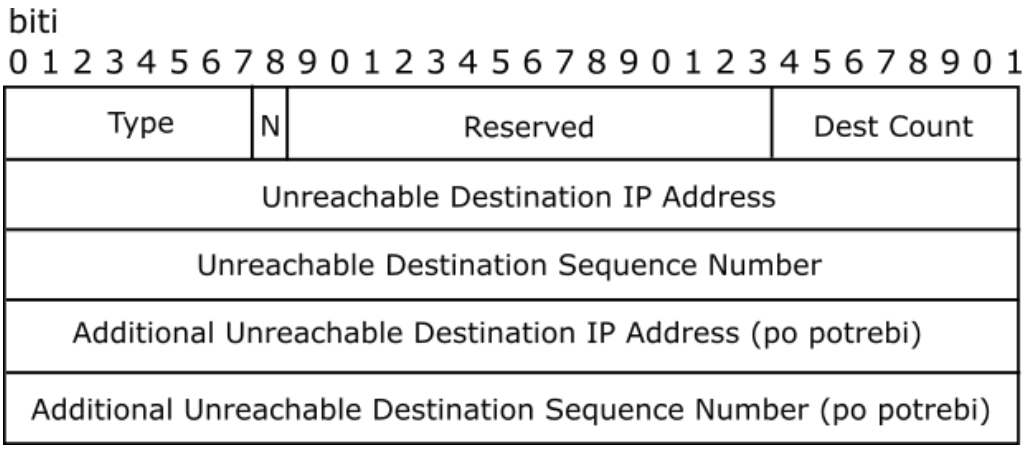
Na slici 2.7 ilustrovana je procedura otkrivanja putanje kod AODV protokola. Izvorni čvor (*source node*) označen je sa S, a odredišni čvor (*destination node*) sa D. Svi ostali čvorovi predstavljaju međučvorove (*intermediate nodes*) na potencijalnoj putanji od izvornog ka odredišnom čvoru.



Slika 2.7. Procedura otkrivanja putanje kod AODV protokola

2.2.2. Procedura održavanja putanje kod AODV protokola

Za proceduru održavanja putanje AODV protokol koristi *Hello* i RERR kontrolne pakete. *Hello* paket predstavlja specijalni tip RREP paketa, koji se koriste za periodično obaveštavanje suseda o prisustvu čvora. Kod ovog paketa vrednost polja *Hop Count* postavljena je na 0 (jer je paket namenjen samo susednim čvorovima). Na slici 2.8 prikazana je struktura RERR paketa koji se sastoji iz sledećih polja: *Type*, N, *Reserved*, *Dest Count*, *Unreachable Destination IP Address*, *Unreachable Destination Sequence Number*, *Additional Unreachable Destination IP Address* (ako je potrebno) i *Additional Unreachable Destination Sequence Number* (ako je potrebno).

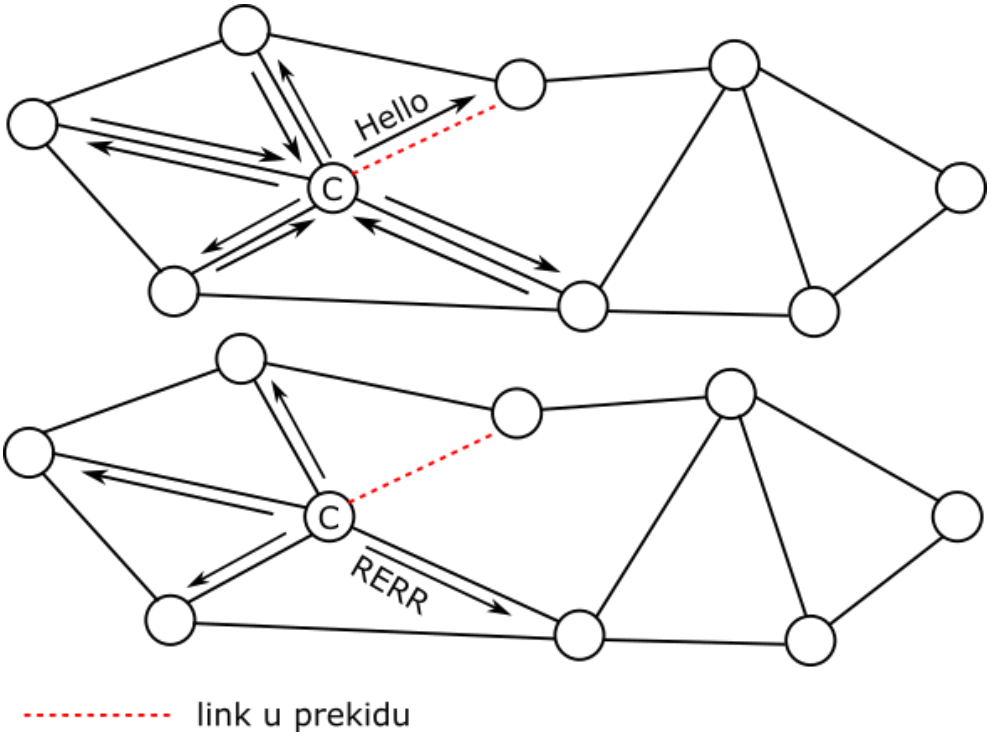


Slika 2.8. Struktura RERR paketa kod AODV protokola

Polje *Type* za RERR paket uvek ima vrednost 3. *No delete* (N) *flag* označava da je čvor izvršio lokalnu popravku veze i čvorovi ka kojima se šalje ne treba da brišu putanju. Polje *Reserved* se postavlja na 0 i ignoriše se na prijemu. Polje *Dest Count* označava broj nedostupnih destinacija uključenih u poruku. Polje *Unreachable Destination IP Address* označava IP adresu odredišta koje je postalo nedostupno zbog prekida veze. Polje *Unreachable Destination Sequence Number*

predstavlja sekvencijalni broj u tabeli rutiranja za destinaciju navedenu u prethodnom polju. Naredna polja se dodaju po potrebi i označavaju IP adresu i sekvencijalni broj dodatnog nedostupnog odredišnog čvora.

Na slici 2.9 ilustrovana je procedura održavanja putanje. Neka određeni čvor C posluži kao primer sprovođenja ove procedure. On najpre proverava da li je poslao neki difuzni paket u poslednjih 1000 ms. Ako nije, slanjem difuznog *Hello* paketa ka svojim susednim čvorovima obaveštava ih da je aktivan. Ukoliko je čvor C u poslednjih 2000 ms dobio od susednog čvora bilo koji paket, a nakon toga isto toliko vremena ne dobije nikakav paket, smatraće da je link ka tom čvoru u prekidu. U tom slučaju, slanjem RERR kontrolnih paketa čvor obaveštava svoje susede o prekidu linka, na osnovu čega oni ažuriraju svoje tabele rutiranja kako bi izbegli slanje podataka putanjama koje sadrže prekinuti link.



Slika 2.9. Procedura održavanja putanje kod AODV protokola

Svaki čvor ima svoju tabelu rutiranja u kojoj čuva podatke određeno vreme, nakon čega ih briše. Podaci se brišu nakon 2 sekunde ako se njima određuje da li je neki čvor aktivan, odnosno nakon 3 sekunde ako se pomoću njih utvrđuje da li je neki link u prekidu. U tabelama rutiranja čvorovi pamte samo podatke o adresama susednih čvorova (odnosno prvih hopova na putanji ka nekom odredištu).

2.3. DSR protokol

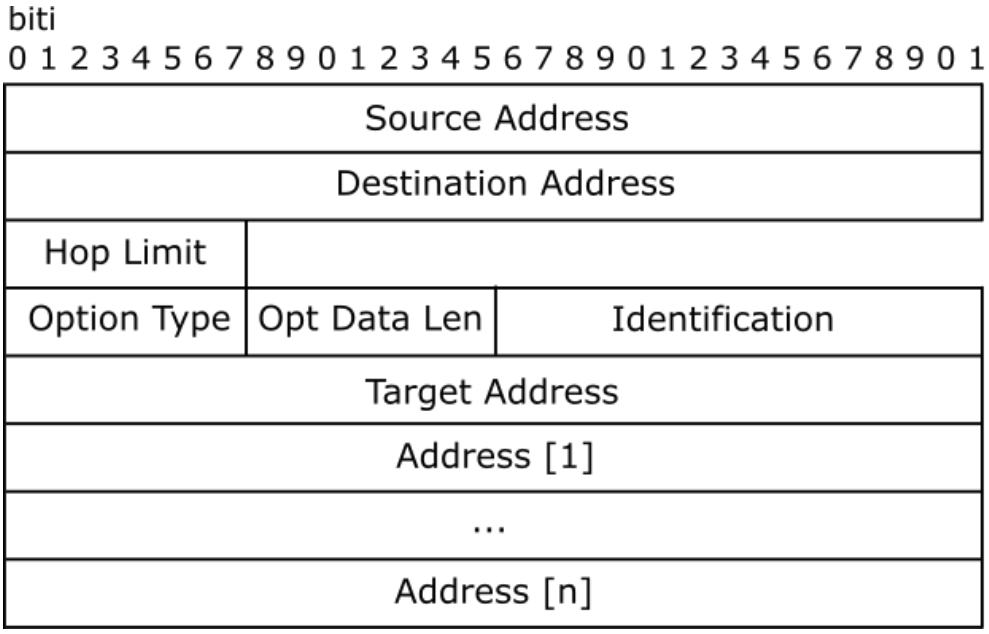
DSR je još jedan popularan reaktivni protokol rutiranja, namenjen mobilnim bežičnim *ad hoc* mrežama sa podrškom za višestruko prosleđivanje (*multihop*). DSR omogućava mreži da se potpuno samostalno organizuje i konfiguriše, bez oslanjanja na bilo kakvu postojeću mrežnu infrastrukturu. Slično kao AODV, DSR protokol funkcioniše korišćenjem mehanizama za otkrivanje i održavanje putanja. Oba mehanizma DSR protokola rade potpuno na zahtev, što znači da se rutiranje vrši samo kada je potrebno uspostaviti putanju između dva čvora ili otkriti promene na postojećim putanjama za slanje podataka. DSR protokol obezbeđuje višestruke putanje do svakog odredišnog čvora, omogućavajući izvornom čvoru da bira i kontroliše putanje koje će se

koristiti za rutiranje njegovih paketa, na primer, radi balansiranja opterećenja ili povećanja pouzdanosti mreže.

DSR protokol je prvenstveno osmišljen za mobilne *ad hoc* mreže, do približno dve stotine čvorova i optimizovan je za pouzdan rad čak i pri vrlo visokim stopama mobilnosti čvorova. Osnovna verzija DSR protokola koristi eksplicitno "izvorno rutiranje", pri čemu svaki paket podataka u svom zaglavlju nosi kompletnu, uređenu listu međučvorova kroz koje treba da prođe na putu do odredišnog čvora. Za razliku od AODV protokola, gde čvorovi u tabelama rutiranja čuvaju samo adresu sledećeg čvora na putanji ka odredištu, kod DSR protokola u tabelama rutiranja se čuvaju kompletne putanje od izvornog do odredišnog čvora, uključujući adrese svih međučvorova kroz koje paket prolazi. Upotreba eksplicitnog izvornog rutiranja omogućava izvornom čvoru da bira i kontroliše putanje koje koristi za slanje svojih paketa, podržava korišćenje više putanja do bilo kog odredišta i jednostavno garantuje da su korišćene putanje bez petlji. Uključivanjem izvorne putanje u zaglavlje svakog paketa podataka, drugim čvorovima koji prosleđuju te pakete omogućava se lako skladištenje informacija o rutiranju, koje mogu biti korišćene u budućnosti. U nastavku su ukratko opisane osnovne procedure ovog protokola – otkrivanje putanje i održavanje putanje.

2.3.1. Procedura otkrivanja putanje kod DSR protokola

Procedura otkrivanja putanje je slična onoj u AODV protokolu i obavlja se uz pomoć dve vrste kontrolnih paketa: RREQ i RREP. Na slici 2.10 prikazan je izgled RREQ paketa kod DSR protokola. Svaki paket se sastoji iz sledećih polja: *Source Address, Destination Address, Hop Limit, Option Type, Opt Data Len, Identification, Target Address, Address [1], ..., Address [n]*.

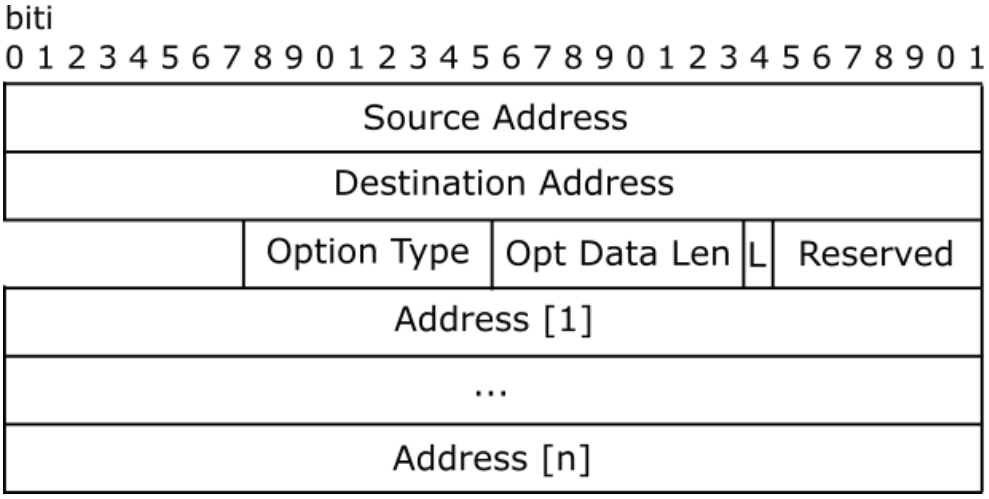


Slika 2.10. Struktura RREQ paketa kod DSR protokola

Polje *Source Address* označava IP adresu izvornog čvora, a *Destination Adress* predstavlja *broadcast* adresu (255.255.255.255), jer se RREQ šalje difuzno u mrežu. *Hop Limit* označava broj hopova koje RREQ sme da prođe. *Option Type* polje služi za identifikovanje vrste paketa (za RREQ paket ima vrednost 1), čvorovi koji ne razumeju opciju u polju ignorišu je. *Opt Data Len* predstavlja dužinu ostatka paketa u bajtima, počev od narednog polja. Vrednost ovog polja mora biti postavljena na $4 \cdot n + 6$, gde je n broj IP adresa u RREQ paketu. Polje *Identification* predstavlja jedinstvenu vrednost koju generiše inicijator (originalni pošiljalac) RREQ paketa. Čvorovi koji iniciraju RREQ generišu novu vrednost polja *Identification* za svaki RREQ zasnovanu, na primer,

na rednom broju na brojaču svih RREQ koje inicira čvor. Ova vrednost omogućava čvoru koji primi paket da utvrdi da li je već primio ovaj RREQ. Ako jeste, mora da odbaci taj paket. *Target Address* je adresa ciljnog čvora za RREQ. Polja *Address [i]*, gde *i* ide od 1 do *n*, označavaju IP adresu *i*-tog čvora na putanji do odredišta, ne računajući izvorni čvor. Svaki čvor koji prosleđuje RREQ paket dodaje svoju adresu na ovu listu, povećavajući vrednost *Opt Data Len* za 4 bajta.

Na slici 2.11 prikazan je izgled RREP paketa kod DSR protokola. Svaki RREP paket sadrži sledeća polja: *Source Address*, *Destination Address*, *Option Type*, *Opt Data Len*, L, *Reserved*, *Address [1]*, ..., *Address [n]*.



Slika 2.11. Struktura RREP paketa kod DSR protokola

Polje *Source Address* označava IP adresu pošiljaoca RREP paketa, dok *Destination Address* predstavlja odredišnu IP adresu RREP paketa (treba da bude ista kao adresa izvornog čvora RREQ paketa). Kao i kod RREQ paketa, *Option Type* polje služi za identifikovanje vrste paketa (za RREP paket ima vrednost 2), čvorovi koji ne razumeju opciju u polju ignorišu je. *Opt Data Len* predstavlja dužinu nastavka paketa (nakon ovog polja) u bajtima i ima vrednost $4 \cdot n + 1$, gde je *n* broj IP adresa u RREP paketu. Polje *Last Hop External* (L) indicira da je poslednji hop RREP paketa (link od *Address* [*n*-1] do *Address* [*n*]) zapravo proizvoljna putanja u mreži izvan DSR mreže. Tačan put izvan DSR mreže nije prikazan u RREP paketu. Polje *Reserved* mora biti poslato sa vrednošću 0 i ignorisano na prijemu. Izvorna putanja (kojom je primljen RREQ) se vraća pomoću RREP paketa. Putanja označava sekvencu hopova, polazeći od izvornog čvora RREQ paketa (navedenog u polju *Destination Address*), i dalje preko međučvorova sa adresama označenim u poljima od *Address [1]* do *Address [n]*. Broj adresa međučvorova (*n*) može se odrediti na osnovu vrednosti polja *Opt Data Len* (*n* = (*Opt Data Len* - 1) / 4).

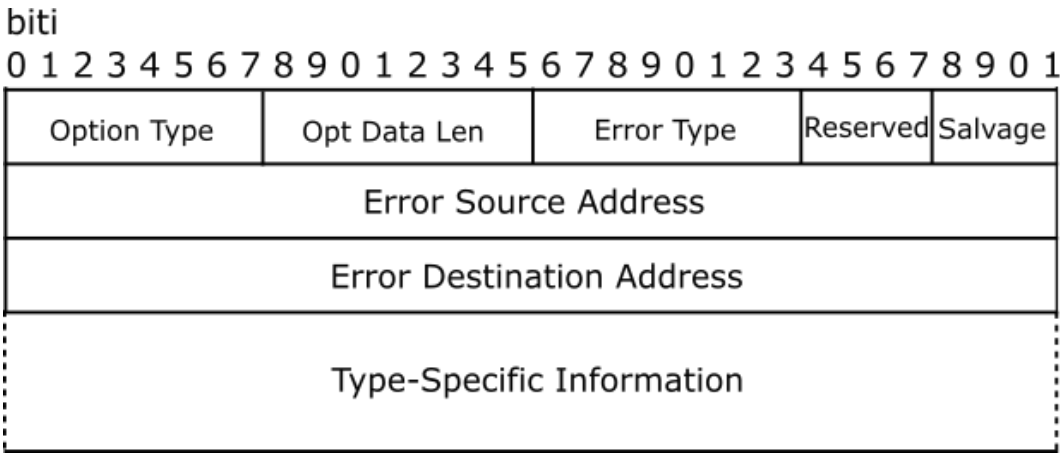
Kroz razmenu RREQ i RREP kontrolnih paketa u DSR protokolu pamti se niz adresa međučvorova kroz koje je paket prošao (za razliku od AODV protokola kod kog se pamti i prenosi samo adresa sledećeg čvora na putanji). Na početku procedure, izvorni čvor nema nijednu putanju do odredišnog čvora u svojoj tabeli rutiranja. Kako bi pronašao putanju do odredišta, izvorni čvor šalje RREQ pakete svim čvorovima u mreži. Ukoliko čvor koji primi RREQ, a nije odredišni čvor, na osnovu polja *Identification* zaključi da je RREQ već ranije stigao u taj čvor, paket se briše. Ovaj mehanizam sprečava kruženje paketa i smanjuje nepotrebnu distribuciju kontrolnih paketa kroz mrežu. U suprotnom, čvor koji primi RREQ upisuje u svoju tabelu rutiranja putanje do svih ostalih čvorova koje je RREQ do tada obišao, čime omogućava da se putanja dinamički kreira dok paket putuje prema odredištu. Zatim, čvor povećava broj hopova za jedan i proverava da li u svojoj tabeli rutiranja ima putanju do odredišnog čvora. Ukoliko putanja postoji, čvor formira putanju od

izvornog do odredišnog čvora na osnovu podataka iz RREQ paketa i svoje tabele rutiranja. Nakon toga ovu putanju šalje izvornom čvoru korišćenjem RREP paketa. Ukoliko čvor koji je primio RREQ nema podatak o putanji do odredišta, on dodaje svoju IP adresu u RREQ paket i proverava da li je paket prošao kroz maksimalan dozvoljen broj čvorova (*hop limit*). Svaki čvor koji primi paket smanjuje *hop limit* za jedan, sve dok ne postane nula, kada paket prestaje da se prosleđuje dalje. Ako *hop limit* nije nula, čvor prosleđuje RREQ ka svim ostalim čvorovima.

U odredišni čvor stiže više RREQ paketa, svaki sa jednom upisanom putanjom od izvornog čvora. Za svaki od primljenih RREQ paketa, odredišni čvor generiše RREP paket u koji upisuje kopiju formirane putanje iz RREQ paketa koji je primio. RREP paket se šalje ka izvornom čvoru putanjom upisanom u odgovarajući RREQ paket. Izvorni čvor prima sve pristigle RREP pakete, upisuje ih u svoju tabelu rutiranja i zatim bira putanju sa najmanjim brojem hopova kojom će slati pakete do odredišnog čvora. Ako se putanja iz tabele rutiranja ne koristi više od 300 sekundi (*timeout* vreme), ona se briše.

2.3.2. Procedura održavanja putanje kod DSR protokola

Mehanizam održavanja putanje u DSR protokolu se značajno razlikuje od onog kod AODV protokola i odvija se tako što izvorni čvor šalje RREQ paket po nekoj putanji kako bi proverio da li je ona još uvek aktivna. Ukoliko je putanja u prekidu, čvor koji to otkrije šalje RERR paket do izvornog čvora, koristeći putanju kojom je RREQ paket stigao. Izvorni čvor u tom slučaju proverava da li ima neku drugu ispravnu putanju do odredišnog čvora u svojoj tabeli rutiranja. Ako postoji, nastavlja da šalje pakete tom putanjom. Ako nijedna ispravna putanja nije dostupna, izvorni čvor pokreće ponovno otkrivanje putanje. Na slici 2.12 je prikazan izgled RERR paketa kod DSR protokola. Svaki RERR paket se sastoji od sledećih polja: *Option Type, Opt Data Len, Error Type, Reserved, Salvage, Error Source Address, Error Destination Address* i *Type-Specific Information*.



Slika 2.12. Struktura RERR paketa kod DSR protokola

Polje *Option Type* ima istu svrhu kao kod RREQ i RREP paketa (za RERR paket ima vrednost 3). U polju *Opt Data Len* označena je dužina paketa nakon ovog polja. Ovo polje mora biti postavljeno na vrednost 10 + veličina svih prisutnih tipski specifičnih informacija (*Type-Specific Information*) u RERR paketu. Dalje ekstenzije formata RERR paketa mogu takođe biti uključene nakon dela koji sadrži *Type-Specific Information.* Prisustvo takvih ekstenzija biće takođe označeno u polju *Opt Data Len*. Kada je *Opt Data Len* veće od onog koje je potrebno za fiksni deo RERR plus neophodne *Type-Specific Information*, preostali okteti se tumače kao ekstenzije. Polje *Error Type* označava tip greške. Trenutno su definisane sledeće vrednosti greški: 1 = nedostupan čvor (NODE_ UNREACHABLE), 2 = nije podržano stanje protoka (FLOW_STATE_NOT_SUPPORTED) i 3 = opcija nije podržana (OPTION_NOT_SUPPORTED). Druge vrednosti polja *Error Type* su

rezervisane za buduću upotrebu. Polje *Reserved* je rezervisano i mora biti poslato na vrednost 0 i ignorisano na prijemu. Polje *Salvage* je četvorobitni ceo broj, kopiran iz polja *Salvage* u DSR *Source Route* opciji paketa koji je trigerovao slanje RERR paketa. Polje *Error Source Address* sadrži adresu čvora od kog potiče RERR (npr. čvor koji je pokušao da prosledi paket i otkrio prekid linka). Polje *Error Destination Address* sadrži adresu čvora kom se mora isporučiti RERR paket. Polje *Type-Specific Information* sadrži specifične informacije za *Error Type* ovog RERR paketa.

Podaci iz tabela rutiranja u DSR protokolu se brišu nakon 300 sekundi, što je znatno ređe u poređenju sa AODV protokolom, gde se podaci brišu nakon 3 sekunde. Takođe, kod DSR protokola nema periodičnog slanja *Hello* kontrolnih paketa. Ovi faktori dovode do značajno manjeg overheda (*overhead*), tj. dodatnih kontrolnih paketa koje razmenjuju čvorovi radi održavanja i ažuriranja tabela rutiranja, u poređenju sa AODV protokolom. Međutim, kod mreža sa velikim brojem čvorova RREQ i RREP paketi mogu biti mnogo veći kod DSR protokola (jer sadrže adrese svih čvorova na putanji od izvora do odredišta), tako da samim tim overhed značajno raste. Takođe, AODV protokol ima bolje performanse u visoko dinamičkim mrežama jer brže otkriva prekide linkova, omogućavajući brže reagovanje na promene u mrežnoj topologiji. Iz ovih razloga se AODV protokol u praksi više koristi.

2.4. DSDV protokol

DSDV protokol spada u proaktivne protokole rutiranja, što znači da svaki čvor u mreži u svakom trenutku ima informaciju o bar jednoj putanji do svih drugih čvorova u mreži. Ovaj protokol se temelji na Belman-Fordovom algoritmu, koji se koristi za izračunavanje najkraće putanje od jednog čvora ka svim drugim čvorovima u težinskom grafu. Tradicionalni *Distance-Vector* (DV) algoritam je klasičan distribuirani Belman-Fordov algoritam. Kod DV algoritma svaki čvor *i* održava skup rastojanja $\{d_{ij}^{x}\}$, za svako odredište *x*, gde *j* služi za rangiranje čvorova koji su susedi čvora *i*. Čvor *i* tretira susedni čvor *k* kao sledeći hop ka odredišnom čvoru *x* ako je $d_{ik}^{x} = min_j\{d_{ij}^{x}\}$. Sukcesivnim izborom sledećeg hopa na ovaj način, dobija se najkraća putanja do čvora *x*. Da bi procene udaljenosti bile ažurne, svaki čvor nadgleda svoje izlazne linkove ka drugim čvorovima i periodično emituje, prema svakom susedu, trenutnu procenu najkraćeg rastojanja do svakog drugog čvora u mreži.

Problem je što ovaj algoritam može prouzrokovati formiranje kratkotrajnih i dugotrajnih petlji. Primarni uzrok formiranja petlji je taj što čvorovi biraju sledeći hop na potpuno distribuiran način, zasnovan na informacijama koje mogu biti i zastarele, samim tim i netačne. Ovaj problem se uglavnom prevazilazi tako što svi čvorovi u mreži učestvuju u nekom obliku međučvornog koordinacionog protokola. Ovakvi mehanizmi koordinacije mogu biti efikasni kada su topološke promene retke, međutim ukoliko dolazi do čestih promena u mrežnoj topologiji, klasični DV protokoli nisu efikasni. Prilagođavanjem tradicionalnih DV protokola dinamičkom okruženju nastao je DSDV protokol, koji rešava problem brzih promena u topologiji, bez potrebe za komplikovanim koordinacionim protokolom.

Kod DSDV protokola, paketi se prenose između čvorova u mreži na osnovu podataka iz tabela rutiranja, koje se čuvaju u svakom čvoru mreže. Svaka tabela rutiranja sadrži sva raspoloživa odredišta, kao i broj hopova do svakog od njih. Svaka stavka u tabeli rutiranja označena je sekvencijalnim brojem, a sekvencijalni brojevi potiču od odredišnog čvora. Da bi se u dinamičnoj i promenljivoj topologiji održala konzistentnost tabela rutiranja, svaki čvor periodično ažurira podatke u svojoj tabeli rutiranja, a ukoliko su dostupne nove informacije o putanjama onda odmah ažurira svoju tabelu rutiranja. Na osnovu ovih podataka čvorovi imaju informaciju o tome koji čvor je dostupan preko bilo kog susednog čvora, kao i o tome koji broj hopova je potreban da bi se stiglo do odredišnog čvora.

DSDV protokol zahteva od svakog mobilnog čvora da obaveštava sve susedne čvorove u određenim trenucima o svojoj tabeli rutiranja (na primer, emitovanjem podataka koje je uneo u svoju tabelu rutiranja). Podaci u tabelama rutiranja se mogu menjati prilično dinamično, tako da ovo oglašavanje mora da bude dovoljno često da se osigura da svaki mobilni čvor može gotovo uvek da pronađe sve ostale čvorove u mreži. Osim toga, svaki mobilni čvor prihvata prenos paketa podataka do drugih čvorova, ukoliko se to od njega zahteva. Svi čvorovi koji međusobno komuniciraju sa ciljem da kreiraju putanju za podatke koje će kasnije razmenjivati, povremeno emituju potrebne informacije, na primer jednom u nekoliko sekundi. Informacije koje emituje svaki mobilni čvor sadrže novi sekvencijalni broj, IP adresu odredišta, broj hopova potrebnih za stizanje do odredišta i sekvencijalni broj dobijenih informacija koje se odnose na to odredište, kao što je originalno označeno od strane odredišta. Jedan od najvažnijih parametara koji treba odrediti je vreme između emitovanja paketa sa informacijama o rutiranju. Međutim, kada mobilni čvor primi neku novu ili značajno izmenjenu informaciju o nekoj putanji, ta informacija će se odmah preneti. Ovo omogućava najbrže moguće širenje informacije o putanji ka svim kooperativnim mobilnim čvorovima.

Kretanje mobilnih čvorova može prouzrokovati prekide veza između njih. Prekinuta veza se označava beskonačnom metrikom (tj. bilo kojom vrednošću većom od maksimalno dozvoljene metrike). Kada je veza sa sledećim hopom prekinuta, bilo kojoj putanji kroz taj hop se odmah dodeljuje beskonačna metrika i novi (ažurirani) sekvencijalni broj. Pošto se ovo karakteriše kao značajna promena putanje, takve modifikovane putanje se odmah oglašavaju putem emitovanja informacionih paketa o rutiranju. Formiranje informacije o prekinutoj vezi je jedina situacija kada sekvencijalni broj generiše bilo koji mobilni čvor, a da nije odredišni. Sekvencijalni brojevi definisani od strane izvornog mobilnog čvora definišu se da budu parni brojevi, a sekvencijalni brojevi generisani da označavaju beskonačne metrike su neparni brojevi. Na ovaj način će realni sekvencijlani brojevi zameniti beskonačnu metriku.

U slučaju veoma velike mreže mobilnih čvorova, neophodno je izvršiti prilagođavanje emitovanja informacionih paketa za rutiranje. Da bi se smanjila količina informacija koje se prenose u ovim paketima, definisana su dva tipa emitovanja. Prvi tip se naziva *full dump* i tu se prenose sve raspoložive informacije o rutiranju. Drugi tip sadrži samo informacije koje su promenjene od poslednjeg *full dump* emitovanja i ovaj tip se naziva *incremental* emitovanje. Prema dizajnu, *incremental* emitovanja informacionih paketa treba da se uklope u jednu jedinicu podataka koja se koristi u mrežnim protokolima (*Network Protocol Data Unit*, NPDU). *Full dump* emitovanje će najverovatnije zahtevati više NPDU jedinica, čak i za relativno male mreže mobilnih čvorova. Informacije ovog tipa se prenose relativno retko, kada se ne događa pomeranje mobilnih čvorova. Očekuje se da mobilni čvorovi primene odgovarajuće metode za određivanje stepena važnosti promena putanja, da bi se odredilo koje informacije o promenama putanja su dovoljno značajne da bi bile poslate unutar svakog *incremental* emitovanja.

Kada mobilni čvor primi nove informacije o rutiranju (obično u *incremental* emitovanju), te informacije se upoređuju sa informacijama koje su već dostupne iz prethodnih informacionih paketa o rutiranju. Za slanje paketa se koristi svaka putanja sa novijim sekvencijalnim brojem. Putanje sa starijim sekvencijalnim brojevima se odbacuju. Putanja sa istim sekvencijalnim brojem kao prethodna, bira se ako ima bolju metriku (tada se postojeća putanja odbacuje) ili se čuva kao manje poželjna. Metrike za putanje koje su izabrane iz novih informacija o rutiranju se sve povećavaju za jedan hop.

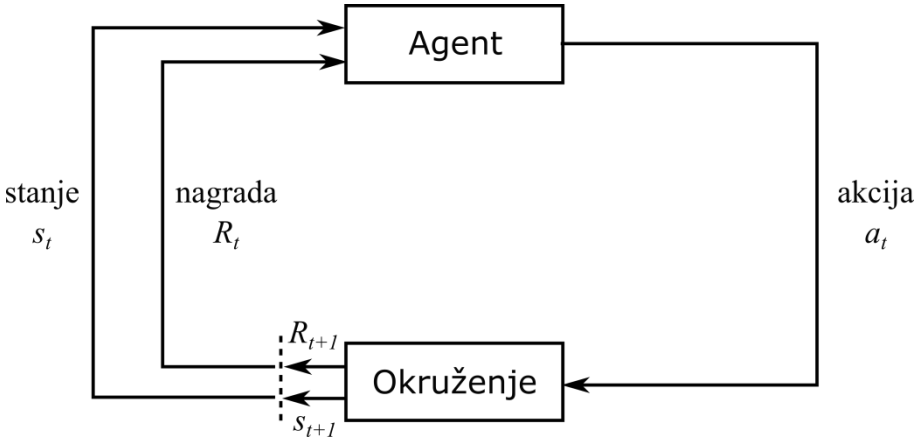
Emitovanje informacija o rutiranju od strane različitih mobilnih čvorova treba donekle smatrati asinhronim događajima, iako se očekuje određena pravilnost. Kod takvog skupa nezavisno prenesenih informacija, može doći do određenih fluktuacija tokom procedura ažuriranja putanja. Moglo bi se ispostaviti da jedan mobilni čvor prima nove informacije o rutiranju u obliku koji će

prouzrokovati stalno menjanje putanje od jednog čvora do drugog, čak i kada se odredišni čvor nije pomerao. Ovo se dešava jer postoje dva načina za određivanje nove putanje. Mogu se izabrati putanje koje imaju noviji sekvencijalni broj ili one koje imaju bolju metriku. Mobilni čvor može uvek primati dve putanje do istog odredišta, svaku narednu sa novijim sekvencijalnim brojem, jednu za drugom (od različitih susednih čvorova), ali uvek može birati prvo putanju sa lošijom metrikom.

Jedno od mogućih rešenja ovog problema je odložiti oglašavanje takvih putanja, kada mobilni čvor može utvrditi da će se putanja sa boljom metrikom uskoro pojaviti. Putanja sa novijim sekvencijalnim brojem mora biti dostupna za korišćenje, ali ne mora biti odmah prosleđena drugim čvorovima, osim ako nije putanja do odredišta koje je ranije bilo nedostupno. Zbog toga svaki mobilni čvor ima dve tabele rutiranja: prva će se koristiti prilikom prosleđivanja paketa (tabela prosleđivanja), a druga služi za oglašavanje putem *incremental* informacionih paketa o rutiranju (tabela za oglašavanje). Da bi se utvrdila verovatnoća neposrednog dolaska informacija o rutiranju koje imaju bolju metričku vrednost, mobilni čvor mora da čuva istoriju ponderisane prosečne vremenske razlike fluktuacija putanja do određenog odredišta, dok se ne dobije putanja sa najboljom metrikom. Takav postupak bi trebao da omogući da se predvidi koliko dugo će se čekati pre oglašavanja novih putanja.

3. PREGLED PROTOKOLA RUTIRANJA BAZIRANIH NA RL ZA DINAMIČKE WANET MREŽE

Kako bi se prevazišli nedostaci tradicionalnih protokola rutiranja u okruženjima sklonim čestim promenama i unapredile ukupne performanse dinamičkih WANET mreža, sve češće se predlaže primena ML u protokolima rutiranja za ove mreže. Najčešće korišćeni tip ML u protokolima rutiranja za dinamičke WANET mreže je RL. Ovaj tip učenja omogućava prikupljanje povratnih informacija od okruženja, kroz stalnu interakciju sa okruženjem, na osnovu kojih je moguće prilagoditi proces rutiranja trenutnom stanju okruženja. U opštem slučaju, učesnici RL procesa su agent učenja (odnosno donosilac odluka) i okruženje (sve ono što okružuje agenta). Na slici 3.1 predstavljen je šematski prikaz interakcije agenta učenja i okruženja u RL procesu. Stanje okruženja u određenom trenutku t može se označiti sa s_t, a akcija koju preduzima agent prema okruženju u tom trenutku može se označiti sa a_t. Podrazumeva se da su skupovi mogućih stanja i akcija konačni. Nakon što agent preduzme akciju a_t, stanje okruženja se menja i u sledećem trenutku postaje s_{t+1}. Okruženje šalje povratnu informaciju agentu koja sadrži nagradu R_{t+1} za preduzetu akciju a_t i novo stanje okruženja s_{t+1}. Nagrada može biti i negativna (ukoliko akcija ne doprinosti unapređenju performansi okruženja) i u tom slučaju predstavlja kaznu za loš izbor akcije. Ovim putem agent potkrepljuje odluke o budućim akcijama znanjem o reakciji okruženja na prethodne akcije.

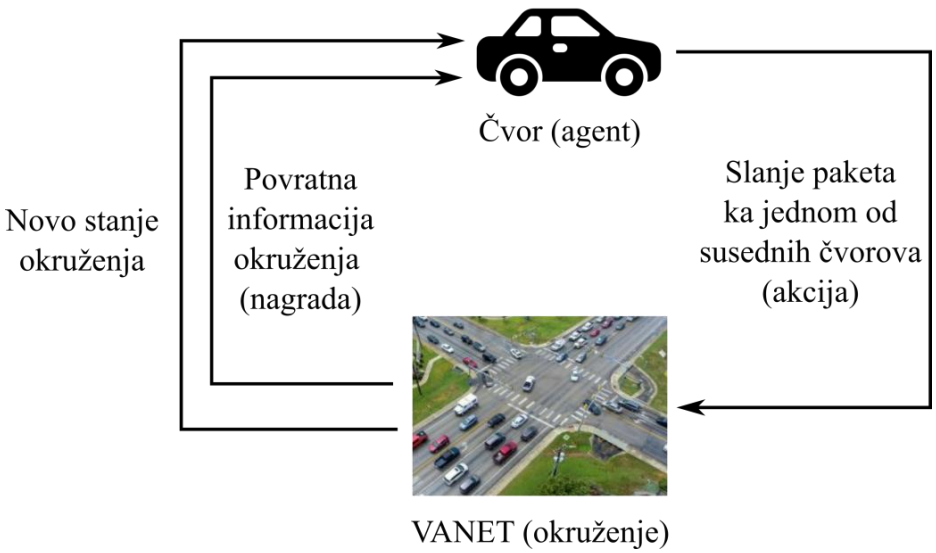


Slika 3.1. Interakcija agenta učenja i okruženja u RL procesu

Cilj agenta učenja je da maksimizira nagrade kroz izbor odgovarajućih akcija, a samim tim da unapredi opšte performanse okruženja. Do toga dolazi balansirajući između eksploatacije stečenog znanja i istraživanja okruženja kako bi stekao dodatno znanje. Eksploatacija znanja podrazumeva preduzimanje akcije koja na osnovu prethodnog iskustva donosi najveću nagradu, dok istraživanje okruženja zahteva preduzimanje nasumičnih akcija kako bi se steklo dodatno znanje. Iz tog razloga, moguće je definisati politiku izbora akcija tako da agent u određenom procentu slučajeva bira najkorisniju akciju, dok u preostalim slučajevima proizvoljno bira akciju koju će da preduzme. Jedna od takvih politika je *ε-greedy*, kod koje agent sa verovatnoćom ε (koja je u opsegu 0-100%)

preduzima akciju sa najvećom Q-vrednošću, dok sa verovatnoćom (100% - ε) preduzima slučajno izabranu akciju iz skupa mogućih akcija.

Proces učenja potkrepljivanjem u jednoj VANET mreži moguće je modelirati na sledeći način. Čvor (vozilo) koji želi da pošalje podatke ka određenom odredištu predstavlja agenta učenja, dok svi ostali čvorovi u mreži predstavljaju njegovo okruženje. Izbor putanje kojom će slati podatke ka odredištu predstavlja akciju koju agent može da preduzme, a ona je ekvivalentna izboru jednog od susednih čvorova da bude sledeći čvor (hop) na putanji. Skup suseda je uvek ograničen, pa je samim tim i skup mogućih akcija konačan. Nakon slanja paketa, susedni čvorovi (okruženje) obaveštavaju agenta učenja o promeni stanja okruženja i o nagradi (ili kazni) za preduzetu akciju. Na nagradu mogu uticati razni uticajni faktori, u zavisnosti od cilja optimizacije mreže, što je detaljnije opisano u nastavku poglavlja. Ilustracija RL procesa u jednoj VANET mreži prikazana je na slici 3.2.



Slika 3.2. Ilustracija RL procesa u dinamičkoj WANET mreži

U nastavku poglavlja najpre su opisani najznačajniji RL algoritmi koji se koriste u protokolima rutiranja za dinamičke WANET mreže. Kako bi se što bolje sagledali trenutni rezultati primene RL za unapređenje performansi ovih mreža, izvršena je klasifikacija protokola baziranih na RL za dinamičke WANET mreže i predstavljena je zastupljenost pojedinih tipova WANET mreža za koje su protokoli razvijeni, tipova primenjene RL tehnike i drugih tehnika u protokolima rutiranja. Nakon toga su detaljnije opisani pojedini predstavnici izvedenih klasa, kako bi se jasnije razumeo princip funkcionisanja ovih protokola. Najpre su opisani protokoli za VANET mreže, a nakon toga i protokoli za FANET mreže.

Nakon klasifikacije, izvršeno je poređenje protokola rutiranja na osnovu korišćenih uticajnih faktora u RL procesu, posmatranih mrežnih performansi koje su optimizovane i korišćenog mrežnog simulatora za evaluaciju i testiranje protokola. U protokolima su korišćeni brojni uticajni faktori u RL procesu i posmatrane su različite mrežne performanse, pa je izvršeno njihovo grupisanje u odgovarajuće tipove kako bi analiza bila što jasnija. Protokoli su najpre pojedinačno analizirani, a zatim je predstavljena zastupljenost pojedinih tipova uticajnih faktora, tipova posmatranih performansi i korišćenih simulacionih alata. Odvojeno su analizirani protokoli za VANET i FANET mreže. Zatim je izvršena analiza ovog pregleda protokola i izvedeni su odgovarajući zaključci na osnovu izvršene klasifikacije i poređenja. Na kraju poglavlja, na osnovu sveobuhvatnog pregleda aktuelne literature, izabrani su i detaljno opisani reprezentativni predstavnici RL baziranih

protokola rutiranja, koji će u nastavku disertacije biti iskorišćeni za poređenje sa novim Q-DRAV protokolom rutiranja.

3.1. Najznačajniji RL algoritmi za dinamičke WANET mreže

U protokolima rutiranja za dinamičke WANET mreže koristi se nekoliko različitih tipova RL algoritama. Najčešće korišćeni RL algoritam je QL, koji podrazumeva računanje Q-vrednosti za svaku potencijalnu akciju a_t, koju agent učenja može da preduzme dok je okruženje u stanju s_t. Ove vrednosti agent čuva u svojoj Q-tabeli i na osnovu njih bira narednu akciju koju će preduzeti. U opštem slučaju, za uređeni par (s_t, a_t) agent računa Q-vrednost preko sledeće relacije:

$$Q(s_t, a_t) \leftarrow (1 - \alpha) \cdot Q(s_t, a_t) + \alpha \cdot (R_{t+1} + \gamma \cdot max_a Q(s_{t+1}, a)). \quad (3.1)$$

U ovoj relaciji R_{t+1} označava nagradu koju je agent dobio od okruženja za preduzetu akciju a_t, a s_{t+1} predstavlja novo stanje okruženja nakon preduzete akcije a_t. Parametar α označava stepen učenja (*learning rate*) koji može uzeti vrednost u opsegu [0,1] i definiše kojom brzinom će agent učiti od okruženja. Drugim rečima, na ovaj način se definiše brzina ažuriranja Q-vrednosti. Treba voditi računa da agent ne uči prebrzo, kako ne bi svaka promena u okruženju drastično uticala na Q-vrednosti i samim tim dovela do njihove nestabilnosti. Sa druge strane, agent ne treba da uči ni presporo, kako ne bi došlo do kasnog prilagođavanja promenama u okruženju. Parametar γ predstavlja diskontni faktor (*discount factor*), koji određuje važnost budućih nagrada i takođe može uzeti vrednost u opsegu [0,1]. Najzad, vrednost $max_a Q(s_{t+1}, a)$ predstavlja maksimalnu Q-vrednost do koje agent može doći preduzimanjem jedne od dostupnih akcija a, kada je okruženje u narednom stanju s_{t+1}. Uglavnom je definisano da Q-vrednost takođe može biti u opsegu [0,1], pri čemu se kod izbora optimalne putanje prednost daje putanjama sa većim Q-vrednostima (ako se ne vrši istraživanje okruženja).

Prednost QL algoritma je njegova jednostavnost, mali procesorski zahtevi i laka implementacija. On je pogodan za primenu u manjim WANET mrežama, koje karakteriše manje potencijalnih stanja, agenata učenja i akcija koje agenti mogu preduzeti. Međutim, kod okruženja sa većim brojem potencijalnih stanja i učesnika u procesu učenja, brzina konvergencije QL algoritma može biti ispod zadovoljavajućeg nivoa. U tim slučajevima se preporučuje primena nešto složenijeg DRL algoritma, gde se za računanje Q-vrednosti koristi duboka Q-mreža, koja predstavlja kombinaciju RL i dubokih neuronskih mreža. Na primer, konvolucione neuronske mreže (*Convolutional Neural Networks*, CNN) mogu se koristiti kao komponente RL agenata, omogućavajući im da direktno uče iz višedimenzionalnih ulaza. Ulaz u mrežu može biti skup parametara koji definiše trenutno stanje okruženja, a izlaz optimalna akcija koju agent treba da preduzme u određenom trenutku. Generalno, DRL se zasniva na treniranju dubokih neuronskih mreža kako bi aproksimirale optimalnu politiku izbora akcija agenta učenja. Za primenu algoritama baziranih na DRL, agenti moraju biti opremljeni procesorima visokih performansi, koji mogu da podrže ovaj veoma zahtevan tip učenja.

Dalje unapređenje performansi i povećanje stabilnosti RL moguće je ostvariti pomoću DDRL algoritma. Ovaj algoritam predstavlja unapređenje DRL, koje je ostvareno uvođenjem duelne arhitekture za određivanje optimalnih Q-vrednosti. Ovaj koncept koristi duelne duboke Q-mreže (*Dueling Deep Q-Networks*, DDQNs) za određivanje optimalnih Q-vrednosti. Suština DDQN pristupa leži u činjenici da nije neophodno izračunavati Q-vrednosti za svaku dostupnu akciju u svakom diskretnom trenutku *t*. Umesto toga, arhitektura DDQN mreže razdvaja RL proces na dve ključne komponente: funkcija vrednosti stanja (*value function*) i funkcija prednosti akcije (*advantage function*). Funkcija vrednosti stanja definiše koliko je određeno stanje dobro samo po sebi, dok funkcija prednosti akcije određuje koliko je određena akcija bolja ili lošija od drugih akcija koje agent može da preduzme u određenom stanju. Ova podela omogućava efikasniju procenu vrednosti akcija, jer mreža posebno proračunava koliko je samo stanje korisno, nezavisno

od konkretnih akcija, dok prednost akcije modelira dodatni doprinos određene akcije u datom stanju. Na taj način, DDQN smanjuje prekomerno oslanjanje na precizno ocenjivanje pojedinačnih akcija i poboljšava stabilnost procesa učenja. Kombinovanjem funkcije vrednosti stanja i funkcije prednosti akcije, agent određuje optimalnu akciju koju treba da preduzme u određenom stanju.

Još jedan RL algoritam koji se koristi u protokolima rutiranja za dinamičke WANET mreže predstavlja SARSA algoritam, koji je dosta sličan QL algoritmu. SARSA uči politiku izbora akcija na osnovu akcija koje zaista sprovodi agent, prateći politiku koju trenutno koristi (npr. *ε-greedy* politiku). Kod ovog algoritma Q-vrednosti se računaju preko sledeće relacije:

$$Q(s_t, a_t) \leftarrow (1-\alpha) \cdot Q(s_t, a_t) + \alpha \cdot (R_{t+1} + \gamma \cdot Q(s_{t+1}, a_{t+1})). \tag{3.2}$$

Jedina razlika u odnosu na QL algoritam je zamena vrednosti $max_a Q(s_{t+1},a)$ sa $Q(s_{t+1},a_{t+1})$, gde a_{t+1} predstavlja akciju koju agent preduzima u trenutku t+1, dok je okruženje u stanju s_{t+1}. Algoritam je dobio naziv upravo po parametrima koji se koriste kod proračuna Q-vrednosti (*State-Action-Reward-State-Action*), a to su stanje okruženja u trenutku t (s_t), akcija koju agent preduzima dok je okruženje u tom stanju (a_t), nagrada za akciju a_t (R_{t+1}), novo stanje nakon preduzete akcije a_t (s_{t+1}) i akcija koju agent preduzima kad je okruženje u novom stanju (a_{t+1}). Budući da SARSA uzima u obzir politiku koju trenutno koristi agent, može biti stabilniji u dinamičkim okruženjima gde promene u okruženju utiču na performanse agenta. Sporiji je od QL algoritma u nekim slučajevima, jer uči prema trenutnoj politici izbora akcija, a ne prema optimalnoj.

Manje zastupljen u protokolima rutiranja, ali značajan i interesantan zbog svoje specifičnosti, jeste MBRL algoritam. Za razliku od *model-free* algoritama, MBRL koristi unapred kreirani model okruženja za određivanje optimalne politike rutiranja. Na taj način simulira potencijalne akcije i njihove posledice, pre nego što ih sprovede u stvarnom okruženju. Ovo omogućava brže pronalaženje optimalnih akcija i izbegavanje rizičnih situacija, ali takođe zahteva složen proces izrade modela, uključujući kreiranje dinamičkog modela prelaska iz stanja u stanje (*dynamic state transition model*), a u nekim slučajevima i modela nagrada. Pošto ima mogućnost simulacije prelaska iz stanja u stanje, MBRL zahteva manje interakcija agenta učenja sa okruženjem. MBRL algoritmi su posebno efikasni kada je okruženje strukturalno predvidivo, jer dobro naučeni modeli mogu brzo generisati realistične prelaze iz stanja u stanje.

3.2. Klasifikacija protokola rutiranja baziranih na RL za dinamičke WANET mreže

Poslednjih godina razvijen je veliki broj novih protokola na bazi RL za dinamičke WANET mreže. Kako bi se doprinelo daljem razvoju ovih protokola, korisno je najpre izvršiti detaljnu analizu aktuelnih rešenja. Zajedničko za sve ove protokole je korišćenje neke od RL tehnika prilikom izbora optimalne putanje za slanje podataka. Međutim, postoje značajne razlike između protokola na koje bi trebalo ukazati (Jevtić i ostali, 2021; Bugarčić i ostali, 2022). Prvenstveno treba razlikovati protokole za VANET i protokole za FANET mreže, zbog drugačijih zahteva koje treba da zadovolje. Zajednička karakteristika ovih mreža je velika dinamičnost mrežnih čvorova i česte promene u mrežnoj topologiji, ali treba uzeti u obzir i brojne njihove razlike kako bi proces rutiranja dao što bolje mrežne performanse. Kod VANET mreža čvorove predstavljaju vozila koja se kreću u dvodimenzionalnom prostoru i moraju da prate saobraćajna ograničenja u vidu ulica, saobraćajne signalizacije, ograničenja brzine, itd. Sa druge strane, mrežni čvorovi kod FANET mreža su bespilotne letelice koje se kreću u trodimenzionalnom prostoru sa mnogo većom slobodom kretanja. To znači da je skup mogućih putanja kod ovih mreža praktično neograničen, a takođe i brzine kretanja čvorova mogu biti dosta veće od onih koje su karakteristične za VANET mreže. Takođe, veliki izazov za FANET mreže je ograničen izvor napajanja energijom i mala

gustina čvorova, što može značajno otežati održavanje mrežnih linkova. Zato se i uticajni faktori u procesu određivanja optimalne putanje prilično razlikuju za ova dva tipa mreža.

Pored tipa mreže za koju su dizajnirani, protokoli se takođe razlikuju po tipu RL koji se koristi u procesu rutiranja. U ove tipove spadaju *Q-Learning* (QL), *Deep Reinforcement Learning* (DRL), *Dueling Deep Reinforcement Learning* (DDRL), SARSA i *Model-Based* RL (MBRL) algoritmi. U nekim protokolima su uz RL, zarad dodatnog unapređenja mrežnih performansi, korišćene i druge tehnike u procesu rutiranja paketa, kao što su *Software-Defined Networking* (SDN), *blockchain* (BC) i *fuzzy* logika (FL). Zbog toga i njihovu eventualnu primenu treba uzeti u obzir prilikom klasifikacije protokola.

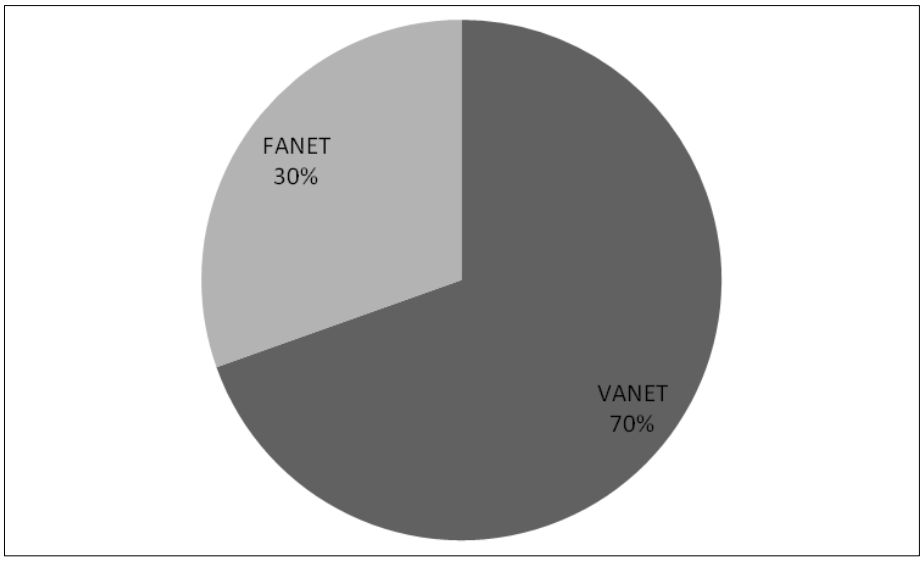
U tabeli 3.1 predstavljena je klasifikacija RL baziranih protokola rutiranja za dinamičke WANET mreže, na osnovu tipa mreže za koju je protokol dizajniran, tipa RL koji se primenjuje u protokolu i eventualne primene još neke tehnike u kombinaciji sa RL u procesu rutiranja. Imajući u vidu značajan napredak u primeni RL tehnike u protokolima rutiranja tokom poslednjih nekoliko godina, u ovoj klasifikaciji su uzeti u obzir savremeniji protokoli (koji su publikovani od 2018. godine pa nadalje).

Tabela 3.1. Klasifikacija protokola rutiranja baziranih na RL za dinamičke WANET mreže

Kat.	*Referenca*	*Tip mreže*	*Tip RL*					*Druge tehnike*		
			QL	*DRL*	*DDRL*	*SARSA*	*MBRL*	*SDN*	*BC*	*FL*
1.	(Ji i ostali, 2019; F. Li i ostali, 2019; G. Li i ostali, 2020; X. Liu i ostali, 2023; Lolai i ostali, 2022; Luo i ostali, 2022; Roh i ostali, 2020; Smida i ostali, 2020; C. Wu i ostali, 2019; J. Wu i ostali, 2018; J. Wu i ostali, 2020; X. Yang i ostali, 2020; D. Zhang, Zhang i Liu, 2019)	VANET	✓							
2.	(Nahar i Das, Jun 2020)	VANET	✓					✓		
3.	(Dai i ostali, 2018)	VANET	✓						✓	
4.	(An i ostali, 2018; S. Jiang i ostali, 2021; C. Wu i ostali, 2018; W. Zhang i ostali, 2021;)	VANET	✓							✓
5.	(Saravanan i Ganeshkumar, 2020; Upadhyay i ostali, 2023; Ye i ostali, 2021)	VANET		✓						
6.	(Ahmed i ostali, 2023; Nahar i Das, Nov. 2020; Y. Yang i ostali, 2019; D. Zhang, Yu i Yang, 2018; D. Zhang i ostali, 2020)	VANET		✓				✓		

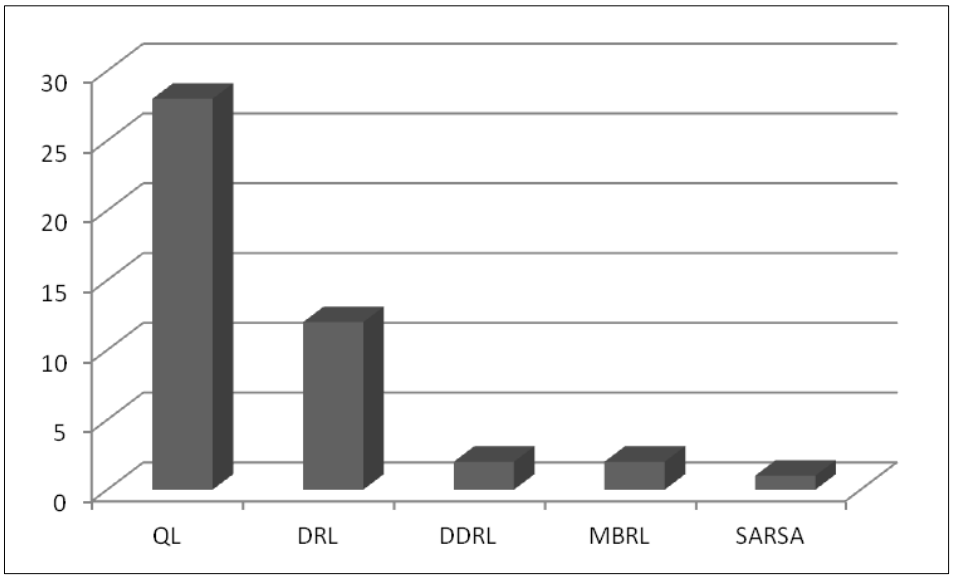
Kat.	*Referenca*	*Tip mreže*	*Tip RL*					*Druge tehnike*		
			QL	*DRL*	*DDRL*	*SARSA*	*MBRL*	*SDN*	*BC*	*FL*
7.	(D. Zhang, Yu, Yang i Tang, 2018)	VANET			✓			✓		
8.	(D. Zhang, Yu i Yang, 2019)	VANET			✓			✓	✓	
9.	(Bi i ostali, 2020)	VANET				✓				
10.	(Jafarzadeh i ostali, 2022)	VANET					✓			✓
11.	(Arafat i Moh, 2021; Da Costa i ostali, 2021; Hosseinzadeh i ostali, 2023; Khan i Yau, 2020; J. Li i Chen, 2020; J. Liu, Wang, He, Jaffres-Runser i ostali, 2020; Mowla i ostali, 2020; Sliwa i ostali, 2021; Zheng i ostali, 2018)	FANET	✓							
12.	(Q. Yang i ostali, 2020)	FANET	✓							✓
13.	(Ayub i ostali, 2022; J. Liu, Wang, He i Hu, 2020; Song i ostali, 2024)	FANET		✓						
14.	(He i ostali, 2020)	FANET		✓						✓

Zastupljenost savremenih RL baziranih protokola rutiranja u VANET i FANET mrežama predstavljena je na slici 3.3. S obzirom da se tek poslednjih godina ubrzano razvijaju FANET mreže (zbog razvoja tehnologije bespilotnih letelica), broj protokola za VANET mreže je i dalje značajno veći (više od dve trećine ukupnog broja protokola).



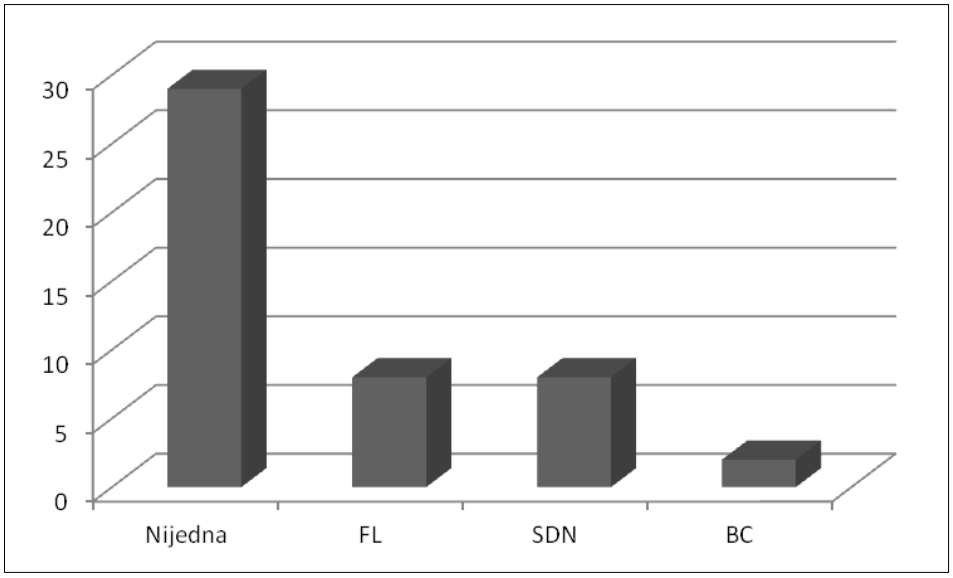
Slika 3.3. Zastupljenost savremenih RL baziranih protokola rutiranja u VANET i FANET mrežama

Na slici 3.4 prikazana je zastupljenost pojedinih tipova RL u savremenim protokolima rutiranja za VANET i FANET mreže. Ubedljivo najzastupljeniji tip je QL, a nakon toga sledi nešto složeniji DRL algoritam. Po dva protokola koriste DDRL i MBRL algoritme, dok je u jednom protokolu rutiranja primenjen SARSA algoritam. Može se zaključiti da većina protokola rutiranja koristi procesorski manje zahtevne algoritme (prvenstveno QL), a da je korišćenje složenijih i procesorski zahtevnih algoritama (DRL, DDRL i MBRL) još uvek manje zastupljeno.



Slika 3.4. Zastupljenost tipova RL u savremenim protokolima rutiranja za VANET i FANET mreže

Na slici 3.5 prikazana je zastupljenost dodatnih tehnika u savremenim protokolima rutiranja za VANET i FANET mreže, koje se koriste u kombinaciji sa RL sa ciljem daljeg unapređenja procesa izbora optimalne putanje za slanje podataka. Može se primetiti da u najvećem broju protokola još uvek nema dodatnih tehnika. U ostalim slučajevima najzastupljenija je primena FL tehnike (koja se koristi recimo za kreiranje modela okruženja) i SDN tehnike (koja podrazumeva uvođenje centralizovane kontrole procesa rutiranja pomoću SDN kontrolera), dok je BC tehnika korišćena u svega dva protokola rutiranja (na primer za povećanje bezbednosti rutiranja).



Slika 3.5. Zastupljenost dodatnih tehnika u savremenim protokolima rutiranja za VANET i FANET mreže

Na osnovu izvršene klasifikacije, može se zaključiti da je većina protokola razvijena za VANET mreže, uz korišćenje QL algoritma i bez korišćenja dodatnih tehnika. U nastavku su detaljnije opisani pojedini protokoli rutiranja iz svake od izvedenih kategorija.

3.3. Opis pojedinih klasnih predstavnika

U tabeli 3.1 protokoli su klasifikovani u 14 kategorija. U ovom odeljku je opisan princip funkcionisanja po jednog protokola iz svake kategorije, sa ciljem boljeg razumevanja principa njihovog funkcionisanja. Prvih 10 kategorija čine protokoli rutiranja za VANET mreže, a zatim slede 4 kategorije protokola rutiranja za FANET mreže. Tim redom su i opisani predstavnici pojedinih kategorija.

3.3.1. Primena QL bez dodatnih tehnika u VANET mrežama

Prvu kategoriju u tabeli 3.1 čine protokoli rutiranja za VANET mreže bazirani na QL algoritmu, koji ne koriste još neku dodatnu tehniku za unapređenje rutiranja. Reprezentativni primer protokola ovog tipa je *Reinforcement learning based hybrid routing* (RHR) algoritam (Ji i ostali, 2019), koji pomaže u rešavanju problema slepih putanja (*blind path*), često prisutnog u rutiranju kod VANET mreža. Ovaj problem se javlja kada određenoj putanji u tabeli rutiranja još nije istekao životni vek, ali je, usled velike mobilnosti čvorova u mreži, sledeći čvor na toj putanji već izašao iz dometa čvora koji šalje pakete. Oslanjanje na samo jednu putanju može dovesti do čestog izbora slepih putanja i samim tim gubitka paketa, a cilj RHR protokola je da ovo spreči. Zbog toga predloženi protokol istovremeno pronalazi više putanja do odredišta (uzimajući u obzir trenutne mrežne uslove) i pokreće QL mehanizam za svaku putanju u tabeli rutiranja, kako bi brzo izabrao novu putanju u slučaju prekida linka na nekoj od korišćenih. Za čuvanje informacija o alternativnim putanjama svaki čvor koristi i održava tabelu rutiranja koja se sastoji iz dva nivoa. Pored osnovne tabele u kojoj su smeštene najbolje trenutne putanje za slanje paketa ka svim odredištima u mreži, čvorovi održavaju posebne podtabele u kojima su smeštene informacije o alternativnim putanjama ka odredištima.

QL algoritam je implementiran u svakom čvoru (koji predstavlja agenta učenja), pri čemu različite selekcije sledećeg hopa (susednog čvora za slanje podataka ka odredištu) predstavljaju specifična stanja, a prijem različitih vrsta paketa usmerenih ka trenutnom sledećem hopu predstavlja odgovarajuće akcije. Za akciju preduzetu u datom stanju, čvorovi dobijaju povratne informacije u vidu nagrade, u zavisnosti od vrste primljenih paketa. Ako čvor primi *broadcast* paket, putanja kojom je paket stigao dobiće negativnu nagradu (odnosno kaznu), jer je velika verovatnoća da putanja nije optimalna. S druge strane, u slučaju prijema *unicast* paketa, putanja dobija pozitivnu nagradu. Nakon toga, čvorovi izračunavaju Q-vrednosti, smeštaju ih u svoje Q-tabele i koristeći ε-*greedy* politiku biraju optimalnu putanju za slanje paketa. Veličina Q-tabele je kontrolisana tako što je ograničen broj potencijalnih putanja koje čvor može da pamti i uvedeno je vreme trajanja putanja (nakon čega se Q-vrednosti za te putanje brišu).

Protokol takođe podrazumeva kontrolisanje opterećenja mreže, tako što proređuje slanje paketa u stabilnijim delovima mreže i stopira slanje paketa ako broj suseda određenog čvora prevazilazi definisani prag. Tako se izbegavaju zagušenja u veoma gustim mrežama. Takođe protokol ograničava da samo čvorovi bliži odredištu (u odnosu na trenutni čvor) mogu biti izabrani za sledeći hop. Na osnovu simulacione analize u NS-3 simulatoru, autori su pokazali da se ovakvim pristupom unapređuju mrežne performanse u pogledu procenta uspešno isporučenih paketa (*Packet Delivery Ratio*, PDR), vremena povratka paketa (*Round Trip Time*, RTT) i overheda.

3.3.2. Primena QL i SDN tehnike u VANET mrežama

Adekvatan predstavnik druge kategorije u tabeli 3.1 je *Adaptive self-learning clustering algorithm with reinforcement routing in SDN-based VANETs* (RL-SDVN) (Nahar i Das, Jun 2020), koji kombinuje primenu QL algoritma i SDN tehnike za klasterovanje i pronalaženje optimalne putanje za prosleđivanje paketa podataka. Glavni cilj RL-SDVN je poboljšanje procesa distribucije poruka (*message dissemination proccess*) i smanjenje prosečnog vremena prenosa podataka. Da bi se omogućilo efikasnije formiranje klastera i pouzdanije rutiranje, predloženi algoritam funkcioniše kroz dve faze. Prvi korak je formiranje klastera i dodeljivanje vozila odgovarajućem klasteru. Vozila periodično šalju kontrolne (*beacon*) poruke svojim susedima, putem kojih razmenjuju sve potrebne informacije za kreiranje klastera. U ove informacije ubrajaju se povezanost sa drugim vozilima, rastojanje između vozila, domet prenosa (*transmission range*) svakog vozila i broj paketa u redu za obradu u određenom vozilu. Vozila sa visokom povezanošću i malim zauzećem reda za obradu paketa biće odabrana za čvorove vođe klastera (*Cluster Heads*, CHs).

Nakon kreiranja klastera, u drugom koraku SDN kontroler (kao agent učenja), uz pomoć CH vozila i geografskih informacija o svakom vozilu, koristeći QL algoritam traži najbolju putanju od izvornog do odredišnog čvora, na osnovu kvaliteta dostupnih putanja. Proces učenja se ponavlja za svaki međučvor koji prosleđuje pakete, sve dok paketi ne stignu do odredišta. U QL procesu, vozila u mreži predstavljaju stanja u kojima agent može biti, dok slanje paketa od jednog vozila do drugog predstavlja moguću akciju. Kada vozilo primi paket, proverava svoju Q-tabelu i ako ima putanju do odredišta, ažurira tabelu, prosleđuje paket i prima pozitivnu nagradu. U suprotnom, odbacuje paket i prima negativnu nagradu. Na vrednost nagrade utiče rastojanje do odredišnog čvora i kašnjenje paketa. Sve neophodne informacije za ažuriranje Q-vrednosti vozila razmenjuju putem *Hello* kontrolnih poruka. Simulacionom analizom u NS-3 simulatoru, pokazano je da predloženi algoritam povećava stabilnost i životni vek klastera, a unapređuje i mrežne performanse u pogledu prosečnog kašnjenja i ostvarenog protoka paketa.

3.3.3. Primena QL i BC tehnike u VANET mrežama

Treća kategorija u tabeli 3.1 se odlikuje primenom QL i BC tehnike, a predstavnik ove kategorije je *Q-learning based action selection strategy* (QLASS) (Dai i ostali, 2018), koji predlaže sigurnosni okvir za podsticanje kooperativnog ponašanja OBU jedinica u VANET mrežama, u cilju zaštite od potencijalnih napada. Okvir je testiran na mreži koja se sastoji od dve vrste čvorova: jedne RSU jedinice i više OBU jedinica. OBU jedinice mogu da pomažu jedne drugima prateći zahteve susednih OBU jedinica, ali takođe mogu biti "sebične" i pokušati da maksimizuju svoje koristi delujući maliciozno i napadajući mrežu kako bi ostvarile neregularnu dobit.

BC tehnika se koristi kako bi se smanjio broj potencijalnih napadača u mreži, na osnovu reputacije svakog mrežnog čvora (vozila). Reputacija je važan parametar koji se deli između čvorova u mreži. Ako OBU jedinica ne učestvuje u napadima, njena reputacija raste, što povećava verovatnoću da će susedne OBU jedinice slediti njene zahteve. Takođe, ova OBU ima veću verovatnoću da bude izabrana za relejnu OBU pri slanju podataka ka odredištu. RSU ima najveću reputaciju, jer nikad ne učestvuje u napadima. Protokol podrazumeva da čvorovi koji prate zahteve izvornih čvorova sa visokom reputacijom ili odbijaju zahteve izvornih čvorova sa niskom reputacijom, takođe dobijaju visoku reputaciju. S druge strane, čvorovi koji napadaju izvorni čvor se kažnjavaju i dodeljuje im se niska reputacija u skladu sa nivoom opasnosti njihove akcije, bez obzira na reputaciju izvornog čvora.

Svaka OBU jedinica koristi QL da izabere optimalnu akciju za postizanje maksimalne koristi, prateći ε-*greedy* politiku. Skup akcija izvornog čvora obuhvata sve ostale čvorove, koji predstavljaju potencijalne relejne čvorove preko kojih se mogu poslati podaci ka odredišnom čvoru.

Akcije relejnog čvora mogu biti lažno predstavljanje (*spoofing*), prisluškivanje (*eavesdropping*), nepoštovanje zahteva ili poštovanje zahteva, dok skup akcija nerelejnog čvora uključuje sve akcije relejnog čvora i ometanje (*jamming*). Stanje okruženja uključuje reputaciju, lokaciju i brzinu čvorova. Na izbor akcije utiče korisnost OBU jedinice od koje dolazi zahtev, koja zavisi od njene reputacije i isplativosti njenih akcija. Autori su koristili sopstveni (*Custom-Made,* CM) simulator kako bi pokazali da ovaj pristup unapređuje mrežne performanse u pogledu PDR, reputacije i korisnosti čvorova u mreži.

3.3.4. Primena QL i FL tehnike u VANET mrežama

Četvrta kategorija u tabeli 3.1 sastoji se od protokola koji se baziraju na primeni QL i FL tehnika u procesu rutiranja. Primer takvog protokola je *Q-learning based adaptive geographic routing approach* (QAGR) (S. Jiang i ostali, 2021), koji zahteva uključivanje bespilotnih letelica u proces rutiranja kako bi se izbegao loš izbor putanje zbog ograničenog dometa komunikacije vozila. Šema rutiranja se sastoji iz vazdušne i zemaljske komponente. Unutar vazdušne komponente, bespilotne letelice kreiraju globalnu putanju koristeći FL i *depth-first-search* (DFS) algoritam (Cormen i ostali, 2009), kako bi se osiguralo da vozila ne šalju pakete u pogrešnom smeru. Na izbor optimalne globalne putanje utiču prosečan broj vozila u određenoj oblasti i prosečna brzina kretanja tih vozila. Informacije o globalnoj putanji bespilotne letelice šalju odgovarajućem vozilu na zemlji kako bi bile iskorišćene kao filter za odbacivanje nesigurnih i opterećenih suseda pri izboru optimalnog sledećeg hopa. Pošto imaju ograničen kapacitet baterije, bespilotne letelice se prvenstveno koriste za određivanje globalne putanje, a samo u slučaju nepovezanosti vozila na zemlji uključuju se u prenos podataka.

Unutar zemaljske komponente, vozila biraju optimalni sledeći hop na osnovu QL, prateći Q-tabelu filtriranu prema globalnoj putanji. QL je modelovan tako da svako stanje uključuje geografsku oblast određenog vozila, udaljenost od vozila do njegovog suseda i broj suseda susednog vozila. Agent učenja može biti bilo koje vozilo u mreži, a skup akcija koje agent može preduzeti uključuje slanje paketa jednom od susednih vozila. Nagrada koju agent dobija za određenu akciju zavisi od jačine primljenog signala (*Received Signal Strength,* RSS), udaljenosti prenosa i broja suseda susednog vozila (koji utiče na broj kolizija između vozila). Izbor odgovarajućih akcija vrši se na osnovu Q-vrednosti, prateći *ε-greedy* politiku. Kada vozilo ne može da pronađe sledeći hop za slanje podataka, proslediće podatke najbližoj bespilotnoj letelici i prenos podataka biće dovršen uz pomoć vazdušne komponente. QAGR poboljšava E2ED, PDR i broj hopova, što je pokazano na osnovu simulacija izvršenih u NS-3 simulatoru.

3.3.5. Primena DRL bez dodatnih tehnika u VANET mrežama

Peta kategorija u tabeli 3.1 uključuje protokole zasnovane na DRL, koji ne koriste dodatne tehnike u procesu rutiranja paketa podataka. Jedan od značajnih protokola iz ove kategorije je *Deep reinforcement learning model for vehicular ad hoc networks* (DRLV) (Saravanan i Ganeshkumar, 2020), u kojem se DRL algoritam koristi za uspostavljanje i odabir najboljih putanja u VANET mreži. Scenario za koji je predložen ovaj model, pored V2V, uključuje V2I komunikaciju, gde određena RSU jedinica pokriva jednu oblast mreže. Celokupna mreža podeljena je na klastere tako da svaki klaster ima svoju gustinu i prosečnu brzinu vozila. Ove dve veličine su međusobno zavisne i obrnuto proporcionalne, jer kada raste gustina vozila smanjuje se njihova prosečna brzina, zbog većih zagušenja u mreži. Povećanje gustine vozila takođe povećava verovatnoću sudara. Promene u gustini vozila se predviđaju pomoću DRL algoritma, obučavanog na osnovu brzine i pravca kretanja vozila u odnosu na najbližu RSU jedinicu. Pored toga, RSU jedinice unutar posmatranog područja povezane su žičnim vezama preko kojih razmenjuju informacije o dolaznim vozilima. Ovo pomaže susednim RSU jedinicama da ažuriraju informacije o dolasku vozila, odnosno o njihovoj poziciji i brzini.

Predloženi pristup je podeljen na dve faze, a to su faza uspostavljanja putanje i faza izbora putanje. U prvoj fazi se za uspostavljanje optimalnih putanja koristi DRL, na osnovu trenutne lokacije vozila, udaljenosti do najbliže RSU jedinice, gustine vozila i kašnjenja paketa. Ove informacije su sadržane u kontrolnim paketima koje čvorovi periodično razmenjuju sa svojim susedima. Faktori koji mogu pomoći u izboru odgovarajuće putanje u ovoj fazi (kroz dodeljivanje adekvatne nagrade ili kazne za preduzetu akciju) su sposobnost isporuke paketa duž putanje, ukupan broj dostupnih putanja između izvornog i odredišnog čvora, kao i kumulativna težina svake putanje.

Druga faza je izbor putanje, u kojoj čvorovi biraju najbolji sledeći hop koristeći DRL. U ovoj fazi agent učenja najpre predviđa moguće prelaske iz jednog stanja u drugo, na osnovu prethodnih događaja. Ovo se može posmatrati kao problem nadgledanog učenja. Na taj način se predviđaju optimalne putanje za prosleđivanje paketa do odredišta. Na osnovu toga, agent preduzima odgovarajuću akciju, čime menja stanje okruženja, i prima odgovarajuću nagradu. Nagrada zavisi od odnosa maksimalne iskorišćenosti veze u slučaju korišćenja trenutne strategije rutiranja i optimalne iskorišćenosti veze. Autori su putem intenzivnih simulacija u *Network Simulator 2* (NS-2) simulatoru pokazali da ovaj model poboljšava PDR, E2ED i overhed u mreži.

3.3.6. Primena DRL i SDN tehnike u VANET mrežama

Za predstavnika šeste kategorije iz tabele 3.1 odabran je *Software-defined trust based deep reinforcement learning framework* (TDRL-RP) (D. Zhang, Yu i Yang, 2018), koji koristi kombinaciju DRL i SDN tehnike kako bi pomogao u pronalaženju optimalne putanje i izračunavanju njene pouzdanosti. TDRL-RP podrazumeva dve komponente učenja – učenje putanje (*path learning*) i učenje poverenja (*trust learning*). U predloženom pristupu ulogu agenta učenja u DRL algoritmu ima centralizovani SDN kontroler, koji pomaže u selekciji najboljeg sledećeg hopa kroz stalnu interakciju sa VANET okruženjem. Stanje okruženja uključuje skup stanja svih vozila, koja obuhvataju poziciju i stepen uspešno prosleđenih paketa (*forwarding ratio*) svakog vozila. Moguća akcija u odgovarajućem stanju okruženja je izbor suseda kojem određeno vozilo treba da prosledi pakete. Nagrada za preduzetu akciju zavisi od informacija o poverenju određenog vozila, na koju utiču stepen uspešno prosleđenih kontrolnih paketa i paketa podataka.

Ako je poverenje vozila iznad usvojenog praga, smatraće se da je to vozilo pouzdano i povećava se verovatnoća izbora tog vozila za sledeći hop. U suprotnom vozilo se označava kao nepouzdano, odnosno maliciozno. Poverenje celokupne putanje od izvornog do odredišnog vozila zavisi od poverenja svih vozila na toj putanji. U protokolu je predloženo da DRL algoritam koristi CNN mrežu čiji je ulaz stanje okruženja, dok je izlaz odgovarajuća Q-vrednost. Na osnovu ove vrednosti agent učenja bira optimalnu putanju, prateći *ε-greedy* politiku. Primena predloženog pristupa poboljšava PDR i ostvareni protok, što je pokazano simulacionom analizom u *Optimized Network Engineering Tools* (OPNET) simulatoru.

3.3.7. Primena DDRL i SDN tehnike u VANET mrežama

Sedma kategorija u tabeli 3.1 obuhvata *Trust based dueling deep reinforcement learning approach* (T-DDRL) (D. Zhang, Yu, Yang i Tang, 2018), koji kombinuje DDRL i SDN tehniku za pronalaženje optimalne putanje za prosleđivanje podataka ka odredišnom čvoru. Ovaj algoritam je sličan TDRL-RP algoritmu (D. Zhang, Yu i Yang, 2018), s tom razlikom što koristi DDRL algoritam za obučavanje agenta učenja (umesto DRL). Slično TDRL-RP algoritmu, T-DDRL razdvaja proces učenja na učenje putanje i učenje poverenja. Ulogu agenta učenja i u ovom protokolu ima SDN kontroler, koji kroz interakciju sa VANET okruženjem pokušava da nađe optimalnu i pouzdanu putanju do odredišta. Potencijalna stanja, akcije i nagrade su definisani na identičan način kao kod TDRL-RP protokola.

Predloženi T-DDRL se inicijalizuje kada izvorno vozilo treba da uspostavi komunikaciju sa odredišnim vozilom. Prvo, izvorno vozilo pokreće proces otkrivanja putanje kako bi uspostavilo putanju za prenos podataka. Ova procedura ima za cilj da uspostavi početne vrednosti poverenja za svako vozilo i pronađe najbolju putanju za prosleđivanje podataka. Optimalna putanja za slanje podataka određuje se pomoću DDRL algoritma, koji podrazumeva korišćenje DDQN za izračunavanje Q-vrednosti. Ova specifična neuronska mreža podeljena je u dva toka: prvi za izračunavanje funkcije vrednosti, a drugi za izračunavanje funkcije prednosti. Ove dve funkcije predstavljaju dva sastavna dela Q-vrednosti. Prvi deo ukazuje na vrednost odgovarajućeg stanja, dok je drugi dodatna vrednost koja se postiže preduzimanjem određene akcije u datom stanju. Na osnovu simulacione analize u OPNET simulatoru, uz pomoć *TensorFlow* (TF) alata za ML, pokazano je da predloženi pristup poboljšava ostvareni protok paketa i E2ED.

3.3.8. Primena DDRL, SDN i BC tehnike u VANET mrežama

Predstavnik osme kategorije je *Blockchain-based distributed software-defined VANET framework* (Block-SDV) (D. Zhang, Yu i Yang, 2019), koji kombinuje primenu DDRL, SDN i BC tehnike kako bi uspostavio pouzdanu arhitekturu za upravljanje komunikacijom u VANET mrežama. Block-SDV se sastoji od tri sloja: sloj uređaja (*Device Layer*, DL), sloj kontrole područja (*Area Control Layer*, ACL) i sloj kontrole domena (*Domain Control Layer*, DCL), kao i servera za obradu podataka na ivici mreže (*Edge Computing Server,* ECS). DL čine vozila koja pokušavaju da ostvare bežičnu komunikaciju, dok ACL obuhvata SDN kontrolere koji prikupljaju informacije o vozilima u svom području i vezama između njih. Prikupljene informacije se šalju DCL sloju, koji se sastoji od SDN kontrolera koji rade na decentralizovan način uz pomoć BC tehnike. SDN kontroleri na ovom sloju prikupljaju filtrirane podatke sa ACL sloja i dele ih sa drugim SDN kontrolerima, kako bi se poboljšala efikasnost obuke i sinhronizovao globalni prikaz mreže među SDN kontrolerima na ovom sloju. DCL je povezan sa BC sistemom, koji se sastoji od određenog broja BC čvorova, među kojima se nalazi jedan primarni čvor odgovoran za zahteve klijenata i nekoliko konsenzus čvorova koji kontrolišu ostale čvorove u mreži. SDN kontroleri na DCL sloju biraju konsenzus čvorove na osnovu karakteristika poverenja svih čvorova u BC sistemu. Konsenzus čvor osigurava da svi čvorovi poštuju pravila protokola i da je prenos informacija pouzdano sproveden. U Block-SDV protokolu, ako je primarni čvor BC sistema zlonameran ili ako su mu performanse degradirane, SDN kontroleri na DCL sloju biraju novi primarni čvor prema stepenu poverenja čvorova u BC sistemu.

Poverenje svakog čvora (vozila) na DL sloju zavisi od stepena uspešno isporučenih paketa ka susednim čvorovima. U modelu poverenja Block-SDV protokola na DL sloju, opseg vrednosti poverenja svakog vozila je ograničen na interval od 0 do 1. Vrednost poverenja 0 znači potpuno nepoverenje, dok vrednost poverenja 1 znači apsolutno poverenje. Ako između dva vozila u nekoj oblasti nije bilo interakcije, početna vrednost poverenja se postavlja na 0,6 (vozilo sa manjim poverenjem). Uveden je prag poverenja (ε), koji se koristi za prepoznavanje zlonamernih vozila. Drugim rečima, ako je vrednost poverenja bilo kog vozila manja od ovog praga, to se vozilo smatra zlonamernim. Kako bi se minimizirala mogućnost neuspeha prenosa podataka, svako vozilo u određenoj oblasti mora da uspostavi komunikaciju sa svojim najpouzdanijim susednim vozilom (čija je vrednost poverenja viša od ε).

Putanja kojom će se slati podaci u mreži određuje se pomoću DDRL algoritma, a osnovni RL parametri definisani su na sledeći način. Svaki SDN kontroler na DCL sloju predstavlja agenta učenja. Stanje okruženja zavisi od pouzdanosti vozila, pouzdanosti svakog čvora u BC sistemu, računarskih resursa ECS, kao i od broja konsenzus čvorova u BC sistemu. Skup akcija koje preduzima agent uključuje izbor primarnog čvora u BC sistemu, ECS kao računskog resursa, broja konsenzus čvorova u BC sistemu i pouzdanih susednih vozila za prosleđivanje paketa. Nakon preduzimanja akcije, agent prima nagradu koja zavisi od ostvarenog protoka u mreži i ostvarenog

protoka BC sistema. Na osnovu dobijene nagrade, agent učenja računa Q-vrednosti korišćenjem DDRL algoritma sa prioritetizovanim ponavljanjem iskustva (*Prioritized Experience Replay, PER*). Učenje sa PER je uvedeno jer je dokazano da agent može efikasnije učiti iz nekih uzoraka nego iz drugih, koji se na primer više puta ponavljaju. Na ovaj način se ubrzava proces učenja. Block-SDV povećava ostvareni protok u VANET mrežama, što je simulacionom analizom pokazano pomoću simulacionog alata koji koristi *Phyton* jezik i TF alata za ML.

3.3.9. Primena SARSA algoritma u VANET mrežama

Predstavnik devete kategorije u tabeli 3.1 je *RL-based routing protocol for clustered EV-VANET* (RLRC) (Bi i ostali, 2020), koji koristi SARSA algoritam učenja za unapređenje rutiranja podataka. U predloženom pristupu, mrežne čvorove predstavljaju električna vozila opremljena satelitskim sistemom za pozicioniranje, pomoću kog mogu da odrede svoju poziciju, brzinu i informacije o pravcu u realnom vremenu. Mreža je podeljena na odgovarajući broj klastera. Svaki klaster ima CH čvor koji je odgovoran za komunikaciju vozila u tom klasteru, a proces učenja se započinje samo za te čvorove. Da bi bio izabran za CH, čvor (vozilo) mora imati dostupan propusni opseg i preostalu energiju iznad unapred definisanog praga. Takođe, CH je odgovoran za prikupljanje i održavanje informacija o vozilima članovima klastera. Broj i lokacija CH čvorova se određuju na osnovu dužine puta, komunikacionog radijusa CH čvorova i ukupnog broja čvorova u mreži. Vozilo koje ima pakete za slanje ka drugom vozilu šalje te pakete svom CH čvoru. Ako ima direktnu putanju do odredišnog čvora, CH čvor mu prosleđuje pakete, a u suprotnom ih prosleđuje susednom CH čvoru koristeći SARSA algoritam. CH čvorovi prosleđuju paket dok ne dođe do odredišnog klastera, gde CH čvor ovog klastera prosleđuje pakete odredišnom vozilu.

SARSA algoritam učenja je organizovan tako da sva vozila predstavljaju okruženje, agent učenja može biti bilo koji CH čvor, a skup stanja za određenog agenta učenja je definisan skupom svih drugih CH čvorova u mreži. Agenti dobijaju neophodne informacije za ažuriranje Q-vrednosti periodičnom razmenom *Hello* kontrolnih paketa. Akcija koju agent učenja može preduzeti je odabir odgovarajućeg CH čvora za prosleđivanje paketa. Nagrada za preduzetu akciju će imati maksimalnu vrednost ako je trenutni čvor sused odredišnom čvoru, a minimalnu vrednost ako trenutni čvor nema sledeći hop. U ostalim situacijama nagrada zavisi od broja hopova do odredišnog čvora, korisnosti linka i dostupnog propusnog opsega. Autori su simulacionom analizom u simulatoru koji je implementiran u *Phyton* jeziku pokazali da primena predloženog protokola unapređuje mrežne performanse u pogledu PDR i broja hopova.

3.3.10. Primena MBRL i FL tehnike u VANET mrežama

Deseta kategorija protokola u tabeli 3.1 karakteriše se primenom MBRL i FL tehnike u protokolima rutiranja, a odgovarajući predstavnik je *Reinforcement routing protocol for VANETs* (RRPV) (Jafarzadeh i ostali, 2022). Ovaj protokol se zasniva na RL tehnici višestrukih agenata (*Multi-Agent Reinforcement Learning*, MARL), što znači da svi čvorovi u mreži predstavljaju agente učenja koji međusobno sarađuju i istovremeno pokušavaju da pronađu optimalnu politiku rutiranja. RRPV protokol koristi MBRL algoritam za rutiranje paketa i sastoji se od dva procesa, učenja modela (*model learning*) i RL, koji se odvijaju istovremeno. FL sistem se koristi za učenje i kreiranje modela okruženja. Glavni cilj je stvoriti model prelaza iz stanja u stanje i model nagrade zasnovan na kvalitetu veze između susednih čvorova, na koji utiču stabilnost veze i kvalitet konekcije. Stabilnost veze između dva čvora ukazuje na to koliko dugo je veza između njih dostupna. Ova veličina prvenstveno zavisi od brzine, pravca i smera kretanja čvorova. Nivoi stabilnosti definisani protokolom su visoka, srednja i niska stabilnost. Kvalitet konekcije se određuje na osnovu odnosa poslatih i primljenih kontrolnih paketa. Nivoi kvaliteta konekcije u protokolu su dobar, srednji i loš kvalitet. Kombinacijom ova dva faktora ocenjuje se kvalitet veze, koji ima devet nivoa (sve moguće kombinacije stabilnosti veze i kvaliteta konekcije). Najviši nivo kvaliteta veze je izvrsna veza, kada

je stabilnost veze visoka i kvalitet konekcije dobar. S druge strane, najniži nivo kvaliteta veze je veoma loša veza, kada je stabilnost veze niska i kvalitet konekcije loš.

Optimalna politika rutiranja određuje se na osnovu kreiranog modela okruženja uz pomoć MBRL algoritma. U okviru ovog algoritma, svaki čvor koji ima pakete za slanje predstavlja agenta učenja koji može promeniti stanje okruženja preduzimanjem određene akcije. Slanje paketa susedima agenta predstavlja skup dostupnih akcija koje agent može da preduzme. Kada primi određeni paket, čvor procenjuje veze sa svim svojim susedima na osnovu prethodno kreiranog modela okruženja, zatim izračunava Q-vrednosti i bira odgovarajuću akciju na osnovu politike rutiranja. U ovom protokolu je izabrana *softmax* politika rutiranja, koja podrazumeva uvođenje faktora τ preko kog se balansira između eksploatacije stečenog znanja i istraživanja okruženja. Kada je vrednost ovog faktora mala, podstiče se eksploatacija stečenog znanja, dok velike vrednosti ovog faktora stimulišu istraživanje okruženja. U VANET okruženju, kada je mreža visoko dinamična i šansa za pronalaženje stabilnog puta mala, vrednost ovog parametra treba biti postavljena na više vrednosti. Za preduzetu akciju, agent dobija nagradu koja zavisi od udaljenosti i kvaliteta veze između čvorova (određenih u procesu učenja modela). Na osnovu simulacija izvedenih pomoću SUMO i *Objective Modular Network Testbed in C++* (OMNeT++) simulatora, pokazano je da ovaj protokol poboljšava PDR, E2ED i overhed u mreži.

3.3.11. Primena QL bez dodatnih tehnika u FANET mrežama

Protokoli rutiranja za FANET mreže bazirani na QL, koji ne uključuju primenu drugih tehnika, klasifikovani su u jedanaestu kategoriju u tabeli 3.1. Predstavnik ove kategorije je *QL-based message prioritising and scheduling algorithm* (QMPS) (J. Li i Chen, 2020), koji ima zadatak da osigura pouzdan prenos podataka u mreži. Prema ovom protokolu, poruke koje se razmenjuju u mreži prvo se klasifikuju na poruke osetljive na kašnjenje (*delay-sensitive*) i poruke koje mogu tolerisati kašnjenje (*delay-tolerance*). Ovo se radi kako bi u slučaju zagušenja mreže ili pogoršanja kvaliteta veze poruke osetljive na kašnjenje imale viši prioritet za slanje. *Delay-sensitive* poruke uključuju razne tipove kontrolnih komandi i koordinacionih poruka za izbegavanje sudara, koje imaju stroge zahteve za kašnjenje i čiji pravovremen prenos značajno utiče na pouzdanost i bezbednost mreže. Kontrolne komande su ključne za efikasnu kontrolu više bespilotnih letelica, dok su koordinacione poruke od suštinskog značaja za efikasno izvršavanje zajedničkih zadataka. Stoga je uspešna isporuka ovih poruka jako važna za obezbeđivanje dostupnosti, pouzdanosti i sigurnosti FANET mreža. Kašnjenje ili gubitak ovih poruka može dovesti do neuspeha zadatka ili pada bespilotne letelice. *Delay-tolerance* poruke uključuju razne poruke koje mogu podneti povećano kašnjenje i gubitak paketa u većem opsegu. Ovde spadaju glasovne poruke, video snimci, fotografije, podaci sa određenih senzora i slično. Kada poruke sa podacima sa senzora pretrpe povećano kašnjenje, to uzrokuje degradaciju performansi za neke specifične aplikacije, ali ne dovodi do pada mreže.

FANET mreža za koju je kreiran ovaj protokol podrazumeva grupu bespilotnih letelica i zemaljsku kontrolnu stanicu. Neke od bespilotnih letelica služe kao čvorovi za prikupljanje informacija (istraživački čvorovi) i opremljeni su brojnim senzorima (kao što su infracrveni senzori slike, kamere, mikrofoni i senzori temperature). Druge bespilotne letelice funkcionišu kao relejni čvorovi, održavajući vezu između istraživačkih čvorova i zemaljske kontrolne stanice. Svaki čvor u mreži predstavlja agenta učenja, koji preduzima određenu akciju u vidu dodeljivanja odgovarajućeg prioriteta za slanje porukama tolerantnim na kašnjenje. Ovo se izvršava pomoću QL algoritma, koji dodeljuje različite prioritete porukama osetljivim na kašnjenje i porukama tolerantnim na kašnjenje u skladu s dinamikom FANET mreže. Nagrada za akciju formira se na osnovu dve metrike. Prva metrika se odnosi na tip poruka (*message metric*) i predstavlja procenat poruka osetljivih na kašnjenje u redu za slanje poruka susednog čvora. Druga metrika se odnosi na kvalitet linka ka susedu (*neighbor link quality metric*) i zavisi od verovatnoće uspešnog prijema poruke od susednog

čvora. Simulacionom analizom u NS-3 simulatoru je pokazano da QMPS algoritam poboljšava E2ED, ostvareni protok i PLR poruka osetljivih na kašnjenje.

3.3.12. Primena QL i FL tehnike u FANET mrežama

Predstavnik dvanaeste kategorije u tabeli 3.1 je *Q-learning-based fuzzy logic routing algorithm* (QL-FLRA), predložen u (Q. Yang i ostali, 2020), koji koristi QL i FL za optimalno rutiranje podataka u FANET mrežama. Prema ovom protokolu, određivanje optimalne putanje vrši se uz pomoć parametara vezanih za link, koji se odnose na pojedinačni link između dva čvora (bespilotne letelice), i parametara vezanih za putanju, koji se odnose na kompletnu putanju od izvora do odredišta. Parametri vezani za link uključuju brzinu prenosa, stanje energije i status leta. Brzina prenosa oslikava efikasnost prenosa podataka i ključna je za pojedinačne veze između dva čvora, kao i za ukupne performanse mreže. Stanje energije je kritičan parametar za FANET mreže i ako se ne uzme u obzir pri izboru optimalne putanje, raste verovatnoća pražnjenja baterije i samim tim otkazivanja određenog čvora. Status leta zavisi od brzine, pravca i smera kretanja čvora. Ako su brzina, pravac i smer kretanja trenutnog čvora slični brzini, pravcu i smeru kretanja susednog čvora, povećava se verovatnoća da će ovi čvorovi ostati u dometu komunikacije u budućnosti. Stoga je sličnost ovih parametara kod dva susedna čvora ključni faktor koji treba uzeti u obzir pri izboru optimalnog sledećeg hopa.

Parametri koji su vezani za link odnose se isključivo na pouzdanost veze između dva susedna čvora, ali ne uzimaju u obzir efikasnost mreže za kompletnu putanju do odredišta. Zbog toga se u QL-FLRA uvode i parametri vezani za putanju kako bi se ocenio celokupan put od izvornog do odredišnog čvora. Parametri vezani za putanju uključuju broj hopova i vreme uspešne isporuke paketa do odredišta. Biranjem putanje sa manjim brojm hopova, smanjuju se vreme isporuke paketa i opterećenje mreže. Protokol podrazumeva da se na početku komunikacije, kada ne postoje formirane putanje za slanje podataka, najpre pronalazi putanja do odredišta pomoću FL sistema samo na osnovu parametara vezanih za link (hop po hop), nakon čega je moguće odrediti parametre vezane za putanju.

Na osnovu parametara vezanih za putanju, odredišni čvor računa Q-vrednosti za svaki od ovih parametara (Q^h za broj hopova i Q^t za vreme uspešne isporuke paketa) i šalje ih izvornom čvoru. Svi prikupljeni parametri na celoj putanji predstavljaju okruženje u QL procesu. Svaki čvor koji ima pakete za slanje predstavlja agenta učenja koji menja stanje okruženja preduzimanjem određene akcije (odabira sledećeg čvora za prosleđivanje paketa ka odredištu). Nagrade koje utiču na izračunavanje Q-vrednosti pod uticajem su broja hopova i vremena uspešne isporuke paketa. Na kraju, na osnovu oba tipa parametara (vezanih za link i za putanju), određuje se optimalna putanja do odredišta uz pomoć FL sistema. Ova putanja se najčešće razlikuje od prvobitne putanje dobijene samo na osnovu parametara vezanih za link, jer su sada uzete u obzir i performanse celokupne putanje (broj hopova i vreme uspešne isporuke paketa). Predloženi protokol poboljšava ostvareni protok, broj hopova i preostalu energiju čvorova u mreži, što je potvrđeno simulacijama izvedenim u CM simulatoru.

3.3.13. Primena DRL bez dodatnih tehnika u FANET mrežama

Trinaesta kategorija u tabeli 3.1 karakteriše se korišćenjem DRL u protokolu rutiranja, bez kombinacije sa drugim tehnikama, a predstavnik ove kategorije je *DRL-based adaptive and reliable routing protocol* (ARdeep) (J. Liu, Wang, He i Hu, 2020). U ovom protokolu okruženje se sastoji od svih čvorova (bespilotnih letelica) u mreži, a svaki čvor koji ima pakete za slanje predstavlja agenta učenja. Stanje okruženja, za agenta učenja, predstavlja status svih veza sa njegovim susedima. Status svake veze formira se na osnovu očekivanog vremena trajanja konekcije (očekivano vreme dok čvorovi ne izađu iz radijusa komunikacije), procenta pogrešno prenetih

paketa (*Packet Error Rate*, PER), preostale energije susednog čvora, udaljenosti između susednog i odredišnog čvora, kao i minimalne udaljenosti između suseda na dva hopa i odredišnog čvora. Akcija koju agent može preduzeti je odabir jednog od susednih čvorova za prosleđivanje paketa.

Susedni čvorovi periodično razmenjuju *Hello* kontrolne poruke, koje sadrže informacije o njegovoj poziciji, brzini i preostaloj energiji. Na osnovu stanja okruženja, agent bira odgovarajuću akciju uz pomoć duboke Q-mreže (*Deep Q-Network*, DQN), čiji ulaz čini status odgovarajuće veze, a izlaz je Q-vrednost veze ka određenom susednom čvoru. Nakon izračunavanja Q-vrednosti, agent prosleđuje paket susedu sa najvišom Q-vrednošću. Nagrada koju agent prima ima maksimalnu vrednost ako je susedni čvor odredište i minimalnu vrednost ako susedni čvor nema nijednog suseda koji je bliži odredištu od njega samog. U ostalim situacijama, nagrada zavisi od udaljenosti tekućeg čvora i suseda do odredišnog čvora, PER između tekućeg čvora i njegovog suseda, očekivanog vremena trajanja veze i nivoa energije suseda. Autori su simulacionom analizom u *Wireless Sensor Network simulator* (WSNet) simulatoru, uz dodatnu podršku zasnovanu na *Python* jeziku i uz korišćenje TF alata, pokazali da ARdeep poboljšava PDR i E2ED u FANET mrežama.

3.3.14. Primena DRL i FL tehnike u FANET mrežama

Predstavnik poslednje kategorije iz tabele 3.1 je *Fuzzy logic reinforcement learning-based routing algorithm for flying ad hoc networks* (FLRL) (He i ostali, 2020), koji koristi FL i DRL za određivanje optimalne putanje za slanje podataka u FANET mrežama. FL sistem ima za cilj da odredi najbolji čvor za prosleđivanje paketa, na osnovu metrike kašnjenja, ocene stabilnosti i faktora efikasnosti korišćenja propusnog opsega. Metrika kašnjenja direktno zavisi od rastojanja trenutnog čvora do susednog čvora za prosleđivanje paketa. Ocena stabilnosti zavisi od brzine susednih čvorova, što su brzine manje to je stabilnost veća. Efikasnost korišćenja propusnog opsega zavisi od ukupnog broja čvorova uključenih u komunikaciju i opada sa povećanjem broja ovih čvorova. Zbog toga je veoma važno smanjiti broj čvorova koji prosleđuju podatke u mreži. Ako trenutni čvor primi previše zahteva od drugih čvorova u isto vreme, efikasnost korišćenja propusnog opsega će biti prilično niska. Kako bi se kreirala što bolja putanja, važno je ove faktore posmatrati istovremeno. Na ovaj način moguće je uz pomoć FL pronaći putanju do odredišta, ali ova putanja možda neće biti najbolja. Stoga se pored FL tehnike koristi i DRL.

U DRL algoritmu, svaki čvor predstavlja agenta učenja, a stanje susednih čvorova poznato je na osnovu FL algoritma. Akcija koju agent može preduzeti je slanje paketa jednom od suseda, na osnovu čega dobija odgovarajuću nagradu. Prema DRL algoritmu, nagrada će biti 0 ako je sused najbolji (optimalan), i -1 ako je sused suboptimalan. Takođe, nagrada će imati minimalnu vrednost (-*N*, gde je *N* ukupan broj čvorova u mreži) ako nije moguće uspostaviti vezu sa susedom, i maksimalnu vrednost (100) ako je sused ujedno i odredište. Nakon dobijene nagrade mogu se izračunati Q-vrednosti, na osnovu kojih se bira optimalni čvor za prosleđivanje podataka. Na ovaj način je u izbor optimalne putanje uključen broj hopova i kvalitet veze. Simulaciona analiza je izvršena u simulatoru implementiranom u *Matrix Laboratory* (MATLAB) jeziku, na osnovu koje je pokazano da ovaj algoritam poboljšava povezanost čvorova i broj hopova u FANET mreži.

3.4. Poređenje protokola rutiranja baziranih na RL za dinamičke WANET mreže

Analiza prethodno opisanih protokola rutiranja zasnovanih na RL pokazuje da njihova efikasnost najviše zavisi od konstrukcije odgovarajuće funkcije nagrade, koja može zavisiti od različitih uticajnih faktora. Stoga je u nastavku ovog poglavlja izvršeno detaljno poređenje protokola rutiranja baziranih na RL za dinamičke WANET mreže, najpre uzimajući u obzir faktore koji utiču na definisanje funkcije nagrade. Izbor uticajnih faktora prvenstveno zavisi od cilja optimizacije protokola (odnosno parametara koji su optimizovani), tako da je za analizu protokola potrebno uzeti u obzir i parametre koji su posmatrani u procesu evaluacije protokola. Testiranje prethodno

publikovanih protokola izvršeno je u pogodno odabranim simulacionim okruženjima, pa je poređenje protokola izvršeno i na osnovu korišćenog simulacionog alata za njihovo testiranje. U tabeli 3.2 najpre je izvršeno poređenje protokola za VANET mreže.

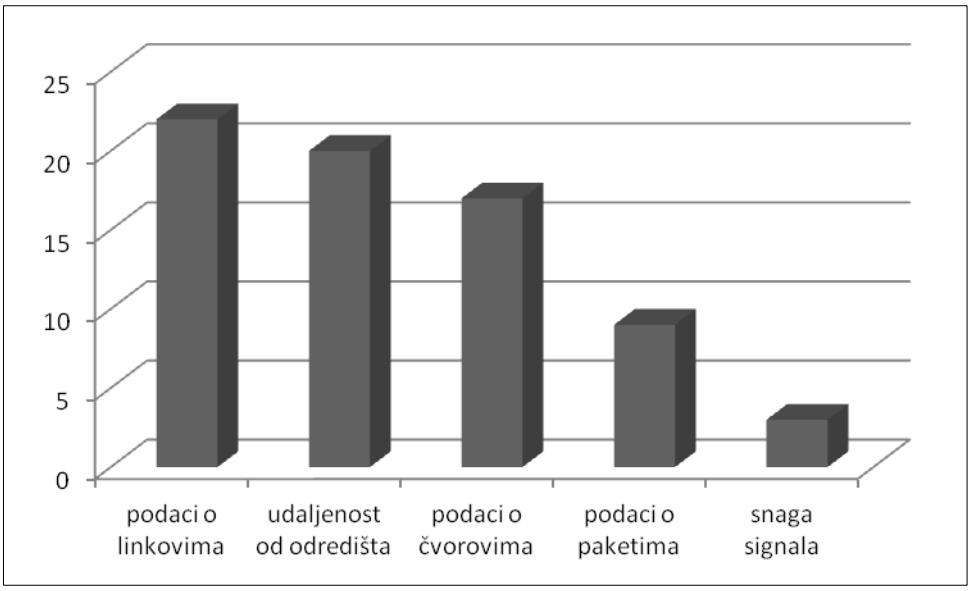
Tabela 3.2. Poređenje protokola rutiranja baziranih na RL za VANET mreže

Referenca	*Protokol*	*Uticajni faktori na nagradu*	*Posmatrane performanse*	*Simulacioni alati*
(An i ostali, 2018)	CEVCS	da li je čvor sused, broj hopova, kvalitet linka	PDR, ostvareni protok paketa	NS-2
(Dai i ostali, 2018)	QLASS	reputacija i isplativost akcija čvora	PDR, reputacija, korisnost	CM
(C. Wu i ostali, 2018)	V2R-CBR	da li je čvor sused, broj hopova, isplativost, kvalitet linka	PDR, broj kolizija MAC okvira, E2ED, ostvareni protok paketa	NS-2
(J. Wu i ostali, 2018)	ARPRL	da li je kontrolni paket stigao od izvornog čvora	PDR, E2ED, broj hopova, overhed	QualNet
(D. Zhang, Yu i Yang, 2018)	TDRL-RP	informacije o poverenju	PDR, ostvareni protok paketa	TF, OPNET
(D. Zhang, Yu, Yang i Tang, 2018)	T-DDRL	informacije o poverenju	ostvareni protok paketa, E2ED	TF, OPNET
(Ji i ostali, 2019)	RHR	tip kontrolnih paketa	PDR, RTT, overhed	NS-3
(F. Li i ostali, 2019)	QGrid	da li je poruka prosleđena odredišnom gridu	PDR, broj hopova, kašnjenje paketa, broj prosleđivanja, ostvareni protok paketa	CM
(C. Wu i ostali, 2019)	DTNP	direktna konekcija, broj hopova, proteklo vreme od poslednje konekcije	kašnjenje paketa, PDR	ONE
(Y. Yang i ostali, 2019)	VDDS	broj hopova, kvalitet linka	ostvareni protok paketa, broj CH gejtveja	CM
(D. Zhang, Yu i Yang, 2019)	Block-SDV	ostvareni protok paketa	ostvareni protok paketa	TF, *Phyton*
(D. Zhang, Zhang i Liu, 2019)	RSAR	broj hopova, pouzdanost linka, propusni opseg	PDR, E2ED, prosečna dužina putanje, overhed	NS-2
(Bi i ostali, 2020)	RLRC	da li je čvor sused odredišnog čvora, broj hopova, korisnost linka, propusni opseg	PDR, broj hopova	*Python*
(G. Li i ostali, 2020)	ECTS	da li podaci o stanju punjenja električnih vozila stižu na odredište	trošak komunikacije, verovatnoća povezanosti, PDR, overhed	NS-3
(Nahar i Das, Jun 2020)	RL-SDVN	rastojanje do odredišnog vozila	kašnjenje, ostvareni protok paketa	NS-3

Referenca	*Protokol*	*Uticajni faktori na nagradu*	*Posmatrane performanse*	*Simulacioni alati*
(Nahar i Das, Nov. 2020)	SeScR	kvalitet dostupnih putanja, brzina vozila, lokacija	stabilnost i trajanje klastera, vreme nepovezanosti klastera, kašnjenje, ostvareni protok paketa, kašnjenje u obradi	SUMO, OMNeT++
(Roh i ostali, 2020)	Q-LBR	opterećenje relejne UAV, zagušenje zemaljske mreže	PDR, iskorišćenost mreže, kašnjenje paketa	OPNET
(Saravanan i Ganeshkumar, 2020)	DRLV	maksimalna iskorišćenost linka pod budućom strategijom rutiranja, optimalna iskorišćenost linka	PDR, E2ED, overhed	NS-2
(Smida i ostali, 2020)	LEQRV	životni vek i kvalitet linka, rastojanje do odredišnog čvora, srednja procenjena ocena (Rodriguez-Bocca, 2008), broj suseda, nivo slobodnog bafera čvora	srednja procenjena ocena, maksimalni odnos signal-šum, strukturna sličnost, E2ED, gubitak okvira	SUMO, NS-3
(J. Wu i ostali, 2020)	QTAR	kvalitet i vreme isteka linka, kašnjenje paketa	PDR, E2ED	QualNet
(X. Yang i ostali, 2020)	HAEQR	da li čvor pripada skupu suseda odredišnog čvora	PDR, E2ED, broj hopova	SUMO, NS-2
(D. Zhang i ostali, 2020)	SD-TDQL	vrednost poverenja svakog vozila, verovatnoća prijema potvrde o uspešnom prenosu	PLR, kašnjenje	MATLAB, TF
(S. Jiang i ostali, 2021)	QAGR	RSS, udaljenost prenosa, kolizije između vozila	PDR, E2ED, broj hopova	NS-3
(Ye i ostali, 2021)	VMDRL	gubitak energije, brzina prenosa	potrošnja energije, PLR, vreme prenosa, verovatnoća prekida komunikacije	CM
(W. Zhang i ostali, 2021)	FLHQRP	da li trenutni klaster pripada skupu suseda odredišnog klastera, gustina saobraćaja u klasteru	PDR, E2ED, broj hopova, overhed	NS-2
(Jafarzadeh i ostali, 2022)	RRPV	kvalitet linka, rastojanje susednog i odredišnog čvora	PDR, kašnjenje, overhed	OMNeT++
(Lolai i ostali, 2022)	RRIN	razlika brzina vozila, pravac kretanja vozila, broj paketa podataka u redu za slanje, slabljenje signala, pouzdanost linka	PDR, PLR, kašnjenje, ostvareni protok paketa	MATLAB

Referenca	*Protokol*	*Uticajni faktori na nagradu*	*Posmatrane performanse*	*Simulacioni alati*
(Luo i ostali, 2022)	IV2XQ	da li je paket prosleđen na odredišnu raskrsnicu	PDR, E2ED, broj hopova, overhed	SUMO, Veins, OMNeT++
(Ahmed i ostali, 2023)	RMDRL	propusni opseg linka, odnos signal-šum, brzina prenosa	ostvareni protok paketa, odnos signal-šum, kašnjenje	NS-2
(X. Liu i ostali, 2023)	QBACC	zauzetost kanala, protok paketa	opterećenje kanala, PDR, PLR, stepen greške u prenosu *Beacon* paketa	Veins
(Upadhyay i ostali, 2023)	IDRL	gustina, brzina i lokacija vozila, rastojanje do RSU	E2ED, PDR, overhed	NS-2

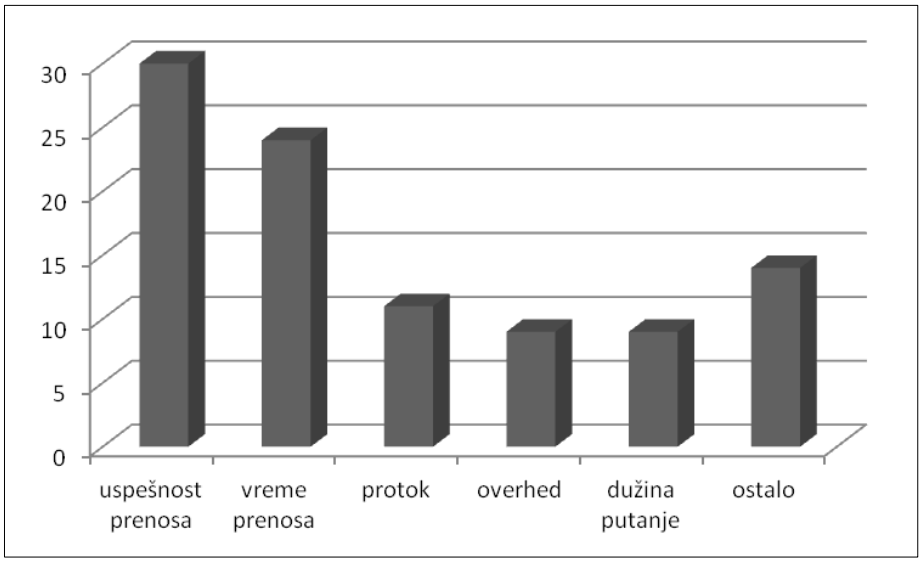
Na slici 3.6 predstavljena je zastupljenost pojedinih tipova uticajnih faktora u protokolima rutiranja za VANET mreže. Kako bi ih bilo lakše analizirati, uticajni faktori su podeljeni na nekoliko tipova, u zavisnosti od toga na šta se odnose.



Slika 3.6. Zastupljenost tipova uticajnih faktora u protokolima rutiranja za VANET mreže

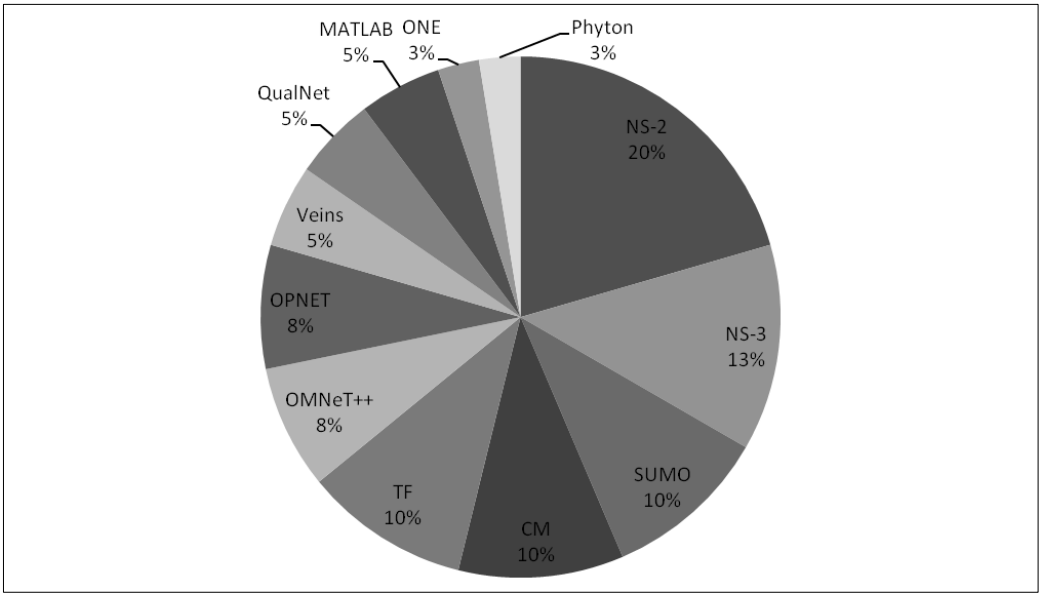
Primetno je da se ovi faktori najčešće odnose na podatke o linkovima između čvorova, kao što su kvalitet, pouzdanost, životni vek, iskorišćenost, propusni opseg, zauzetost linka, itd. Odmah iza slede uticajni faktori koji se odnose na podatke o udaljenosti od odredišnog čvora, gde se ubrajaju broj hopova i rastojanje do odredišnog čvora, lokacija odredišnog čvora, da li je trenutni čvor sused odredišnog čvora, itd. Naredna najzastupljenija grupacija uticajnih faktora su podaci o čvorovima, između ostalih poverenje, reputacija, brzina, pravac kretanja, lokacija, isplativost akcija, nivo slobodnog bafera čvora, broj susednih čvorova, broj kolizija između čvorova, gubitak energije čvora, itd. Nakon toga slede uticajni faktori koji se odnose na podatke o paketima koji se šalju. Ovde spadaju tip kontrolnih paketa, kašnjenje, brzina prenosa, verovatnoća prijema, ostvareni protok paketa, itd. Na kraju, u par protokola na izbor optimalne putanje utiču faktori vezani za snagu signala, kao što su slabljenje i jačina primljenog signala.

U zavisnosti od cilja optimizacije, u protokolima su praćeni različiti metrički pokazatelji mrežnih performansi. Na slici 3.7 predstavljena je zastupljenost pojedinih tipova posmatranih performansi u VANET mrežama.



Slika 3.7. Zastupljenost tipova posmatranih performansi za evaluaciju protokola rutiranja za VANET mreže

U najvećem broju protokola, ove performanse se odnose na uspešnost prenosa paketa i tu spadaju PDR, PLR, gubitak okvira, verovatnoća povezanosti i verovatnoća prekida komunikacije. Takođe, veoma značajan skup performansi odnosi se na vreme prenosa paketa. Tu spadaju RTT, kašnjenje, vreme prenosa i kašnjenje u obradi paketa. U nešto manjem obimu zastupljeni su parametri koji se odnose na ostvareni protok paketa, overhed rutiranja i dužinu putanje (broj hopova, broj prosleđivanja, prosečna dužina putanje). Može se primetiti da značajan broj parametara spada u kategoriju "ostalo", a tu se ubrajaju raznovrsni mrežni parametri kao što su iskorišćenost mreže, trošak komunikacije, odnos signal-šum, reputacija čvora, korisnost čvora, potrošnja energije, stabilnost i trajanje klastera, opterećenje kanala, broj kolizija okvira na podsloju kontrole pristupa mediju (*Media Access Control*, MAC), itd.



Slika 3.8. Zastupljenost simulacionih alata za testiranje protokola rutiranja za VANET mreže

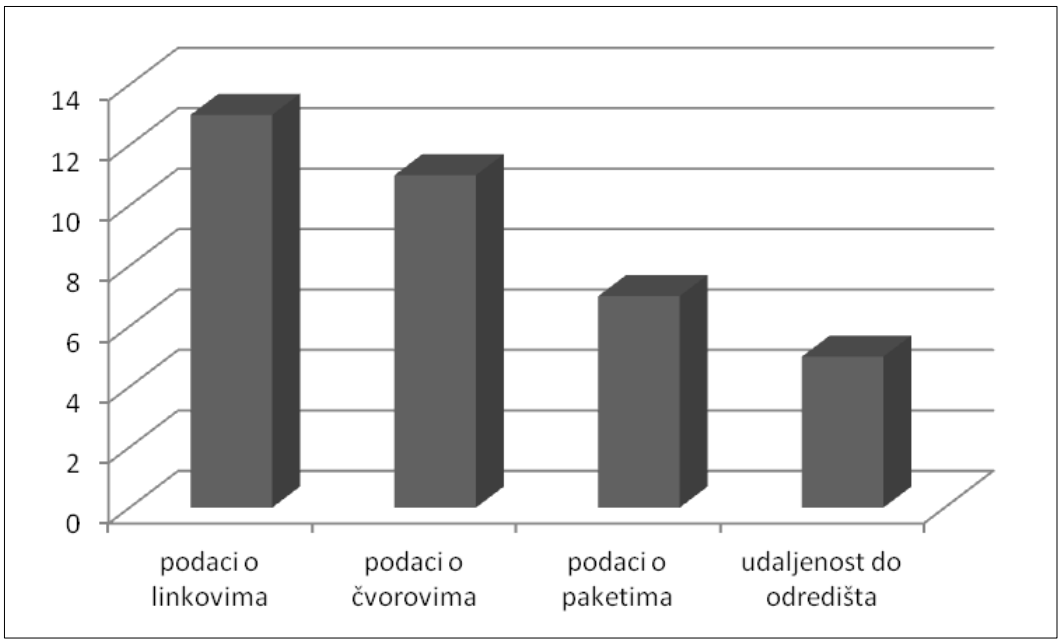
Evaluacija performansi i testiranje predloženih protokola u svim publikovanim radovima vrši se korišćenjem različitih simulacionih alata, a na slici 3.8 prikazana je njihova procentualna zastupljenost. Najčešće korišćeni simulatori su NS-2 i NS-3, slede SUMO i TF alati, kao i CM simulatori (razvijeni od strane istraživača za testiranje konkretnog protokola). U nešto manjem obimu korišćeni su OMNeT++, OPNET, *Vehicles in Network Simulation* (Veins), QualNet simulatori i simulacioni alati kreirani u MATLAB jeziku. Za testiranje po jednog protokola korišćeni su *Opportunistic Network Environment* (ONE) simulator i simulacioni alat baziran na *Phyton* jeziku.

U tabeli 3.3 izvršeno je poređenje protokola za FANET mreže. Iako imaju mnogo toga zajedničkog sa protokolima za VANET mreže, protokoli rutiranja za FANET mreže moraju da se prilagode nešto drugačijem mrežnom okruženju pa su zato i uticajni faktori na nagradu u RL procesu prilično drugačiji od onih kod VANET mreža.

Tabela 3.3. Poređenje protokola rutiranja baziranih na RL za FANET mreže

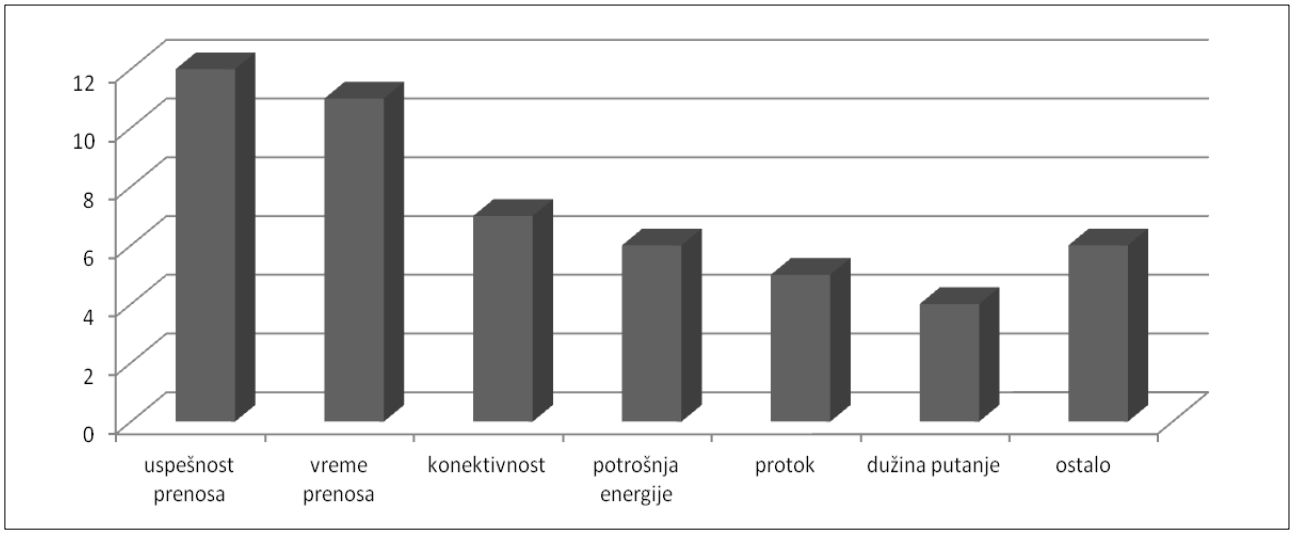
Referenca	*Protokol*	*Uticajni faktori na nagradu*	*Posmatrane performanse*	*Simulacioni alati*
(Zheng i ostali, 2018)	RLSRP	uslovna verovatnoća uspešnog ili neuspešnog prenosa paketa do sledećeg čvora	indikatori uspešnosti uspostavljanja putanje, prosečan životni vek putanje, broj hopova, PDR, protok bez ponovnog prenosa, kašnjenje paketa	MATLAB, NS-2
(He i ostali, 2020)	FLRL	optimalnost susednog čvora, konektivnost linka	broj hopova, konektivnost linka	MATLAB
(Khan i Yau, 2020)	RL-FANET	uspešnost prenosa paketa	potrošnja energije, broj prekida linkova, životni vek mreže	MATLAB
(J. Li i Chen, 2020)	QMPS	proporcija poruka osetljivih na kašnjenje u redu za slanje, verovatnoća uspešnog prijema poruke od susednog čvora	E2ED, ostvareni protok paketa, PLR	NS-3
(J. Liu, Wang, He i Hu, 2020)	ARdeep	link vodi/ne vodi ka odredišnom čvoru, link je/nije lokalni minimum, rastojanje do odredišnog čvora, PER, očekivano vreme trajanja veze, nivo energije susednog čvora	PDR, E2ED	TF, *Python*, WSNet
(J. Liu, Wang, He, Jaffres-Runser i ostali, 2020)	QMR	link vodi/ne vodi ka odredišnom čvoru, link je/nije lokalni minimum, E2ED, potrošnja energije	E2ED, PDR, potrošnja energije	WSNet
(Mowla i ostali, 2020)	AFRL	detektovan/nije detetektovan ometač	tačnost, uspešnost isporuke, broj hopova, broj iteracija do postizanja konvergencije, kumulativna nagrada	NS-3

Referenca	*Protokol*	*Uticajni faktori na nagradu*	*Posmatrane performanse*	*Simulacioni alati*
(Q. Yang i ostali, 2020)	QL-FLRA	broj hopova, vreme uspešne isporuke paketa	broj hopova, preostala energija čvorova, ostvareni protok paketa	CM
(Arafat i Moh, 2021)	QTAR	koji je tip čvora sledeći hop, E2ED, brzina čvora, potrošnja energije	PDR, E2ED, potrošnja energije, životni vek mreže, overhed	MATLAB
(Da Costa i ostali, 2021)	Q-FANET	link vodi/ne vodi ka odredišnom čvoru, link je/nije lokalni minimum	maksimalno E2ED, džiter, PDR	WSNet
(Sliwa i ostali, 2021)	PARRoT	vreme isteka linka, promena u skupu suseda čvora za prosleđivanje	PDR, E2ED	OMNeT++, INETMANET
(Ayub i ostali, 2022)	AI-Hello	domet prenosa, raspoloživi vazdušni prostor, broj dronova, opsezi brzina	potrošnja energije, overhed, PDR, ostvareni protok paketa, E2ED	NS-3
(Hosseinzadeh i ostali, 2023)	QRF	vreme trajanja linka, preostala energija čvora, kvalitet linka	potrošnja energije, PDR, overhed, E2ED, životni vek mreže	NS-2
(Song i ostali, 2024)	DRL-AdCAR	mogućnosti kodovanja, dobitak kodovanja, kvalitet linka	performanse kodovanja, ostvareni protok paketa, PDR, E2ED	NS-3, OpenAI Gym



Slika 3.9. Zastupljenost tipova uticajnih faktora u protokolima rutiranja za FANET mreže

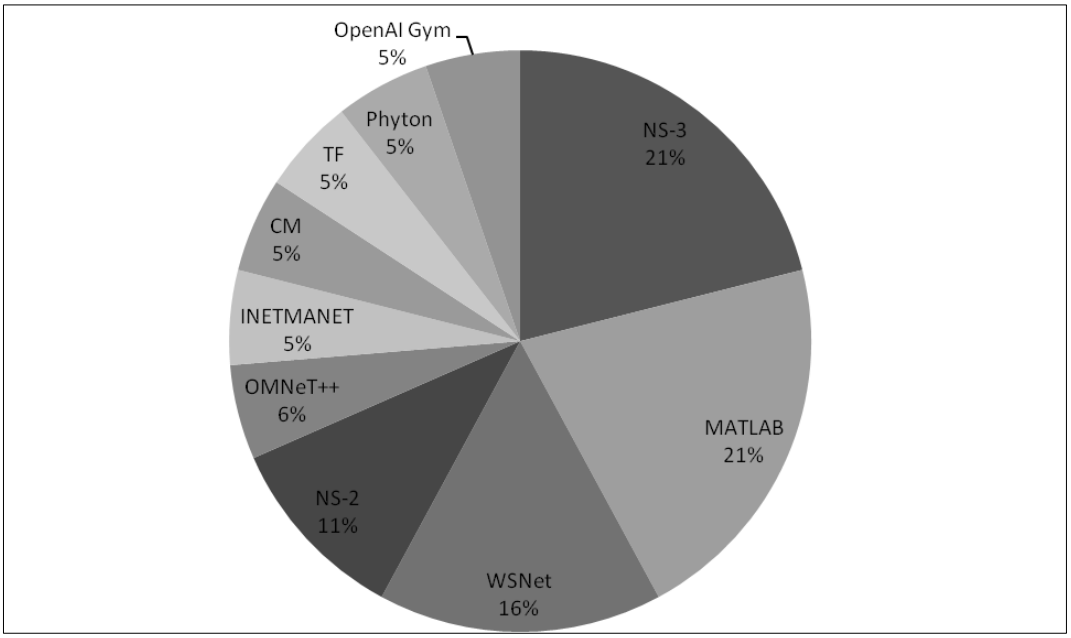
Na slici 3.9 prikazana je zastupljenost pojedinih tipova uticajnih faktora u protokolima rutiranja za FANET mreže. Najzastupljeniji uticajni faktori su oni koji se odnose na podatke o linkovima u mreži. Ovi podaci uključuju vreme isteka, kvalitet, konektivnost linka, da li je link lokalni minimum, itd. Nakon njih, slede uticajni faktori koji se odnose na podatke o čvorovima u mreži, kao što su potrošnja energije, brzina, pouzdanost čvorova, itd. Odmah iza su faktori koji se odnose na podatke o paketima koji se šalju u mreži, gde spadaju kašnjenje, verovatnoća uspešnog prijema, vreme uspešne isporuke paketa, itd. Za razliku od VANET mreža, gde su bili među najzastupljenijim, uticajni faktori koji se odnose na udaljenost od odredišnog čvora kod FANET mreža su najmanje zastupljeni. Ovi podaci mogu biti broj hopova do odredišta, rastojanje do odredišta ili informacija da li link vodi ka odredištu.



Slika 3.10. Zastupljenost tipova posmatranih performansi za evaluaciju protokola rutiranja za FANET mreže

Na slici 3.10 prikazana je zastupljenost pojedinih tipova performansi koje su korišćene za evaluaciju predloženih protokola rutiranja za FANET mreže. Kao i kod VANET mreža, najčešće je cilj optimizacije protokola povećati uspešnost prenosa podataka i smanjiti kašnjenje (vreme prenosa) paketa, tako da su uticajni faktori koji se odnose na ove performanse i ovde najzastupljeniji. Uspešnost prenosa uključuje PDR, PLR, uspešnost i tačnost isporuke paketa. Podaci koji se odnose na vreme prenosa paketa su kašnjenje paketa i varijacija kašnjenja (džiter). S obzirom da FANET mreže karakteriše mala gustina i velika brzina čvorova, važna mrežna performansa je konektivnost čvorova u mreži. Ovo obuhvata podatke koji pokazuju da li je moguće uspostaviti konekciju sa drugim čvorovima u mreži i tu se ubrajaju živorni vek mreže, životni vek putanje, broj prekida linkova, konektivnost linka, itd. Nakon toga, najzastupljenije performanse su potrošnja energije (koja je kritičan parametar FANET mreže), ostvareni protok paketa i dužina putanje do odredišta. U par protokola posmatrane su i "ostale" mrežne performanse, kao što su overhed, konvergencija i kumulativna nagrada.

Na slici 3.11 predstavljena je zastupljenost simulacionih alata za testiranje protokola rutiranja za FANET mreže. U ovom slučaju najčešće korišćeni simulacioni alati su NS-3 i MATLAB, a odmah nakon njih i WSNet. U par protokola je korišćen NS-2 simulator, dok su za testiranje po jednog protokola rutiranja korišćeni OMNeT++, INTERMANET, CM, TF, *Phyton* i OpenAI Gym simulaciona okruženja.



Slika 3.11. Zastupljenost simulacionih alata za testiranje protokola rutiranja za FANET mreže

3.5. Sumarni zaključci pregleda protokola

Analizom literature iz ovog poglavlja može se uočiti da novi protokoli rutiranja bazirani na RL mogu postići značajno bolje mrežne performanse od tradicionalnih algoritama, kako u VANET, tako i u FANET mrežama. Ovim je potkrepljena jedna od polaznih hipoteza ove disertacije. Integracijom RL sa drugim tehnikama, dodatno se unapređuju performanse ovih mreža. RL je nova i složena tehnika koja treba adekvatno da se primenjuje kako bi se iskoristile sve njene potencijalne prednosti. Ova tehnika je još uvek predmet intenzivnog istraživanja, a postoje mnoga otvorena pitanja i ograničenja koja treba prevazići. Jedna od dilema koja se može primetiti je izbor odgovarajuće vrste RL za dati problem rutiranja. Analizom najnovije literature može se uočiti da većina autora koristi QL, nešto manje DRL, nakon čega slede DDRL i MBRL, dok se SARSA algoritam primenjuje samo u jednom protokolu (slika 3.4). Na tip RL prvenstveno utiče veličina mreže za koju se protokol kreira.

Pored toga, autori i dalje traže optimalnu definiciju agenta učenja, njegovih akcija i stanja okruženja. Kada je mreža centralizovana, najčešći pristup je odabrati ovaj centralni uređaj kao agenta za učenje, dok svi mrežni čvorovi (vozila ili bespilotne letelice) formiraju okruženje. U distribuiranim *ad hoc* mrežama, uobičajeno rešenje je da se svi čvorovi koriste kao agenti, dok u algoritmima rutiranja zasnovanim na klasterima CH čvorovi obično preuzimaju ulogu agenata. U pojedinim protokolima agente učenja predstavljaju paketi koji se šalju, a stanja u kojima mogu da se nađu su pojedini mrežni čvorovi. Da bi se dodatno poboljšale mrežne performanse, RL se može koristiti u kombinaciji sa nekim drugim tehnikama, kao što su FL, SDN i BC, ali većina autora još uvek ne koristi ovu mogućnost (slika 3.5).

Imajući u vidu da je QL relativno jednostavna tehnika i da za izbor optimalne putanje za slanje podataka koristi podatke smeštene u Q-tabele, ona je pogodna za relativno male mreže, zbog čega je većina protokola rutiranja analiziranih u ovom poglavlju ograničena na primenu u ovoj vrsti mreža. Međutim, QL nije adekvatno rešenje za složene mreže sa velikim brojem čvorova, jer će prostor mogućih akcija, stanja i veličina Q-tabela rasti eksponencijalno. U tim slučajevima treba koristiti DRL ili neku metodu za ograničavanje Q-tabele. Implementacija DRL algoritma je dosta kompleksnija, zahteva veće računarske resurse i značajno vreme konvergencije, pa je pogodnije rešenje za mreže sa centralizovanim entitetima kao što su SDN ili mreže zasnovane na klasterima sa

RSU jedinicama. Praktična primena ovih tehnika takođe mora pažljivo razmotriti aspekte sigurnosti. Iako je centralizovani pristup veoma dobro rešenje, u nedavnim studijama autori razmatraju integraciju BC tehnike koja pruža distribuirani sistem upravljanja poverenjem. Trenutno, manji broj autora koristi DRL, posebno kada uključuje neku drugu tehniku, ali se broj protokola zasnovanih na DRL konstantno povećava.

Pored najvažnijeg pitanja izbora vrste RL, prisutni su različiti pristupi u definisanju uticajnih faktora na nagradu (tabele 3.2 i 3.3), što očigledno zavisi od parametara koje treba optimizovati. Uticajni faktori se uglavnom odnose na udaljenost do odredišta, podatke o linkovima između čvorova, podatke o čvorovima i podatke o paketima. Prilikom formiranja nagrade, agent se oslanja na različite mehanizme povratnih informacija koji obično uključuju razmenu dodatnih kontrolnih paketa, što svakako povećava opterećenje mreže. Nažalost, ovo se ne može izbeći, ali je poželjno razmotriti mogućnost korišćenja hijerarhijskog rutiranja koje ograničava područje za razmenu kontrolnih paketa, čime se smanjuje dodatno opterećenje koje unose ovi paketi (overhed). Najčešće je cilj optimizacije protokola povećanje uspešnosti isporuke paketa i smanjenje vremena prenosa paketa, a kod FANET mreža se dodatno teži povećanju konektivnosti čvorova i optimizaciji potrošnje energije.

Još jedan važan aspekt pri predlaganju novih protokola je proces njihove evaluacije i testiranja. Najbolja metoda validacije protokola su *testbed* eksperimenti koji koriste realno okruženje za prikupljanje podataka. Međutim, nijedan od analiziranih radova nije koristio ovaj pristup, očigledno zbog njegove kompleksnosti i visoke cene realizacije. Umesto toga, korišćena su različita simulaciona okruženja za testiranje protokola i analizu rezultata njihove primene. Kao što se može videti iz slika 3.8 i 3.11, većina autora koristi simulatore otvorenog koda ili kreira sopstvena simulaciona okruženja. Posmatrajući zbirno VANET i FANET mreže, najzastupljeniji mrežni simulatori su NS-2 i NS-3. Upravo je ovo jedan od motiva za korišćenje NS-3 simulatora kao alata za implementaciju i testiranje postojećih i novog predloženog protokola rutiranja u ovoj doktorskoj disertaciji.

Detaljna analiza aktuelne literature imala je ključnu ulogu u izboru relevantnih predstavnika RL baziranih protokola rutiranja za poređenje sa novim Q-DRAV protokolom (Bugarčić i ostali, 2024) u nastavku disertacije. U tu svrhu izabrani su QLAODV (C. Wu i ostali, 2010) i ARPRL (J. Wu i ostali, 2018) protokoli, koji pokazuju značajno unapređenje mrežnih performansi u poređenju sa ranijim protokolima rutiranja. Oba protokola podrazumevaju samo V2V komunikaciju i baziraju se na QL algoritmu, tako da ne zahtevaju dodatnu komunikacionu infrastrukturu (RSU jedinice, bazne stanice, itd.) i prevelike računarske resurse. Polazna osnova za oba protokola je AODV protokol (opisan u poglavlju 2.2), koji je izabran za predstavnika tradicionalnih protokola rutiranja prilikom poređenja sa novim protokolom. Iz tog razloga će QLAODV i ARPRL protokoli u nastavku biti nešto detaljnije opisani.

3.6. Relevantni predstavnici RL baziranih protokola rutiranja za dinamičke WANET mreže

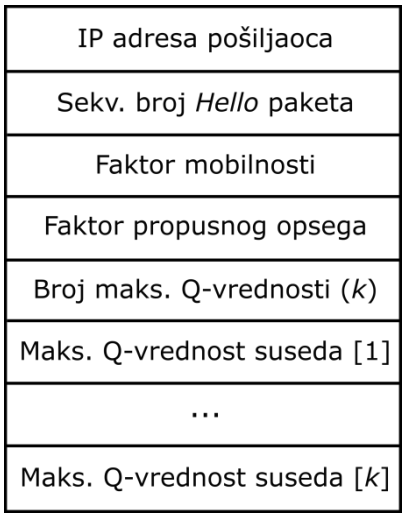
S obzirom da je QL najpopularniji RL algoritam u protokolima rutiranja za dinamičke WANET mreže, u nastavku su predstavljena dva izuzetno značajna protokola rutiranja bazirana na ovom algoritmu. Ovi protokoli su iskorišćeni za poređenje sa novim Q-DRAV protokolom rutiranja u simulacionoj analizi, pa je stoga bitno detaljno objasniti njihovu matematičku formulaciju i principe funkcionisanja. Najpre je opisan QLAODV protokol, koji predstavlja modifikaciju popularnog AODV protokola. Zatim je predstavljen ARPRL protokol, koji uključivanjem novih metrika u QL proces doprinosi daljem unapređenju mrežnih performansi. Oba protokola su dizajnirana za primenu u VANET mrežama.

3.6.1. QLAODV protokol

Jedan od najznačajnijih primera modifikacije tradicionalnih protokola rutiranja primenom RL, kako bi se unapredile performanse dinamičkih WANET mreža, definitivno je QLAODV protokol. On predstavlja modifikaciju tradicionalnog reaktivnog AODV protokola rutiranja, koja je ostvarena uključivanjem QL algoritma u proces izbora optimalne putanje za slanje podataka. Kombinovanjem reaktivnog otkrivanja putanja, periodične razmene podataka o stanju linkova i preventivne promene putanja, QLAODV vrlo dobro ispunjava zahteve V2V komunikacije. Protokol funkcioniše tako što izvorni čvor kada želi da pošalje podatke ka odredištu, najpre proverava u svojoj tabeli rutiranja da li ima neku dostupnu putanju. Ukoliko nema nijednu, započinje proceduru otkrivanja putanje identičnu kao kod AODV protokola, kako bi kreirao putanju do odredišnog čvora. Da bi se sprečilo često pokretanje ove procedure u gustim mrežama (što je slučaj kod AODV protokola), QLAODV koristi mehanizam dinamičke promene putanja za preventivno prebacivanje na nove putanje, čime se smanjuje overhed protokola rutiranja.

Ukoliko ima više dostupnih putanja, čvor koristi QL algoritam za izbor optimalne. U ovom procesu svaki čvor se ponaša kao agent učenja, koji kroz interakciju sa okruženjem pokušava da izabere optimalnu putanju za slanje paketa. Svakoj putanji agent dodeljuje odgovarajuću Q-vrednost, koja može biti u opsegu [0,1], a za slanje paketa bira onu sa najvećom Q-vrednošću. Na Q-vrednosti utiču broj hopova, stabilnost mreže i zauzetost propusnog opsega čvora. Ove vrednosti se smeštaju u dinamičke Q-tabele u čvorovima, čije dimenzije zavise od broja odredišta i broja suseda čvora koji ih održava. U procesu učenja učestvuju svi čvorovi u mreži, koji međusobno sarađuju kako bi kreirali optimalnu putanju za slanje podataka.

Kako bi čvorovima u mreži bilo omogućeno da vrše interakciju sa okruženjem i na taj način prikupljaju potrebne informacije za ažuriranje Q-vrednosti, uvedeno je periodično slanje difuznih *Hello* kontrolnih paketa ka susednim čvorovima. Na osnovu informacija iz *Hello* paketa čvorovi ažuriraju Q-vrednosti za sve moguće putanje ka svim odredištima u mreži. Struktura *Hello* paketa u QLAODV protokolu prikazana je na slici 3.12.



Slika 3.12. Struktura *Hello* paketa kod QLAODV protokola

Svaki paket sadrži najpre IP adresu pošiljaoca, koja služi za identifikaciju izvornog čvora na prijemu, dok sekvencijalni broj označava redosled poslatih *Hello* paketa. Slede faktor mobilnosti (*Mobility Factor*, MF) i faktor propusnog opsega (*Bandwith Factor*, BF), koji predstavljaju uticajne faktore za ažuriranje Q-vrednosti i reflektuju relativno kretanje posmatranog čvora u odnosu na susedne čvorove i opterećenost propusnog opsega posmatranog čvora, respektivno. Paket sadrži i

niz maksimalnih Q-vrednosti do svih odredišta u mreži, koje čvor očitava iz svoje Q-tabele, kao i njihov broj (*k*). Ove vrednosti čvor ažurira periodično po isteku odgovarajućeg tajmera i smešta ih u *Hello* paket pre njgovog slanja ka susednim čvorovima.

Na početku komunikacije čvor nema nikakve informacije o okruženju, tako da se sve Q-vrednosti postavljaju na 0. Po prijemu *Hello* paketa od svog suseda *n*, čvor *c* ažurira Q-vrednosti za putanje ka svim odredištima koje vode preko tog suseda. Ove Q-vrednosti se vezuju za uređeni par (*d*, *n*), gde *d* označava odredišni čvor i *n* susedni čvor preko kog se šalju podaci ka odredištu. Nakon ažuriranja, čvor *c* smešta ove Q-vrednosti u svoju Q-tabelu. S obzirom da se istraživanje okruženja obavlja periodičnom razmenom *Hello* paketa, svaki čvor ima informaciju koji sused je u određenom trenutku najbolji izbor za sledeći hop ka odredištu. Zbog toga, kada bira putanju kojom će poslati pakete ka odredištu, uvek bira onu sa najvećom Q-vrednošću. Čvor *c* računa Q-vrednost za putanju do odredišta *d* preko susednog čvora *n* na osnovu sledeće jednačine:

$$Q_c(d,n) \leftarrow \left(1-\alpha_{c,n}\right)\cdot Q_c(d,n) + \alpha_{c,n}\cdot\left(R_{c,n} + \gamma_{c,n}\cdot max_{y\in Nei(n)}Q_n(d,y)\right). \tag{3.3}$$

Ova jednačina prati formu opšte jednačine QL algoritma (3.1), s tim što se Q-vrednost odnosi na uređeni par (odredište, sused) i parametri $\alpha_{c,n}$, $\gamma_{c,n}$ i $R_{c,n}$ se odnose na tekući čvor *c* i njegovog suseda *n*, od kog je primio *Hello* paket. U QLAODV protokolu definisana je fiksna vrednost od 0,8 za stepen učenja ($\alpha_{c,n}$), jer je eksperimentalnim putem utvrđeno da je ovo optimalna vrednost. S obzirom da je početna Q-vrednost svakog linka jednaka 0, to znači da će $Q_c(d, n)$ postajati veće sa svakim ažuriranjem, ako se ostali elementi u relaciji (3.3) ne menjaju. Generalno, ako je trajanje veze dugo, veća je verovatnoća da će ona i dalje biti stabilna u budućnosti, što je slučaj sa vozilima koja se kreću u istom smeru. Diskontni faktor ($\gamma_{c,n}$) ima promenljivu vrednost, na koju utiču mobilnost čvorova i zauzetost propusnog opsega. Računa se na osnovu sledeće jednačine:

$$\gamma_{c,n} \;=\; 0{,}9\cdot\sqrt{(MF_n\cdot BF_n)}. \tag{3.4}$$

Vrednost diskontnog faktora je uvek manja od 1, pa se prolaskom kroz svaki čvor preko njega umanjuje Q-vrednost. Na taj način je broj hopova uključen u definisanje Q-vrednosti (prednost se daje putanjama sa manjim brojem hopova). Parametar MF_n predstavlja faktor mobilnosti susednog čvora *n*, sadržan je u *Hello* paketu i računa se u čvoru *n* (pre slanja ovog paketa) preko sledeće jednačine:

$$MF_n = \begin{cases} \sqrt{\dfrac{\left|N_n \cap N_n^p\right|}{\left|N_n \cup N_n^p\right|}}, & za\ N_n \cup N_n^p \neq 0; \\ 0, & inače. \end{cases} \tag{3.5}$$

U ovoj jednačini sa N_n je označen trenutni skup suseda čvora *n*, dok je sa N_n^p označen skup suseda čvora *n* u trenutku prethodnog slanja *Hello* paketa. Prema ovoj jednačini faktor mobilnosti će biti maksimalan (jednak 1) ukoliko se ovi skupovi potpuno preklapaju, što znači da nije došlo do promene skupa suseda, odnosno da je taj deo mreže približno stacionaran. U tim uslovima će putanja biti stabilnija, pa joj iz tog razloga treba dati prioritet. S druge strane, faktor mobilnosti će biti minimalan (jednak 0) ukoliko se ova dva skupa uopšte ne preklapaju, što znači da se topologija tog dela mreže izuzetno brzo menja, pa je velika mogućnost prekida linkova. Zbog toga susede sa malim faktorom mobilnosti treba izbegavati pri izboru najbolje putanje do odredišta.

Parametar BF_n označava faktor propusnog opsega susednog čvora *n*, takođe je sadržan u *Hello* paketu i računa se pre slanja ovog paketa u čvoru *n* preko jednačine:

$$BF_n = \frac{AvailableBW_n}{MaxBW_n}. \tag{3.6}$$

U ovoj jednačini, $AvailableBW_n$ predstavlja dostupni propusni opseg čvora *n*, dok $MaxBW_n$ označava maksimalni propusni opseg za čvor *n*. Maksimalni propusni opseg čvora određuju karakteristike bežičnog medijuma za prenos i identičan je za sve čvorove u mreži, a dostupni propusni opseg čvora se može izračunati kao razlika maksimalnog propusnog opsega i iskorišćenog propusnog opsega za susedni čvor *n*. Iskorišćeni propusni opseg ($UsedBW_n$) se računa u unapred definisanom vremenskom intervalu *T*, na osnovu sledeće jednačine:

$$UsedBW_n = \frac{k \cdot S_B \cdot 8}{T}. \tag{3.7}$$

U ovoj jednačini *k* predstavlja ukupan broj paketa koje je čvor *n* poslao ili primio unutar intervala *T*, a S_B predstavlja veličinu paketa u bajtima. Ukoliko nema poslatih i primljenih paketa u tom intervalu, iskorišćeni propusni opseg biće jednak 0. Tada je dostupni propusni opseg jednak maksimalnom, pa je i faktor propusnog opsega maksimalan (jednak 1). Na ovaj način se prednost daje čvorovima sa najmanjom zauzetošću propusnog opsega, pa se tako izbegavaju zagušenja u mreži, a samim tim i prevelika kašnjenja i gubici paketa. Ukoliko iskorišćeni propusni opseg dostigne maksimalni propusni opseg za čvor *n*, dostupni propusni opseg (ujedno i faktor propusnog opsega) će imati minimalnu vrednost (jednaku 0). Na ovaj način se destimuliše izbor putanja preko suseda koji su preopterećeni.

Parametar $R_{c,n}$ predstavlja nagradu (*Reward*) za preduzetu akciju, odnosno za izbor putanje preko susednog čvora *n*. Računa se preko sledeće jednačine:

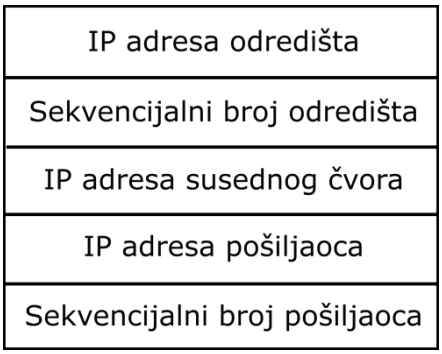
$$R_{c,n} = \begin{cases} 1, & c \in N_d; \\ 0, & inače. \end{cases} \tag{3.8}$$

U ovoj jednačini N_d predstavlja skup suseda odredišnog čvora *d*. Nagrada će biti jednaka 1 ako je čvor primio *Hello* paket od odredišnog čvora, u suprotnom će biti jednaka 0. Na ovaj način se nagrađuju direktne putanje do odredišnog čvora, koje su gotovo uvek najbolja moguća solucija za slanje paketa. Biranjem direktnih putanja smanjuju se kašnjenja paketa u mreži i povećava se verovatnoća uspešne isporuke paketa do odredišta.

Konačno, u relaciji (3.3) $max_{y \in Nei(n)} Q_n(d,y)$ predstavlja maksimalnu Q-vrednost koju ima susedni čvor *n* za putanju do odredišnog čvora *d*, preko jednog od svojih susednih čvorova *y*. Ova vrednost je, zajedno sa faktorom mobilnosti i faktorom propusnog opsega, sadržana u *Hello* paketu koji je trenutni čvor *c* primio od susednog čvora *n*. Uzimajući sve u obzir, QLAODV obezbeđuje izbor kraćih, stabilnijih i manje opterećenih putanja.

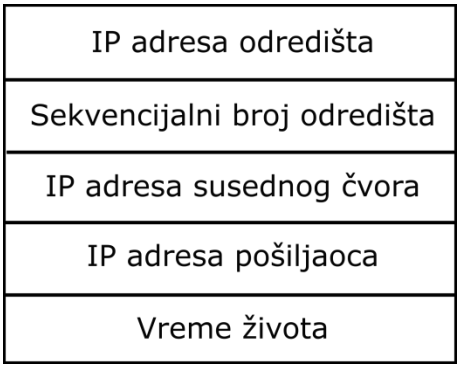
U visoko dinamičkim mrežama postoji mogućnost da putanja naučena iz lokalne komunikacije vrlo brzo nije aktivna, iako je Q-vrednost prvog linka na putanji relativno visoka. Kako bi sprečio korišćenje takve putanje, QLAODV protokol uključuje mehanizam provere nove putanje, u slučaju pojave putanje sa većom Q-vrednošću od trenutno korišćene. To znači da izvorni čvor neće odmah promeniti putanju kojom šalje pakete, već prvo ispituje novu putanju slanjem *unicast* kontrolnog paketa sa zahtevom za promenu putanje (*Route Change Request,* RCNG-REQ) preko suseda koji ima veću Q-vednost.

Kako bi uspešno stigao do odredišta, ovaj paket mora da sadrži IP adresu i sekvencijalni broj odredišnog čvora, kao i IP adresu sledećeg hopa na putanji ka odredištu. Paket mora da sadrži i IP adresu i sekvencijalni broj izvornog čvora, da bi odredišni čvor znao ko je pošiljalac. Struktura RCNG-REQ paketa prikazana je na slici 3.13.



Slika 3.13. Struktura RCNG-REQ paketa kod QLAODV protokola

Međučvorovi prosleđuju RCNG-REQ paket prateći svoje Q-tabele, sve dok paket ne stigne do odredišnog čvora. Odredišni čvor ukoliko primi RCNG-REQ paket, šalje odgovor za promenu putanje (*Route Change Reply*, RCNG-REP) ka izvornom čvoru, istom putanjom kojom je primio RCNG-REQ. Ovaj paket ima sličnu strukturu kao RCNG-REQ, s tim što umesto sekvencijalnog broja izvornog čvora (koji više nije neophodan) sadrži vreme života paketa (*lifetime*). Struktura RCNG-REP paketa prikazana je na slici 3.14.



Slika 3.14. Struktura RCNG-REP paketa kod QLAODV protokola

Međučvorovi po prijemu RCNG-REP paketa ažuriraju svoje Q-tabele i prosleđuju ovaj paket ka izvornom čvoru. Ukoliko RCNG-REP paket uspešno stigne do izvornog čvora u predviđenom vremenu, to znači da je nova putanja ispravna. Tek po prijemu RCNG-REP paketa, ukoliko nije isteklo vreme života paketa, izvorni čvor ažurirajući svoju Q-tabelu menja putanju kojom šalje podatke i na taj način smanjuje rizik od gubitka paketa usled pojave kratkotrajnih kvalitetnih putanja. Ukoliko čvor ne primi RCNG-REP paket u predviđenom vremenu, resetovaće Q-vrednost za putanju koju je proveravao na 0. Sprečavanjem korišćenja neaktivnih putanja, smanjuje se potreba za pokretanjem procedure otkrivanja putanje, a samim tim se smanjuje overhed i zagušenje u mreži.

Za simulacionu analizu i testiranje QLAODV protokola autori koji su ga predložili koristili su NS-2 simulator. Pokazali su da QLAODV daje bolje rezultate od poređenih protokola u pogledu PDR, overheda, broja grešaka pri prenosu, E2ED i broja hopova. Protokoli su testirani uz varijabilne maksimalne dozvoljene brzine kretanja i gustine vozila.

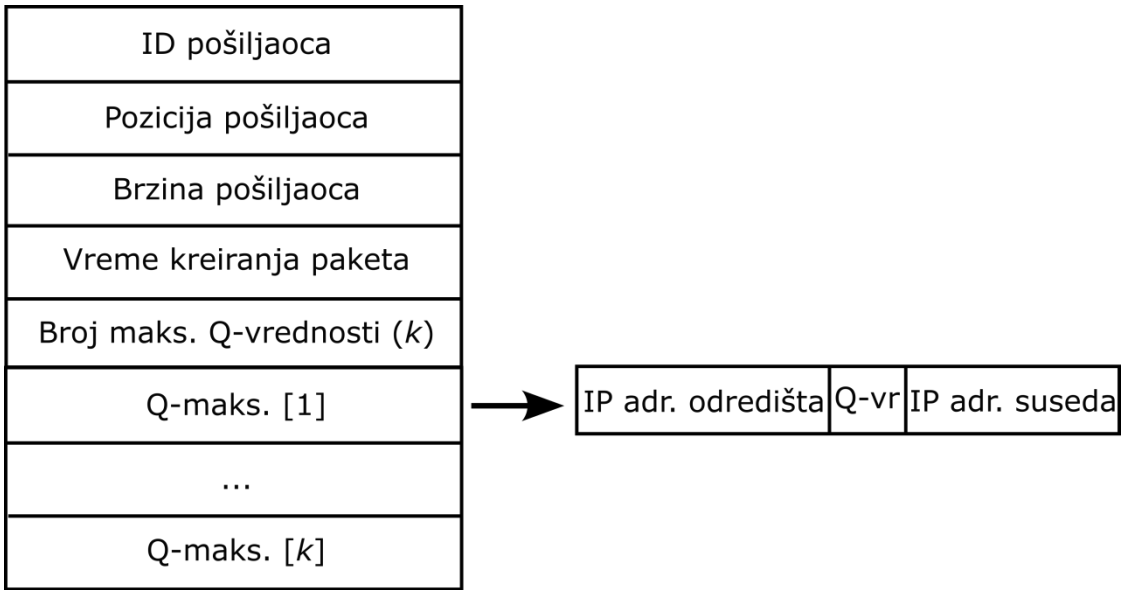
3.6.2. ARPRL protokol

ARPRL protokol predstavlja još jedan vrlo značajan protokol baziran na QL, koji dodatno unapređuje mrežne performanse VANET mreža. Protokol podrazumeva da svaki čvor održava dve tabele – tabelu suseda i Q-tabelu. Tabela suseda se koristi za smeštanje dinamičkih informacija o

susedima i ažurira se razmenom *Hello* kontrolnih paketa. Slično QLAODV protokolu, svaki čvor održava i tabelu Q-vrednosti za sve moguće putanje ka svim odredišnim čvorovima u mreži i na osnovu tih vrednosti bira putanju kojom će slati podatke. Uticajni faktori pri računanju Q-vrednosti u ovom protokolu su relativne brzine kretanja vozila, pouzdanost i stabilnost linkova u mreži. Osnovna razlika u odnosu na QLAODV je što se ove Q-vrednosti, pored ažuriranja putem periodične razmene *Hello* kontrolnih paketa između susednih čvorova, ažuriraju po prijemu paketa podataka i po prijemu povratne informacije o gubitku paketa sa MAC podsloja.

Protokol funkcioniše tako što čvor kada želi da pošalje podatke ka odredištu, najpre proverava u svojoj Q-tabeli da li ima Q-vrednost različitu od 0 za neku putanju do odredišta i ako ima šalje pakete putanjom sa najvećom Q-vrednossćću. Ukoliko nema, inicira proceduru pronalaženja putanje slanjem difuznih *Learning Probe Request* (LPREQ) kontrolnih paketa. Međučvorovi prosleđuju ove pakete dok ne stignu do odredišnog čvora. Kada odredišni čvor dobije LPREQ paket, odgovara slanjem *unicast Learning Probe Reply* (LPREP) paketa istom putanjom kojom je primio LPREQ paket. Ova procedura je veoma slična proceduri otkrivanja putanje kod AODV protokola.

Prvi način ažuriranja Q-vrednosti kod ovog protokola je periodičnom razmenom *Hello* kontrolnih paketa. Struktura ovih paketa prikazana je na slici 3.15.



Slika 3.15. Struktura *Hello* paketa kod ARPRL protokola

Svaki paket sadrži identifikacioni broj (ID) čvora koji šalje paket, kao i poziciju i brzinu kretanja ovog čvora. Pozicija i brzina kretanja vozila dostupni su preko *Global Positioning System* (GPS) modula kojim su opremljena sva vozila. Paket takođe sadrži vreme kreiranja *Hello* paketa, niz maksimalnih Q-vrednosti koje čvor ima do svakog odredišta u mreži, kao i broj ovih vrednosti (k). Uz svaku maksimalnu Q-vrednost dostupni su i podaci o IP adresama odredišta i sledećeg hopa (susednog čvora) na putanji ka odredištu. Podaci o sledećem hopu su bitni zbog izbegavanja stvaranja petlji u rutiranju. Ako čvor ne primi *Hello* paket od određenog suseda u okviru unapred definisanog vremenskog intervala, sve Q-vrednosti za putanje koje vode preko tog suseda biće resetovane na 0. U suprotnom, kada primi *Hello* paket od susednog čvora n, čvor c ažurira Q-vrednosti do svih odredišnih čvorova d prateći jednačinu (3.3), kao kod QLAODV protokola, s tim što se parametri $\alpha_{c,n}$, $\gamma_{c,n}$ i $R_{c,n}$ računaju na drugačiji način. Stepen učenja ($\alpha_{c,n}$) računa se na osnovu sledeće jednačine:

$$\alpha_{c,n} = max\left(0{,}2;\ \frac{||v_c|-|v_n||}{v_{max}-v_{min}}\right). \tag{3.9}$$

U ovoj jednačini je definisano da $\alpha_{c,n}$ ne može biti manje od 0,2 kako ne bi došlo do presporog ažuriranja Q-vrednosti. Parametri v_c i v_n označavaju brzine kretanja trenutnog čvora *c* i njegovog suseda *n*, dok v_{max} i v_{min} označavaju maksimalnu i minimalnu definisanu brzinu kretanja čvorova u mreži. Iz jednačine (3.9) je jasno da će $\alpha_{c,n}$ biti veće sa povećanjem razlike brzina kretanja čvorova u mreži, pa će se i Q-vrednosti brže ažurirati. Ukoliko nema velike razlike u brzini kretanja vozila, Q-vrednosti će se ažurirati sporije. Cilj je da se brzina ažuriranja Q-vrednosti uskladi sa brzinom promena u mrežnoj topologiji.

Diskontni faktor ($\gamma_{c,n}$) kod ARPRL protokola računa se na osnovu sledeće jednačine:

$$\gamma_{c,n} = \begin{cases} \frac{\sum_{n=1}^{N} R_{c,n}}{N}, N \neq 0; \\ 0, \quad N = 0. \end{cases} \tag{3.10}$$

U ovoj jednačini *N* predstavlja skup svih suseda čvora *c*. Primetno je da će $\gamma_{c,n}$ biti veće što je prosečna nagrada suseda čvora *c* veća. To znači da će Q-vrednosti za ovaj čvor biti veće ako njegovo okruženje karakterišu visoke nagrade, pa će se stimulisati izbor ovog čvora za sledeći hop prilikom slanja podataka ka odredištu. Nagrada za nekog suseda *n* računa se preko sledeće jednačine:

$$R_{c,n} = 100 + HMRR_{c,n} + LET_{c,n}. \tag{3.11}$$

Parametar $HMRR_{c,n}$ predstavlja stepen uspešno primljenih *Hello* paketa (*Hello Message Reception Ratio*) i računa se na osnovu sledeće jednačine:

$$HMRR_{c,n} = \begin{cases} 100 * \frac{CNT_r(c,n)}{CNT_s(n)}, \quad CNT_s(n) \geq 15; \\ 100 * \frac{CNT_r(c,n)}{CNT_s(n)} * \left(1 - \left(\frac{1}{2}\right)^{CNT_s(n)}\right), inače. \end{cases} \tag{3.12}$$

Parametar $CNT_r(c,n)$ označava koliko je *Hello* paketa poslatih od čvora *n* čvor *c* uspešno primio, a parametar $CNT_s(n)$ označava koliko je ukupno *Hello* paketa čvor *n* poslao ka čvoru *c*. Primetno je da će $HMRR_{c,n}$ biti veći što je veći broj uspešno primljenih *Hello* paketa od nekog suseda, pa samim tim raste i nagrada za putanju preko tog suseda. Cilj je stimulisati izbor pouzdanijih putanja, sa manjim brojem izgubljenih paketa. Ovaj parametar će biti nešto manji ako je broj poslatih *Hello* paketa ispod 15, jer su takve putanje skorijeg veka pa se sa manjom sigurnošću može tvrditi da su pouzdane.

Još jedan parametar koji figuriše u funkciji nagrade je $LET_{c,n}$, koji označava faktor stabilnosti linka od čvora *n* do čvora *c* i računa se na osnovu sledeće jednačine:

$$LET_{c,n} = \begin{cases} 100, \ za \ A = 0 \ i \ B = 0; \\ min\left(100, \frac{-(AB + CD) + \sqrt{(A^2 + C^2) \cdot R^2 - (AD - BC)^2}}{A^2 + B^2}\right), inače. \end{cases} \tag{3.13}$$

Parametri *A*, *B*, *C* i *D* računaju se preko sledećih jednačina:

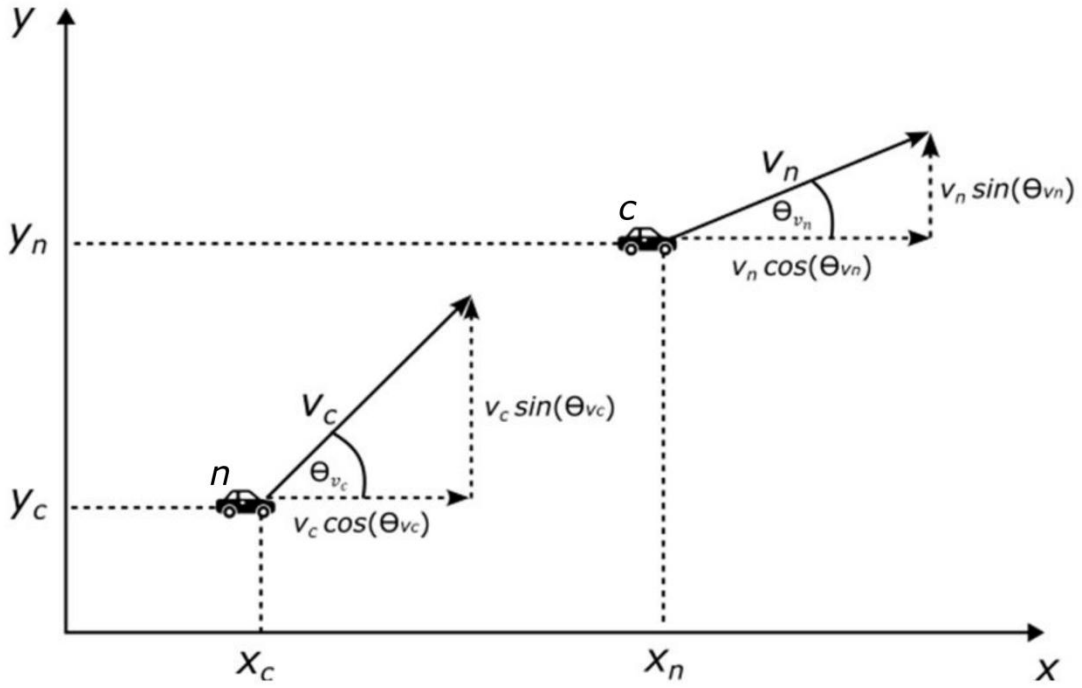
$$A = v_c \, cos(\Theta_{v_c}) - v_n cos(\Theta_{v_n}), \tag{3.14}$$

$$B = x_c - x_n, \tag{3.15}$$

$$C = v_c\, sin(\Theta_{v_c}) - v_n sin(\Theta_{v_n}), \tag{3.16}$$

$$D = y_c - y_n. \tag{3.17}$$

U prethodnim jednačinama $v_c\, cos(\Theta_{v_c})$ i $v_n cos(\Theta_{v_n})$ predstavljaju projekcije brzina čvorova *c* i *n* na *x* osu, dok $v_c\, sin(\Theta_{v_c})$ i $v_n sin(\Theta_{v_n})$ predstavlju projekcije brzina ovih čvorova na *y* osu, respektivno. Parametri x_c, x_n, y_c i y_n predstavljaju koordinate čvorova *c* i *n* u dvodimenzionalnom koordinatnom sistemu. Na slici 3.16 grafički su predstavljene ove veličine. Čvorovi sa većom stabilnošću će imati veće Q-vrednosti, pa će samim tim imati prednost pri izboru sledećeg čvora na putanji ka odredištu. Cilj je dati prednost čvorovima koji se kreću u istom (ili sličnom) smeru i čije je rastojanje što manje.



Slika 3.16. Grafički prikaz određivanja parametara za računanje faktora stabilnosti linka kod ARPRL protokola

Drugi način ažuriranja Q-vrednosti je po prijemu paketa podataka od izvornog čvora. Kada trenutni čvor *c* primi pakete podataka od izvornog čvora *s*, preko susednog čvora *n*, ažuriraće odgovarajuću Q-vrednost za ovu putanju na osnovu sledeće jednačine:

$$Q_c(s,n) \leftarrow (1-\alpha_{c,n})Q_c(s,n) + \alpha_{c,n} \cdot (R_{c,n} + \gamma_{c,n} \cdot max_{y \epsilon Nei(n)} Q_n(s,y)). \tag{3.18}$$

Parametri $\alpha_{c,n}$, $\gamma_{c,n}$ i $R_{c,n}$ su definisani relacijama (3.9), (3.10) i (3.11). S obzirom da se Q-vrednosti povećavaju sa svakim ažuriranjem (nagrada je dosta veća od 1), na ovaj način se stimuliše izbor putanja sa većom uspešnošću isporuke podataka.

Treći način ažuriranja Q-vrednosti kod ARPRL protokola je po prijemu povratnih informacija o gubitku paketa sa MAC podsloja (koristi se IEEE 802.11 MAC). Kada primi informaciju o gubitku paketa od susednog čvora *n*, trenutni čvor *c* ažurira Q-vrednosti za putanje do svih destinacija d_i preko ovog susednog čvora prateći sledeću jednačinu:

$$Q_c(d_i,n) \leftarrow 0{,}5 \cdot Q_c(d_i,n). \tag{3.19}$$

Na ovaj način se kažnjavaju putanje na kojima dolazi do gubitaka paketa, pa će biti manja verovatnoća njihovg izbora za optimalnu putanju do odredišta.

Autori koji su predložili ovaj protokol testirali su ga u QualNet simulatoru. Simulacioni rezultati su pokazali da ARPRL daje bolje performanse od protokola sa kojima je poređen u pogledu prosečnog PDR, E2ED i broja hopova pri varijabilnim gustinama vozila, maksimalnim dozvoljenim brzinama vozila i količini generisanog saobraćaja.

4. PREDLOG NOVOG RL BAZIRANOG PROTOKOLA RUTIRANJA ZA VANET MREŽE

Sa nedavnim napretkom u oblasti mašinskog učenja, posebno RL baziranih algoritama, raste interes za unapređenje procesa rutiranja u VANET mrežama integracijom ove inovativne tehnologije u protokole rutiranja. Pokazalo se da ovaj pristup doprinosi značajnom unapređenju mrežnih performansi u poređenju sa tradicionalnim protokolima rutiranja, koji ne uspevaju da isprate brze promene u mrežnoj topologiji. Jedan od istaknutih protokola rutiranja koji koristi RL i podrazumeva V2V komunikaciju je QLAODV, opisan u poglavlju 3.6.1. Kao što je već opisano, ovaj protokol predstavlja unapređenje AODV protokola i koristi QL za pronalaženje optimalne putanje za prosleđivanje paketa. Problem njegove primene prvenstveno se ogleda u visokim gubicima paketa, naročito u gustim mrežama. Takođe, prisutna su velika kašnjenja paketa i visok džiter, čak i u mrežama sa manjim brojem vozila. ARPRL predstavlja još jedan uspešan V2V protokol (detaljnije opisan u poglavlju 3.6.2), koji koristi QL za rutiranje u VANET mrežama. Njegov glavni nedostatak su loše mrežne performanse u gotovo stacionarnim mrežama, što je česta pojava u gustim urbanim sredinama kada se vozila kreću sporo zbog gužvi u saobraćaju. Takođe, u manjim gradskim mrežama ARPRL daje velika kašnjenja paketa i džiter.

Kako bi se dalje optimizovao proces V2V komunikacije i unapredile mrežne performanse VANET mreža, razvijen je novi dinamički protokol rutiranja baziran na QL za urbane VANET mreže – Q-DRAV (Bugarčić i ostali, 2024). Ovaj protokol je u 6. poglavlju upoređen sa AODV protokolom, koji je izabran za predstavnika tradicionalnih protokola rutiranja, kao i sa QLAODV i ARPRL protokolima, koji su odabrani za predstavnike RL baziranih protokola rutiranja (oba pokazuju vrlo dobre mrežne performanse u VANET mrežama koje podrazumevaju samo V2V komunikaciju). Međutim, pod različitim saobraćajnim uslovima, ovi protokoli i dalje ne uspevaju adekvatno da ublaže degradaciju mrežnih performansi uzrokovanu visokom dinamičnošću čvorova u VANET okruženju. Iz tog razloga je bilo potrebno razviti novi protokol, koji prevazilazi ove nedostatke. Pri izboru optimalne putanje Q-DRAV koristi QL algoritam, a cilj protokola je minimizacija gubitaka paketa usled prekida mrežnih linkova i povećanje ostvarenog protoka paketa, uz istovremeno smanjenje kašnjenja paketa i džitera.

4.1. Metodologija razvoja

Protokol koji je poslužio kao prva inspiracija za razvoj novog protokola bio je QLAODV, koji je poboljšao performanse popularnog AODV protokola i prilagodio ga VANET okruženju koristeći QL. Sledeći protokol baziran na QL koji je razmatran pri razvoju novog protokola je ARPRL, koji je dodatno unapredio performanse V2V rutiranja u VANET mrežama. Zajedničko za ova dva protokola je da koriste QL za optimizaciju procesa rutiranja i ne zahtevaju spoljnu infrastrukturu (uključuju samo V2V komunikaciju). Cilj novog protokola je da dodatno unapredi mrežne performanse VANET mreža, uključivanjem dodatnih i modifikovanjem postojećih faktora koji utiču na QL proces.

Tabela 4.1 prikazuje pregled faktora koji utiču na optimizaciju procesa rutiranja u QLAODV, ARPRL i predloženom Q-DRAV protokolu rutiranja. QLAODV protokol uzima u obzir mobilnost

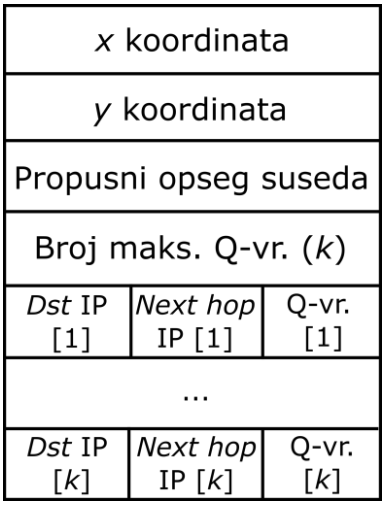
vozila, dostupnost propusnog opsega vozila i postojanje direktne veze ka odredištu. ARPRL protokol prilikom izbora optimalne putanje za slanje podataka uzima u obzir brzinu kretanja vozila, stabilnost i pouzdanost veze ka susednom vozilu i procenat gubitka paketa. S druge strane, Q-DRAV protokol uzima u obzir dostupnost propusnog opsega susednog vozila, postojanje direktne veze do odredišta, pouzdanost veze između susednih vozila, gubitke paketa na MAC podsloju, rastojanje između susednih vozila, dostupnost propusnog opsega putanje do odredišta, broj hopova na putanji do odredišta, kašnjenje paketa, potencijalno stvaranje petlji prilikom kreiranja putanje i gustinu vozila u mreži. Uzimanjem u obzir višestrukih metrika, Q-DRAV pokušava sveobuhvatnije da sagleda trenutno stanje mreže i u skladu sa tim dodatno optimizuje proces rutiranja.

Tabela 4.1. Uticajni faktori u QLAODV, ARPRL i Q-DRAV protokolima rutiranja

Parametar	*QLAODV*	*ARPRL*	*Q-DRAV*
Mobilnost vozila (na bazi suseda)	✓		
Dostupnost propusnog opsega vozila	✓		✓
Postojanje direktnog linka ka odredištu	✓		✓
Brzina vozila (na bazi GPS-a)		✓	
Stabilnost linka		✓	
Pouzdanost linka		✓	✓
Gubici paketa		✓	✓
Rastojanje između vozila (na bazi GPS-a)			✓
Dostupnost propusnog opsega putanje			✓
Broj hopova			✓
Kašnjenje paketa			✓
Kreiranje petlji			✓
Gustina vozila			✓

4.2. Opis Q-DRAV protokola

Za izbor optimalne putanje za slanje paketa, Q-DRAV protokol koristi QL algoritam koji podrazumeva tri metode za ažuriranje Q-vrednosti. Prva i osnovna metoda ažuriranja odvija se prilikom razmene *Hello* kontrolnih paketa između susednih čvorova (vozila). Druga metoda ažuriranja Q-vrednosti inicira se prilikom prijema povratnih informacija o gubitku paketa sa MAC podsloja, dok se treća metoda ažuriranja vrši korišćenjem paketa za ispitivanje putanje (*Route Probe Packets*, RPP).



Slika 4.1. Struktura *Hello* paketa kod Q-DRAV protokola

Kako bi blagovremeno ispratila česte promene u mrežnoj topologiji, koje su karakteristične za VANET mreže, vozila periodično razmenjuju *Hello* kontrolne pakete sa svojim susedima. Pretpostavlja se da su sva vozila opremljena GPS uređajima koji obezbeđuju preciznu lokaciju vozila.

Prva metoda za ažuriranje Q-vrednosti odvija se nakon što vozilo primi *Hello* paket od svog suseda. *Hello* paketi sadrže sve potrebne informacije za ažuriranje Q-vrednosti, na osnovu kojih vozila procenjuju kvalitet putanja koje vode ka odredištu preko nekog od svojih suseda. Ove informacije uključuju koordinate suseda po *x* i *y* osi, dostupni propusni opseg suseda (*Neighbor Bandwidth*, NB), broj odredišta (*k*) za koja sused ima maksimalne Q-vrednosti u svojoj Q-tabeli, kao i sekvencu tih maksimalnih Q-vrednosti zajedno sa informacijama o IP adresama odredišta (*Dst* IP) i sledećeg hopa na putanji do odredišta (*Next hop* IP). Struktura *Hello* paketa prikazana je na slici 4.1.

Kada čvor *c* primi *Hello* paket od suseda *n*, ažurira Q-vrednost za odredište *d* preko tog suseda na osnovu sledeće jednačine:

$$Q_c(d,n) \leftarrow \left(1-\alpha_{c,n}\right)\cdot Q_c(d,n) + \alpha_{c,n}\cdot\left(R_{c,n} + \gamma_{c,n}\cdot max_{y\epsilon Nei(n)}Q_n(d,y)\right). \tag{4.1}$$

Ključni korak kod ažuriranja Q-vrednosti je definisanje vrednosti parametara $\alpha_{c,n}$, $\gamma_{c,n}$ i $R_{c,n}$, preko kojih je potrebno uključiti željene uticajne faktore u QL proces. Za inicijalno ažuriranje Q-vrednosti, stepen učenja ($\alpha_{c,n}$) ima konstantnu vrednost. Eksperimentalnim putem je utvrđeno da se najbolji rezultati mrežnih performansi dobijaju kada se ova vrednost postavi na 0,6. U suprotnom, $\alpha_{c,n}$ se izračunava kao:

$$\alpha_{c,n} = max(0{,}6\,;RDC_{c,n}\,;LU_{c,n}). \tag{4.2}$$

Parametar $RDC_{c,n}$ (*relative distance change*) predstavlja relativnu promenu rastojanja između trenutnog čvora *c* i njegovog suseda *n*. Izračunava se preko sledeće jednačine:

$$RDC_{c,n} = \sqrt{\left|\frac{\Delta d_{c,n}}{\Delta d_{max}}\right|}. \tag{4.3}$$

U prethodnoj jednačini, $\Delta d_{c,n}$ predstavlja promenu rastojanja između posmatranog čvora *c* i njegovog suseda *n* tokom perioda pojedinačnog ažuriranja Q-vrednosti (period između dva uzastopna slanja *Hello* paketa), dok Δd_{max} predstavlja maksimalnu moguću promenu rastojanja tokom tog intervala. Na ovaj način, stepen učenja između čvorova koji brže menjaju udaljenost se ubrzava, jer se pretpostavlja da postoje značajnije promene u kvalitetu veze između njih. Nelinearna funkcija kvadratnog korena se koristi za dodatno isticanje malih promena udaljenosti. Kako bi se brže ažurirale putanje sa manjom pouzdanošću, parametar $\alpha_{c,n}$ uzima u obzir i kriterijum pouzdanosti linka preko koeficijenta $LU_{c,n}$ (*link unreliability*), koji se računa na osnovu sledeće jednačine:

$$LU_{c,n} = \sqrt{1-(1-2^{-SH_{c,n}})\cdot\frac{RH_{c,n}}{SH_{c,n}}}. \tag{4.4}$$

U ovoj jednačini parametar $RH_{c,n}$ (*received hello*) predstavlja broj *Hello* paketa koje je čvor *c* primio od čvora *n*, a parametar $SH_{c,n}$ (*sent hello*) broj *Hello* paketa koje je čvor *c* poslao ka čvoru *n*. Kod nestabilnih linkova $RH_{c,n}$ je značajno manji od $SH_{c,n}$, tako da je koeficijent $LU_{c,n}$ blizu 1. Ovo utiče na povećanje $\alpha_{c,n}$, odnosno ažuriranje Q-vrednosti se ubrzava. Prema relaciji (4.2), uvek se bira maksimum između dva pomenuta kriterijuma ($RDC_{c,n}$ i $LU_{c,n}$). Takođe, osigurava se da $\alpha_{c,n}$ nije manje od 0,6, kako ažuriranje Q-vrednosti ne bi bilo previše sporo. Konačni cilj je ubrzati

ažuriranje nepouzdanih putanja, dok se istovremeno održava stabilnost Q-vrednosti pouzdanih putanja.

Diskontni faktor ($\gamma_{c,n}$) se koristi za fino podešavanje Q-vrednosti. Za inicijalno ažuriranje Q-vrednosti, ovaj parametar će imati konstantnu vrednost od 0,3. Ova vrednost je takođe izabrana eksperimentalnim putem, nakon što je utvrđeno da daje najbolje mrežne performanse. U suprotnom, $\gamma_{c,n}$ se izračunava preko sledeće jednačine:

$$\gamma_{c,n} = 0{,}25 + 0{,}1 \cdot NB_n. \tag{4.5}$$

Prednost prilikom odabira sledećeg hopa daje se susednim čvorovima sa većim dostupnim propusnim opsegom čvora, putem parametra NB_n. Ovaj parametar se nalazi u *Hello* paketu koji šalje susedni čvor n i izračunava se kao:

$$NB_n = 1 - RUB_n. \tag{4.6}$$

gde RUB_n (*relative used bandwidth*) predstavlja relativni iskorišćeni propusni opseg susednog čvora n i izračunava se na sledeći način:

$$RUB_n = \frac{UsedBW_n}{MaxBW_n}. \tag{4.7}$$

U prethodnoj jednačini, $MaxBW_n$ predstavlja maksimalan propusni opseg koji je dostupan čvoru n (definisan karakteristikama bežičnog medijuma za prenos, isti je za sve čvorove), dok $UsedBW_n$ predstavlja iskorišćeni propusni opseg čvora n. Eksperimentalna istraživanja su pokazala da se najbolji rezultati postižu kada $\gamma_{c,n}$ varira oko vrednosti 0,3. Pošto NB_n može da se kreće od 0 do 1, $\gamma_{c,n}$ uzima vrednosti od 0,25 (kada je propusni opseg potpuno zauzet) do 0,35 (kada je propusni opseg potpuno slobodan). Na ovaj način se izbegavaju putanje koje idu preko preopterećenih suseda i smanjuju se zagušenja u mreži.

Parametar $R_{c,n}$ predstavlja nagradu za preduzetu akciju, odnosno za odabir odgovarajućeg suseda za prosleđivanje paketa ka odredištu. Cilj je dati prioritet kraćim putanjama, dok se istovremeno izbegava kreiranje petlji. Ako čvor primi *Hello* paket direktno od odredišta, ovoj putanji svakako treba dati prioritet, tako da se Q-vrednost automatski postavlja na 1 (odnosno na maksimum). U drugim slučajevima, $R_{c,n}$ se definiše kao:

$$R_{c,n} = \begin{cases} 0{,}6\,, & 2\ hopa; \\ -0{,}5, & petlja; \\ 0, & inače. \end{cases} \tag{4.8}$$

Prvi slučaj se javlja kada je odredišni čvor ujedno i sledeći hop u Q-tabeli susednog čvora n. Takvoj putanji treba dati nagradu, s obzirom na to da ima samo dva hopa do odredišta. Drugi slučaj se dešava kada je trenutni čvor c ujedno i sledeći hop u tabeli rutiranja susednog čvora n (što znači da će se stvoriti petlja). Takvu putanju treba kazniti, tj. njena Q-vrednost treba da se smanji. U ostalim slučajevima, nema ni nagrade ni kazne.

Poslednji član u jednačini (4.1), $max_{y \epsilon Nei(n)} Q_n(d,y)$, predstavlja maksimalnu Q-vrednost koju u svojoj Q-tabeli ima čvor n za putanju do odredišnog čvora d, preko nekog od svojih suseda y (*Nei(n)* predstavlja skup suseda čvora n). Na ovaj način se u izbor optimalne putanje uključuje Q-vrednost drugog hopa od trenutnog čvora c, ka odredišnom čvoru d. Algoritam 1 prikazuje pseudokod za ažuriranje Q-vrednosti prilikom razmene *Hello* kontrolnih paketa između susednih čvorova.

Algoritam 1. Ažuriranje Q-vrednosti prilikom razmene Hello kontrolnih paketa

Oznake:

V_c: Trenutno vozilo
V_n: Susedno vozilo
x_n: x koordinata vozila V_n
y_n: y koordinata vozila V_n
NB_n: Dostupan propusni opseg za vozilo V_n
N: Broj različitih vozila u Q-tabeli
$Q_{Vn}(V_i,V_j)$: Q-vrednost u Q-tabeli vozila V_n za putanju do odredišnog vozila V_i preko susednog vozila V_j

Kada istekne *Hello* tajmer susednog vozila V_n, ovo vozilo:

1: Izračunava NB_n na osnovu (4.6)
2: Dodaje x_n, y_n, NB_n u *Hello* paket
3: **for** i = 1 to N **do**
4: **if** V_n ima $Q_{Vn}(V_i,V_j) \neq 0$ za bilo kog suseda V_j **then**
5: Ažurira $V_{max} \leftarrow \text{argmax}_{Vj}[Q_{Vn}(V_i,V_j)]$
6: Dodaje IP adresu vozila V_i, IP adresu vozila V_{max} i $Q_{Vn}(V_i,V_{max})$ u *Hello* paket
7: Inkrementira broj V_{max}
8: **end if**
9: **end for**
10: Dodaje broj V_{max} u *Hello* paket
11: Difuzno šalje *Hello* paket
12: Resetuje *Hello* tajmer

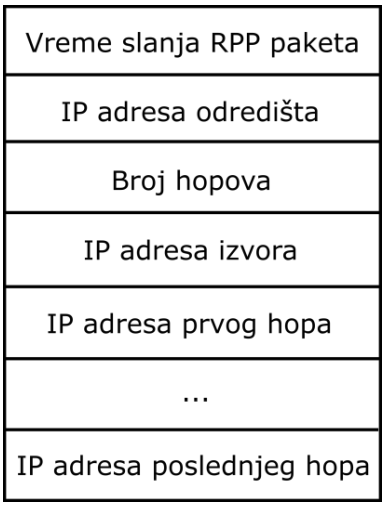
Kada primi *Hello* paket od susednog vozila V_n, trenutno vozilo V_c:

13: Očitava x_n, y_n, NB_n, broj V_{max} iz *Hello* paketa
14: Ažurira $M \leftarrow$ broj V_{max} iz *Hello* paketa
15: **for** j = 1 to M **do**
16: Očitava IP adresu vozila V_j, IP adresu vozila V_{max} i $Q_{Vn}(V_j,V_{max})$ iz *Hello* paketa
17: **if** $V_j = V_n$ **then**
18: Ažurira $Q_{Vc}(V_j,V_n) \leftarrow 1$
19: **else if** $Q_{Vc}(V_j,V_n)$ ne postoji u Q-tabeli **then**
20: Ažurira $\alpha_{c,n} \leftarrow 0.6$
21: Ažurira $\gamma_{c,n} \leftarrow 0.3$
22: Izračunava $R_{c,n}$ na osnovu (4.8)
23: Ažurira $Q_{Vc}(V_j,V_n)$ na osnovu (4.1)
24: **else**
25: Izračunava $\alpha_{c,n}$ na osnovu (4.2), $\gamma_{c,n}$ na osnovu (4.5) i $R_{c,n}$ na osnovu (4.8)
26: Ažurira $Q_{Vc}(V_j,V_n)$ na osnovu (4.1)
27: **end if**
28: **end for**

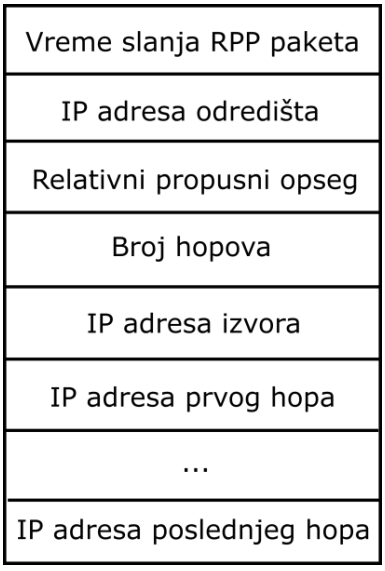
Druga metoda za ažuriranje Q-vrednosti dešava se po prijemu povratnih informacija sa MAC podsloja o gubitku paketa. Cilj je kazniti putanje na kojima dolazi do gubitaka, smanjujući time verovatnoću njihove selekcije pri slanju paketa ka odredištu. Ako čvor c primi informaciju sa MAC podsloja o gubitku paketa poslatih ka susedom čvoru n, ažurira Q-vrednosti za putanje do svih destinacija d_i, koje idu preko tog suseda prateći sledeću jednačinu:

$$Q_c(d_i, n) \leftarrow 0{,}6 \cdot Q_c(d_i, n). \tag{4.9}$$

Treća metoda za ažuriranje Q-vrednosti uključuje korišćenje RPP paketa. Na ovaj način se uzima u obzir kvalitet cele putanje, a ne samo linka prema susedu, kao u prethodna dva slučaja. Iako čvor uvek šalje podatke preko suseda sa najvećom Q-vrednošću, moguće je da celokupna putanja do odredišta preko tog suseda nije optimalna. Stoga, čvor periodično šalje RPP pakete odredištu preko svih suseda čije Q-vrednosti nisu manje od 95% maksimalne Q-vrednosti.



Slika 4.2. Struktura RPP paketa kod Q-DRAV protokola



Slika 4.3. Struktura RPP-ACK paketa kod Q-DRAV protokola

Struktura RPP paketa prikazana je na slici 4.2, a RPP-ACK paketa na slici 4.3. Oba tipa paketa sadrže sledeće informacije: vreme slanja RPP paketa, IP adrese odredišta, izvora i svih međučvorova na putanji od izvora do odredišta (svaki čvor koji primi RPP paket dodaje svoju IP adresu), kao i broj hopova (*Hop Count*, HC) na putanji od izvora do odredišta. Pored ovih zajedničkih polja, RPP-ACK sadrži i informaciju o relativnom propusnom opsegu putanje (*Route Relative Bandwidth*, RRB). Odredišni čvor po prijemu RPP paketa najpre očitava vrednosti svih polja i upisuje ih u RPP-ACK paket. Zatim dodaje RRB polje (inicijalizovano na 0) i šalje paket ka izvornom čvoru istom putanjom kojom je primio RPP.

Po prijemu RPP-ACK paketa izvorni čvor računa faktor kvaliteta putanje (*Route Quality Factor*, RQF) na osnovu sledeće jednačine:

$$RQF = RTD \cdot HC \cdot RRB. \tag{4.10}$$

U ovoj jednačini RTD (*round trip delay*) predstavlja vreme proteklo od slanja RPP paketa do prijema RPP-ACK paketa. Vrednost HC je sadržana u RPP paketu (kao što je prikazano na slici 4.2) i najpre se postavlja na 0, a onda se pri svakom prijemu ovog paketa (bilo od strane međučvorova ili odredišnog čvora) inkrementira za 1. Na kraju putanje, odredišni čvor očitava ovu vrednost i upisuje je u RPP-ACK paket. Vrednost RRB parametra je sadržana u RPP-ACK paketu (kao što je prikazano na slici 4.3), a ažuriraju je čvorovi na putanji od odredišnog do izvornog čvora. Svaki put kada neki međučvor primi RPP-ACK paket, ažurira RRB polje u paketu dodajući primljenoj vrednosti svoj relativni iskorišćeni propusni opseg (koji računa prema jednačini (4.7)). Nakon ovog ažuriranja, čvor prosleđuje RPP-ACK, a ovaj proces se nastavlja sve dok paket ne stigne do izvornog čvora.

RQF parametar se koristi za odabir putanja sa manjim kašnjenjem paketa, manjim brojem hopova i manje zauzetim propusnim opsegom. To znači da RQF parametar takođe treba da bude što niži. Nakon isteka vremenskog ograničenja za prijem RPP-ACK paketa, izvorni čvor određuje koji sused ima najniži RQF i nagrađuje putanju preko tog suseda prema sledećoj relaciji:

$$Q_c(d,n) \leftarrow 1{,}1 \cdot Q_c(d,n). \tag{4.11}$$

Putanje preko suseda kojima je poslat RPP, ali RPP-ACK nije primljen unutar definisanog perioda, kažnjavaju se prema relaciji:

$$Q_c(d,n) \leftarrow 0{,}6 \cdot Q_c(d,n). \tag{4.12}$$

Algoritam 2 prikazuje pseudokod za ažuriranje Q-vrednosti prilikom razmene RPP i RPP-ACK kontrolnih paketa.

Algoritam 2. Ažuriranje Q-vrednosti prilikom razmene RPP i RPP-ACK kontrolnih paketa

Oznake:

V_c: Trenutno vozilo, izvor RPP paketa
V_d: Odredišno vozilo
V_i: Vozilo na putanji od V_c do V_d
V_n: Optimalno susedno vozilo (sa najboljom Q-vrednošću) vozila V_c
$Qmax_{Vc}(V_d,V_n)$: Maksimalna Q-vrednost u Q-tabeli vozila V_c za putanju do vozila V_d preko vozila V_n
$Q_{Vc}(V_d,V_i)$: Q-vrenost u Q-tabeli vozila V_c za putanju do vozila V_d preko susednog vozila V_i
$T(V_c)$: Trenutno vreme vozila V_c
$HC(V_c, V_d)$: Broj hopova od vozila V_c do vozila V_d
$RRB(V_c, V_d)$: Relativni propusni opseg putanje od vozila V_c do vozila V_d
RUB_{Vi}: Relativni iskorišćeni propusni opseg vozila V_i
$RQF_{Vi}(V_c,V_d)$: Faktor kvaliteta putanje od vozila V_c do vozila V_d preko vozila V_i
M: Broj poslatih RPP paketa tokom jednog RPP tajmera

Kada istekne RPP tajmer trenutnog vozila V_c, ovo vozilo:

1: Ažurira $N \leftarrow$ broj suseda vozila V_c
2: Ažurira $M \leftarrow 0$

3: **for** j = 1 to N **do**
4: **if** $Q_{Vc}(V_d,V_j) >= 0{,}95 \cdot Qmax_{Vc}(V_d,V_n)$ **then**
5: Ažurira $HC(V_c, V_d) \leftarrow 0$
6: Dodaje $T(V_c)$, IP adresu vozila V_d, $HC(V_c, V_d)$ i IP adresu vozila V_c u RPP paket
7: Šalje RPP paket prema vozilu V_d preko vozila V_j
8: Ažurira $M \leftarrow M + 1$
9: **end if**
10: **end for**

Kada primi RPP paket od trenutnog vozila V_c, vozilo V_i:

11: Inkrementira $HC(V_c, V_d)$ u RPP paketu i dodaje svoju IP adresu u RPP paket
12: Ažurira $V_{jmax} \leftarrow \text{argmax}_{Vj}[Q_{Vi}(V_d,V_j)]$
13: Šalje RPP paket prema vozilu V_d preko vozila V_{jmax}

Kada primi RPP paket od vozila V_i, odredišno vozilo V_d:

14: Očitava sve vrednosti iz RPP paketa i dodaje ih u RPP-ACK paket
15: Izračunava RUB_{Vd} na osnovu (4.7)
16: Ažurira $RRB(V_c, V_d) \leftarrow RUB_{Vd}$
17: Dodaje $RRB(V_c, V_d)$ u RPP-ACK paket
18: Šalje RPP-ACK paket prema vozilu V_c istom putanjom kojom je primilo RPP paket

Kada primi RPP-ACK paket od odredišnog vozila V_d, vozilo V_i:

19: Izvlači i očitava $RRB(V_c, V_d)$ iz RPP-ACK paketa
20: Izračunava RUB_{Vi} na osnovu (4.7)
21: Ažurira $RRB(V_c, V_d) \leftarrow RUB_{Vi} + RRB(V_c, V_d)$
22: Dodaje novo $RRB(V_c, V_d)$ u RPP-ACK paket
23: Šalje RPP-ACK prema vozilu V_c istom putanjom kojom je primilo RPP paket

Kada primi RPP-ACK paket od odredišnog vozila V_d preko vozila V_i, trenutno vozilo V_c:

24: Očitava $RRB(V_c, V_d)$ iz RPP-ACK paketa
25: Izračunava $RQF_{Vi}(V_c, V_d)$ na osnovu (4.10)
26: Ažurira $ReceivedRPP\text{-}ACK_{Vi} \leftarrow$ TRUE

Kada trenutnom vozilu V_c istekne vremenski limit za prijem RPP-ACK paketa, ovo vozilo:

27: Ažurira $V_{imin} \leftarrow \text{argmin}_{Vi}[RQF_{Vi}(V_c, V_d)]$
28: **for** i = 1 to M **do**
29: **if** $ReceivedRPP\text{-}ACK_{Vi}$ = TRUE **then**
30: **if** $V_i = V_{imin}$ **then**
31: Ažurira $Q_{Vc}(V_d,V_{imin})$ na osnovu (4.11)
32: **end if**
33: **else**
34: Ažurira $Q_{Vc}(V_d,V_i)$ na osnovu (4.12)
35: **end if**
36: **end for**

Još jedan način za poboljšanje performansi VANET mreža je sprečavanje preopterećenja mreže u slučaju velike gustine čvorova. Treba napomenuti da se sa povećanjem broja čvorova, broj poslatih *Hello* paketa i veličina svakog *Hello* paketa povećavaju. Zbog toga, Q-DRAV protokol osigurava da se *Hello* paketi šalju smanjenim intenzitetom u gustim mrežama. Prvi uslov za uzdržavanje od slanja *Hello* paketa susednom čvoru je da je *Hello* paket poslat tom susedu u prethodnom periodu slanja. Ovo sprečava uzastopno otkazivanje slanja *Hello* paketa i samim tim gubitak informacija o susedima. Drugi uslov je da je ukupan broj maksimalnih Q-vrednosti u paketu veći od definisanog praga (vrednost od 250 je usvojena u ovom protokolu, ali ju je moguće menjati u zavisnosti od veličine mreže). Smanjenje opterećenja mreže vrši se definisanjem verovatnoće slanja *Hello* paketa na sledeći način:

$$p = \begin{cases} (N_{th}/N_{av})^2, & N_{av} > N_{th}; \\ 1, & inače. \end{cases} \tag{4.13}$$

U ovoj relaciji N_{av} predstavlja prosečan broj suseda trenutnog čvora u prethodnom periodu (usvojen je period od 50 sekundi), a N_{th} je prag, tj. minimalni broj suseda za sprovođenje smanjenja opterećenja mreže (usvojen prag je 25 suseda). Oba parametra je moguće promeniti u zavisnosti od veličine i gustine mreže. *Hello* paket se šalje određenom susedu sa verovatnoćom p koja se smanjuje sa povećanjem gustine vozila, čime se smanjuje opterećenje mreže. S druge strane, zbog velikog broja drugih suseda (minimum 25), ne postoji opasnost od nemogućnosti pronalaženja putanje do odredišta i gubitka paketa podataka.

Poređenje novog Q-DRAV protokola i postojećih QLAODV, ARPRL i AODV protokola izvršeno je u NS-3 simulacionom okruženju. Iz tog razloga, u sledećem poglavlju će najpre biti dat kratak opis NS-3 simulatora kako bi se bolje razumele njegove karakteristike, struktura i elementi simulatora za modelovanje *ad hoc* mreža. Simulacioni scenariji za poređenje protokola i rezultati simulacije će detaljnije biti predstavljeni u 6. poglavlju.

5. SIMULACIONO OKRUŽENJE ZA TESTIRANJE PROTOKOLA

Simulacija je veoma važna moderna tehnologija pomoću koje je moguće ispitivati performanse modela predstavljenog skupom ulaznih podataka, algoritama i procedura. Računarska simulacija može da modelira hipotetičke i stvarne objekte na računaru, na osnovu čega se oni mogu proučavati i analizirati. Simulacije se mogu primeniti u različitim naukama, inženjeringu, ili drugim poljima primene. Takođe se mogu koristiti za pomoć u modeliranju i analizi mnogih prirodnih sistema. Mrežni simulator omogućava ispitivanje ponašanja mreže bilo putem interkonekcije mrežnih entiteta koristeći matematičke formule ili putem snimanja i reprodukcije zapažanja aktivnosti u mreži (Gupta i ostali, 2013).

Mrežni simulator omogućava korisnicima da testiraju scenarije koji su teški ili skupi za realizaciju u realnim uslovima. Posebno su korisni za testiranje novih mrežnih protokola ili za unapređenje postojećih protokola u kontrolisanom i izvodljivom okruženju. Moguće je dizajnirati raznovrsne mrežne topologije koristeći različite tipove čvorova (host, hab, bridž, ruter, mobilna jedinica, itd.). Nakon toga, lako se može proučiti ponašanje protokola rutiranja u različitim topologijama, s obzirom na to da je mrežna topologija samo skup simulacionih parametara. Većina mrežnih simulatora je zasnovana na paradigmi diskretne simulacije zasnovane na događajima (*discreat event simulation*).

5.1. Poređenje osnovnih karakteristika mrežnih simulatora

Jedna od podela mrežnih simulatora je na komercijalne simulatore i simulatore otvorenog koda (*open source*). Neki od najpoznatijih simulatora otvorenog koda su NS-2, NS-3 i OMNeT++. Prednost ovih simulatora je u tome što omogućavaju slobodan pristup svim potencijalnim korisnicima, tako da svaki korisnik može pronaći eventualne nedostatke u njima i doprineti njihovom poboljšanju. Ovi simulatori omogućavaju implementaciju potpuno novih modela i modifikaciju postojećih, radi sprovođenja željenih simulacija. Za simulatore otvorenog koda takođe važi da su veoma fleksibilni i pogodni za usvajanje novih tehnologija. Mrežni simulatori OMNeT++ i NS-2 su realizovani kao dvojezični (*dual language*) simulatori, što znači da koriste dva programska jezika: jedan (najčešće C++) za kreiranje modela u jezgru simulatora, dok se drugi programski jezik koristi za konfiguraciju mreže. Ovakav pristup nije problematičan i ne unosi dodatnu kompleksnost ako se simulator koristi samo za pokretanje simulacija sa postojećim modelima, jer je jezik za konfiguraciju mreže uglavnom relativno jednostavan. Međutim, kada korisnici žele da implementiraju sopstvene protokole i modele, upotreba dva jezika može dovesti do nepotrebne kompleksnosti. S obzirom na to da je NS-3 jednojezični (*single language*) simulator, napisan u C++ programskom jeziku, nije iznenađujuće što ga sve više korisnika bira za svoja istraživanja.

Najpoznatiji komercijalni mrežni simulatori su OPNET i QualNet. Neke od prednosti komercijalno dostupnih simulatora su detaljna i kvalitetna dokumentacija, tehnička podrška, podrška radu sa veoma skalabilnim mrežama, podrška za industrijske standarde i sertifikate. Najčešće pokazuju bolje performanse od simulatora otvorenog koda. Najveća mana komercijalno dostupnih simulatora

je visoka cena, što mnogim istraživačima može biti presudan faktor da se opredele za simulatore otvorenog koda (koji su besplatni). Takođe, kod komercijalnih simulatora korisnicima nije omogućeno menjanje jezgra simulatora, za razliku od simulatora otvorenog koda gde korisnici mogu učestvovati u unapređivanju i razvoju narednih verzija simulatora.

5.1.1. NS-2

NS-2 je simulator diskretnih događaja razvijen pomoću C++ programskog jezika, sa korisničkim interpreterom objektno orijentisanih *Tool Command Language* (TCL) skriptova (*Object TCL*, OTCL). OTCL se koristi za programiranje scenarija simulacije, dok se C++ koristi za kreiranje protokola i modela, kao i za efikasnu obradu generisanih događaja. Kompajlirani C++ objekti postaju dostupni OTCL interpreteru, čime se omogućava upravljanje već pripremljenim C++ objektima na nivou OTCL programskog jezika.

NS-2 je prvobitno bio razvijen za *Unix* platforme, međutim danas se može naći i verzija za *Windows* operativni sistem zbog velikog interesovanja njegovih korisnika. Vizuelizacija je obezbeđena putem *Network Animator* (NAM) aplikacije, a analiza dobijenih rezultata može da se obavi pomoću različitih aplikacija koje omogućavaju grafički prikaz ispitivanih performansi mreže. NS-2 podržava širok spektar protokola, uključujući UDP, *Internet Protocol* (IP), AODV, DSR, DSDV, kao i mnoge druge. Omogućava simulaciju kako žičnih, tako i bežičnih mreža. Ovaj simulator se često koristi u akademskim istraživanjima i za razvoj novih mrežnih tehnologija.

5.1.2. NS-3

NS-2 simulator je dugo bio široko korišćen simulator za istraživanje i edukaciju na Internetu i drugim mrežnim sistemima. Međutim, uveliko se radi na zameni ovog simulatora, a NS-3 predstavlja njegovog naslednika. NS-3 simulator se na više načina razlikuje od svog prethodnika, a to uključuje:

- novo softversko jezgro: dizajnirano je radi poboljšanja skalabilnosti, modularnosti, stila kodiranja i dokumentacije; jezgro je napisano u C++ programskom jeziku opciono sa dodatnim skriptovima pisanim u *Python* programskom jeziku; uključeno je nekoliko C++ stilova dizajna kao što su pametni pokazivači, šabloni i povratni pozivi; mogućnost agregacije objekata omogućava lakšu ekstenziju paketa i modela;
- aspekt na realističnosti: Internet čvorovi su dizajnirani tako da budu verniji prikaz realnih računara;
- integracija softvera: arhitektura koja podržava inkorporaciju više softvera otvorenog koda za umrežavanje, kao što su protokol stekovi jezgra, *routing daemons* (Mohta i ostali, 2010) i analizatori *trace* paketa; time se smanjuje potreba za ponovnim pisanjem modela za simulaciju;
- podrška za virtuelizaciju: NS-3 može biti pokrenut unutar virtuelnih mašina kako bi omogućio fleksibilnije testiranje mrežnih protokola i aplikacija;
- integracija testiranja: NS-3 omogućava korisniku, koji svoje istraživanje bazira na testiranju, da eksperimentiše sa novim protokol stekovima i emituje/konzumira mrežne pakete preko stvarnih upravljačkih programa ili *Virtual Local Area Network* (VLAN) mreža;
- sistem atributa: korisnicima je potrebno sredstvo za identifikaciju i potencijalno ponovnu dodelu svih vrednosti koje se koriste za konfiguraciju parametara u simulatoru. NS-3 obezbeđuje sistem atributa koji integriše rukovanje i dokumentaciju podrazumevanih i konfigurisanih vrednosti;

- arhitektura za praćenje: NS-3 formira okvir za praćenje i prikupljanje statističkih podataka pomoću dizajna zasnovanog na povratnim pozivima, koji razdvaja izvore podataka od sakupljača podataka, omogućavajući prilagođavanje *tracing* ili statističkog izlaza bez ponovne izgradnje jezgra simulatora;
- topologija: zarad jednostavnijeg korišćenja, određeni broj objekata tipa *stock* topologija unapred je definisan u simulatoru; to znači da korsnici ne moraju samostalno kreirati složene mrežne strukture, već mogu koristiti gotove modele; na ovaj način se omogućava korisnicima da kreiraju mrežu u jednoj liniji koda, sa konfigurabilnim argumentima (kao što su broj čvorova, propusni opseg linkova, itd.); najčešće korišćene *stock* topologije uključuju topologiju stabla, meš topologiju, topologiju zvezde i slučajne topologije proizvoljne veličine; ovakvi objekti, odnosno topologije, su ugrađeni iz *Georgia Tech Network Simulator* (GTNetS) simulatora.

5.1.3. OMNeT++

OMNeT++ je ekstenzibilni, modularni C++ simulacioni okvir, prevashodno namenjen za izgradnju mrežnih simulatora. Pod mrežom se podrazumevaju žične, bežične i druge komunikacione mreže. Može se koristiti besplatno za nekomercijalne simulacije, kao što su akademska istraživanja i nastava. OMNeT++ modeli obezbeđuju podršku za senzorske mreže, WANET mreže, Internet protokole, modeliranje performansi, itd. Takođe nudi *Eclipse* integrisano razvojno okruženje (*Integrated Development Environment*, IDE), grafičko *runtime* okruženje i niz drugih alata. Postoje nadogradnje za simulaciju u realnom vremenu, emulaciju mreže, integraciju baze podataka, *SystemC* integraciju i razne druge funkcije.

Iako sam po sebi nije mrežni simulator, OMNeT++ je postao široko rasprostranjen kao mrežna simulaciona platforma u naučnom okruženju. U OMNeT++ se mogu nadograditi različita simulaciona okruženja koja implementiraju simulatore za različite sisteme, kao što su INET za žične i bežične mreže, Simu5G za mobilne LTE i 5G mreže, Veins za VANET mreže (koji ujedno integriše i kontrolu kretanja vozila kroz SUMO simulator), itd. OMNeT++ obezbeđuje modularnu arhitekturu za modele koji su programirani u C++ programskom jeziku, a zatim su grupisani u veće komponente i module korišćenjem jezika višeg nivoa (*Network Description*, NED). OMNeT++ ima široku *Graphical User Interface* (GUI) podršku, a zbog svoje modularne arhitekture smiulacioni modeli se lako mogu ugraditi u korisničke aplikacije. Da bi se upravljalo i smanjilo vreme potrebno za izvođenje velikih simulacija, razvijeni su dodatni alati, na primer, bazirani na *Python* programskom jeziku.

5.1.4. OPNET

OPNET predstavlja jedan od najpopularnijih komercijalnih mrežnih simulatora. Kao i većina ostalih komercijalnih mrežnih simulatora, ovaj simulator nudi potpunu tehničku podršku i kompletnu dokumentaciju. Zahvaljujući prilagodljivom izvornom kodu moguće je vršiti simulacije mreža različitih veličina. Ovaj simulator spada u simulatore diskretnih događaja. Raspoloživ je za *Windows* i *Unix* platforme.

Podržava razne tehnologije *Local Area Network* (LAN) i *Wide Area Network* (WAN) mreža, kao što su *Asynchronus Transfer Mode* (ATM), IP, mobilne mreže, bežični LAN, itd. Sadrži integrisanu aplikaciju za analizu rezultata i otklanjanje grešaka zasnovanu na grafičkom korisničkom interfejsu. OPNET simulator ima ugrađenu kompatibilnost za rad sa *Internet Protocol version 4* (IPv4) i *Internet Protocol version 6* (IPv6) protokolima i podršku za veliki broj protokola rutiranja (između ostalih AODV i DSR). Korisnički interfejs je realizovan pomoću C/C++ programskog jezika. Ima ugrađenu podršku za *Hardware In The Loop* (HITL) simulacije i *Distributed Simulations* (DS) mogućnosti.

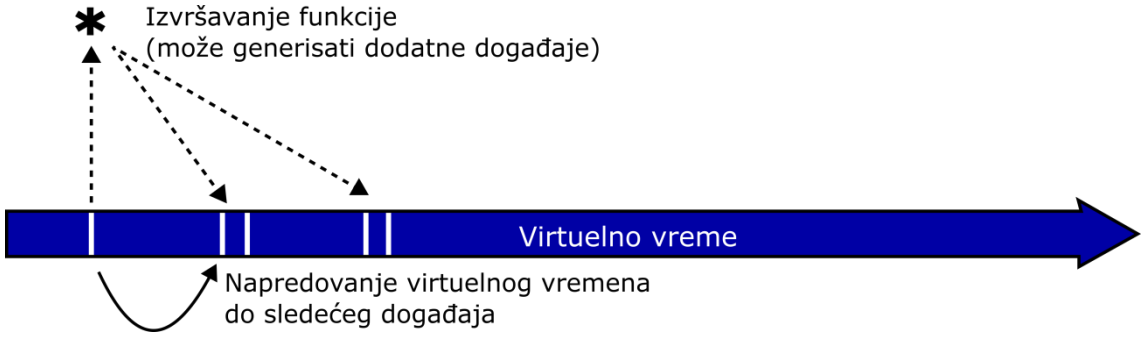
5.1.5. QualNet

Još jedan popularan komercijalni mrežni simulator je QualNet, koji se koristi za simulaciju i modeliranje različitih vrsta komunikacionih mreža, uključujući bežične, mobilne i *ad hoc* mreže. Razvijen je od strane kompanije *Scalable Network Technologies* i često se koristi u akademskim istraživanjima, industriji i za razvoj protokola i sistema u oblasti telekomunikacija i računarskih mreža. Ovaj simulator se može koristiti za simulaciju raznih bežičnih standarda, kao što su *5G, Wi-Fi, ZigBee, Bluetooth,* itd. QualNet takođe nudi podršku za veliki broj mrežnih protokola, uključujući protokole za WANET mreže kao što su AODV, DSR i OLSR. Korisnicima ovog simulatora omogućeno je i kreiranje sopstvenih protokola i modela. Takođe, ovaj simulator omogućava simulaciju u realnom vremenu, čime se obezbeđuje testiranje modela u uslovima bliskim realnom okruženju.

5.2. Osnovni koncepti NS-3 simulatora

NS-3 je simulator diskretnih događaja prvenstveno namenjen za istraživačke i obrazovne svrhe. Kao i NS-2, spada u simulatore otvorenog koda. Razvoj ovog simulatora započet je 2006. godine, a prva verzija je puštena u rad 2008. godine. NS-3 nije ekstenzija NS-2 simulatora, već je potpuno novi simulator. Sličnost između NS-2 i NS-3 simulatora je ta što su oba napisana u C++ programskom jeziku, ali NS-3 ne podržava NS-2 *Applicatons Programming Interface* (API). Kôd koji je implementiran u NS-2 simulator može se lako preneti na NS-3 simulator.

Na slici 5.1 ilustrovan je opšti princip simulacije diskretnih događaja. Vreme simulacije kreće se u diskretnim skokovima od događaja (*event*) do događaja (nije kontinualno). C++ funkcije planiraju (*schedule*) događaje koji se javljaju u određenim vremenskim trenucima simulacije. Konceptualno, simulator prati niz događaja koji su planirani da budu izvršeni u određenom simulacionom vremenu. Zadatak simulatora je da izvršava događaje po sekvencijalnom vremenskom redosledu. Kada dođe do završetka jednog događaja, simulator će preći na sledeći događaj (ili će se zaustaviti ukoliko je to poslednji događaj). Ovakav princip rada se podrazumeva kada se govori o simulatoru diskretnih događaja.

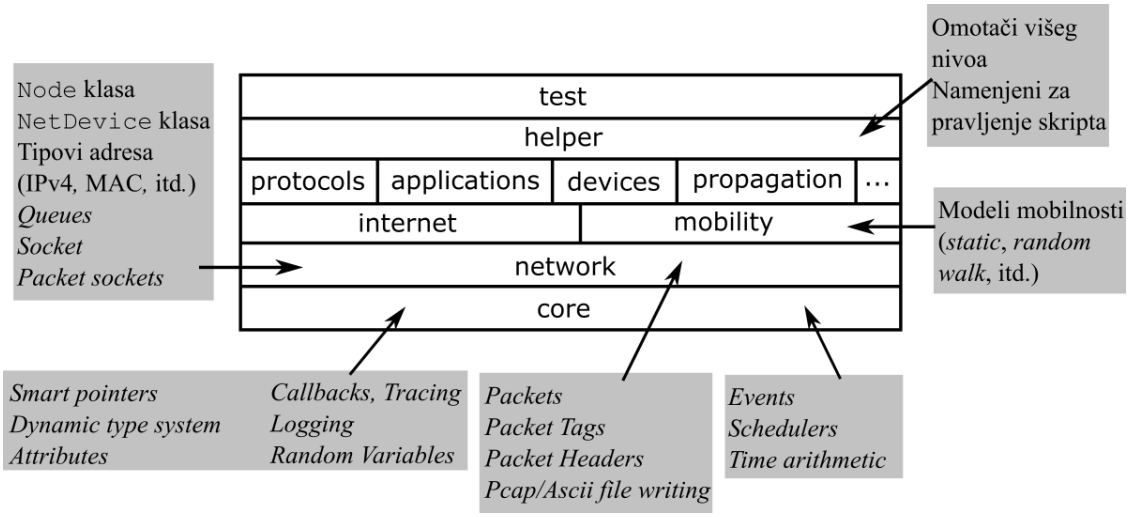


Slika 5.1. Opšti princip simulacija diskretnih događaja

5.2.1. Organizacija simulatora

NS-3 simulator realizovan je kao sistem međusobno povezanih softverskih biblioteka. Korisnički programi su kreirani tako da povezuju ove biblioteke u jedan simulacioni scenario, a napisani su u C++ ili *Python* programskom jeziku. Ciljni sistem treba da ima softversko razvojno okruženje koje omogućava efikasno kompajliranje biblioteka kao i korisničkih programa. NS-3 simulator, u budućnosti, mogao bi da bude distribuiran kao sistem unapred instaliranih softverskih biblioteka. NS-3, kao simulator otvorenog koda, obezbeđuje javno dostupan izvorni kôd i omogućava korisnicima da implementacijom sopstvenih modela doprinesu njegovom razvoju, poboljšanju i

otkrivanju eventualnih nedostataka. Sam interfejs simulatora takođe je otvoren za buduće poboljšanje. Otvorenost koda omogućava veću fleksibilnost i usvajanje većine novih tehnologija na brži način od komercijalnih mrežnih simulatora.



Slika 5.2. Organizacija softvera NS-3 simulatora

Izvorni NS-3 kôd se uglavnom nalazi u `src` direktorijumu i njegova struktura je prikazana na slici 5.2. Ovaj kôd se sastoji iz više modula organizovanih u više nivoa, a svaki modul zavisi samo od modula ispod njega. Implementacija simulacionog jezgra (`core`) nalazi se u direktorijumu `src/core`. U `core` modulu su deklarisane sve osnovne klase koje, bez specifičnih atributa, služe kao osnova za izvođenje svih drugih klasa dobijenih putem nasleđivanja. Paketi (*packets*) su osnovni objekti u mrežnom simulatoru i implementirani su u `src/network` direktorijumu. Modul `network` je zadužen za inicijalizaciju, održavanje i povezivanje klasa sa nasleđenim osobinama od osnovnih `core` klasa. Ova dva modula imaju za cilj da obuhvate generičko simulaciono jezgro koje mogu koristiti različite vrste mreža, a ne samo mreže bazirane na Internetu.

`Mobility` modul omogućava implementaciju modela za pokretljivost čvorova, npr. *Wi-Fi* stanice koje nemaju fiksnu poziciju. `Internet` modul pruža mogućnost rutiranja unutar IPv4 i IPv6 protokola, a obezbeđuje i TCP i UDP modele. Zatim slede `protocols`, `aplications`, `devices`, `propagation` i drugi moduli sa svojim bibliotekama modela kojima se dalje proširuju mogućnosti simulatora. Omotači višeg nivoa (*helper*) zaduženi su za jednostavnije pravljenje skripta i omogućuju lakše kreiranje, povezivanje i agregaciju svih objekata nižeg nivoa. NS-3 simulator takođe obezbeđuje i odgovarajuće okruženje za testiranje modela (*test*). Svi moduli poseduju poseban skup programskih rutina kojima se obezbeđuje testiranje i verifikacija ispravnosti osnovnih funkcija modela.

Svaki od modula ima poddirektorijume `model`, `helper`, `examples`, `test`, `doc` i datoteku `CMakeLists.txt`. U poddirektorijumu `model` smešteni su osnovni C++ programi koji predstavljaju realizaciju konkretnog modela. U poddirektorijumu `helper` definišu se pomoćne klase za pojednostavljenje konfiguracije i upravljanja mrežnim entitetima, omogućavajući lakšu integraciju i ponovnu upotrebu koda unutar simulacija. U poddirektorijumu `examples` nalaze se primeri korišćenja posmatranog modela. U poddirektorijumu `test` su programi za testiranje modela i validaciju koda. U poddirektorijumu `doc` smeštena je dokumentacija za taj modul. Datoteka `CMakeLists.txt` služi za konfiguraciju i upravljanje procesom kompajliranja modula koristeći `CMake` sistem. U njoj se definiše koji implementacioni (ekstenzija `.cc`) i heder (ekstenzija `.h`) fajlovi treba da se kompajliraju, zavisnost od drugih modula, opcije kompajliranja,

testovi i primeri. `CMake` sistem je zamenio `Waf` sistem koji se koristio za kompajliranje do ns-3.36 verzije simulatora. Aktuelna verzija simulatora je ns-3.43 (NS-3, 2025).

Za dalje proširenje funkcionalnosti simulatora korisnici mogu ručno proširiti simulator kreiranjem novog modula u direktorijumu `src`. Proces započinje formiranjem i implementacijom strukture direktorijuma, koja obuhvata poddirektorijume `model`, `helper`, `examples`, `test` i `doc`. U datoteci `CMakeLists.txt` novi modul se registruje pomoću funkcije `ns3_add_library()`, uz definisanje zavisnosti od drugih modula (funkcija `target_link_libraries()`) i uključivanje testova (funkcija `ns3_add_test_library()`) i primera (funkcija `ns3_add_example()`). Nakon završetka implementacije, modul se kompajlira i testira pomoću `CMake` sistema, omogućavajući integraciju u postojeći okvir simulatora.

5.2.2. NS-3 model objekta

U NS-3 simulatoru se koriste tri specijalne bazne klase:

- `Object`,
- `ObjectBase` i
- `SimpleRefCount`.

Nije neophodno da NS-3 objekti pripadaju klasama koje se nasleđuju iz ovih baznih klasa, međutim klase koje su naslednice baznih klasa imaju određene specijalne osobine. Klase koje proističu iz klase `Object`, automatski dobijaju i sledeća svojstva (koja će biti objašnjena u nastavku):

- sistem pametnih pokazivača (*smart pointers*),
- sistem agregacije objekata i
- sistem atributa.

Naslednice klase `ObjectBase` dobijaju poslednja dva svojstva, ali ne i sistem pametnih pokazivača. Klase koje su nasleđene iz klase `SimpleRefCount` dobijaju samo sistem pametnih pokazivača.

5.2.2.1. Pametni pokazivači

Sistem pametnih pokazivača je u NS-3 simulaturu realizovan pomoću klase `Ptr`, koja predstavlja šablonizovanu klasu za čuvanje pokazivača. Šablonom klase definiše se funkcija cele familije klasa koje imaju isti izgled, ali definisanu funkcionalnost obavljaju nad različitim tipovima podataka. Tip podataka zadaje se u <> zagradama. Na primer, `Ptr<Packet>` definiše pametni pokazivač za tip `Packet`. Konkretna klasa sa zadatim tipom podataka, generisana iz šablona, naziva se šablonska klasa.

Najvažnija funkcija šablonizovane klase `Ptr` je da obezbeđuje automatsko uništenje objekta kada ga više niko ne koristi, odnosno kada ne postoji više nijedan pokazivač na taj objekat. Na ovaj način se čuva memorija, tako da pametni pokazivači pomažu u upravljanju memorijom i njenom optimalnom korišćenju. Takođe, klasa `Ptr` omogućava korisnicima da manipulišu pametnim pokazivačima kao da su obični pokazivači: moguće ih je upoređivati sa nulom ili sa drugim pokazivačima, dodeljivati im nultu vrednost, itd. Konkretni pokazivač je moguće izvući iz pametnog pokazivača pomoću funkcija `GetPointer()` i `PeekPointer()`. Za skladištenje novog pokazivača na objekat unutar pametnog pokazivača, koristi se funkcija `CreateObject()`

koja ujedno služi i za kreiranje novog objekta. Ova funkcija je takođe šablonizovana, a tip kreiranog objekta mora da se poklapa sa tipom pametnog pokazivača.

5.2.2.2. Agregacija objekata

Kako bi se izbegla dva problema koja su se javljala kod NS-2 simulatora, *downcasting* i "slaba bazna klasa", u NS-3 simulator uveden je sistem agregacije objekata. *Downcasting* se odnosi na postupak eksplicitne konverzije pokazivača bazne klase na pokazivač podklase, tako da se API podklase može koristiti. *Downcasting* se može na bezbedan način izvršiti uz pomoć funkcije `GetObject()`. "Slaba bazna klasa" se odnosi na probleme koji nastaju kada baznoj klasi nedostaju neophodne funkcionalnosti, te se zbog toga ne može efikasno ponovo iskoristiti.

5.2.2.3. Atributi

NS-3 sistem atributa predstavlja osnovu za dodeljivanje vrednosti internih promenljivih objektima koji su deo simulacije, čime se obezbeđuje podešavanje parametara simulatora, konfiguracija čvorova i mreže, pristup objektima za generisanje statističkih rezultata simulacije i slično. Svaki NS-3 objekat ima određeni broj atributa, koje je najpre neophodno definisati, a zatim izvršiti njihovo podešavanje. Uzimajući u obzir značaj sistema atributa, u NS-3 simulatoru obezbeđen je veći broj mogućih načina za ispravno podešavanje njihovih vrednosti. U tabeli 5.1 prikazani su najčešće korišćeni metodi za postavljanje vrednosti atributa. U nastavku je pojedinačno opisan svaki od navedenih metoda:

Tabela 5.1. Najčešće korišćeni metodi za postavljanje vrednosti atributa u NS-3 simulatoru

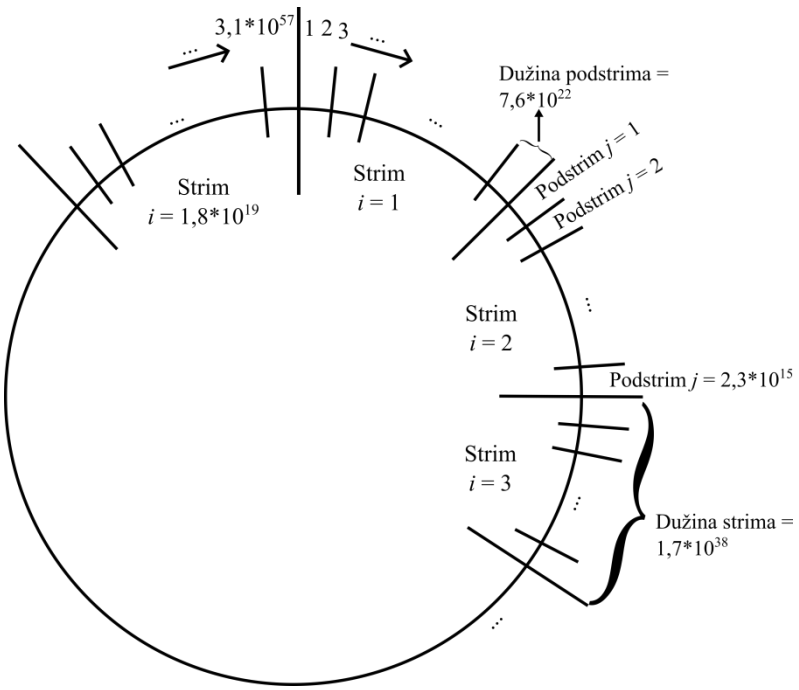
Metod	*Oblast važenja*
1) `GetTypeId()`	Svi primerci klase
2) `CommandLine`	Svi budući primerci klase
3) `Config::SetDefault()`	
4) `ConfigStore`	
5) `ObjectFactory`	Svi primerci klase kreirani uz pomoć mehanizma *object factory*
6) *Helper* metodi	Svi primerci klase kreirani korišćenjem pomoćne klase
7) `Object::SetAttribute()`	Tačno jedan primerak klase
8) `Config::Set()`	

1) Prilikom definisanja atributa u funkciji `GetTypeId()` postavljaju se ujedno i podrazumevane vrednosti atributa. S obzirom da se na ovaj način postavljaju podrazumevane vrednosti u odgovarajućem modelu, oblast važenja je nad svim kreiranim objektima ovog tipa.
2) Jedan od načina na koji je moguće podesiti vrednosti atributa u NS-3 simulatoru, bez editovanja i kompajliranja skripta, je pomoću argumenata komandne linije (*command line arguments*). Ovaj metod sadrži mehanizam za raščlanjivanje (*parse*) argumenata komandne linije i automatsko postavljanje lokalnih i globalnih promenljivih na osnovu tih argumenata. Oblast važenja kod ovog metoda za postavljanje vrednosti atributa je za sve buduće primerke klase, jer se podrazumevane vrednosti definišu unutar skripta i odnose se na objekte kreirane nakon definisanja ovih podrazumevanih vrednosti.
3) Podešavanje podrazumevanih vrednosti atributa može se vršiti i pomoću `Config` sistema, korišćenjem funkcije `Config::SetDefault()`. Slično kao u prethodnom metodu, oblast važenja je za sve objekte koji se kreiraju nakon odgovarajuće linije koda u skriptu gde se poziva funkcija `Config::SetDefault()`.

4) Vrednosti atributa u NS-3 simulatoru moguće je sačuvati u *American Standard Code for Information Interchange* (ASCII) ili *Extensible Markup Language* (XML) tekstualnoj datoteci i učitati u nekom budućem pokretanju simulacije. Ovo je omogućeno zahvaljujući specijalizovanoj bazi podataka za vrednosti atributa i podrazumevane vrednosti, koja se naziva `ConfigStore`. Zahvaljujući ovoj bazi podataka, moguće je iz odgovarajuće datoteke učitati vrednosti određenih atributa u skript. Zbog toga nije potrebno svaki put definisati željene vrednosti atributa u skriptu, već ih je sve moguće definisati u određenoj datoteci. Oblast važenja je, kao i kod prethodna dva metoda, za sve buduće primerke klase.
5) *Object factory* mehanizam omogućava korisnicima da konstruišu C++ objekte tako da ne moraju da znaju koja konkretna klasa objekata se pravi. Ovaj mehanizam se koristi za instanciranje objekata i za konfigurisanje atributa na tim objektima. U NS-3 simulatoru realizovan je pomoću klase `ObjectFactory`. Ova klasa ima funkcije članice `SetTypeId()`, `GetTypeId()`, `IsTypeIdSet()`, `Set()` i `Create()`. Oblast važenja je za sve objekte koji su kreirani uz pomoć mehanizma *object factory*.
6) Ako se za postavljanje vrednosti atributa koriste *Helper* metodi, pomoćna klasa je odgovorna za podešavanje atributa, tako da se oblast važenja odnosi na sve primerke klase koji su kreirani korišćenjem ove klase. Pomoćna klasa obezbeđuje da se u korisničkom skriptu na jednostavan način kreira, upravlja i pristupa bilo kom kreiranom objektu te klase.
7) Ukoliko je potrebno promeniti vrednost atributa nekog konkretnog objekta, najjednostavnije je koristiti funkciju `SetAttribute()`, koja je definisana u klasi `Object`. Svaka klasa koja ima sistem atributa, s obzirom da je naslednica klase `Object`, automatski poseduje i funkciju `SetAttribute()`. Za korišćenje ove funkcije neophodno je imati pokazivač na objekat čija se funkcija poziva. Ovi pokazivači su iz korisničkog skripta najčešće teško dostupni ili potpuno nedostupni, osim u slučaju kada je objekat kreiran pozivom funkcije `CreateObject()`. S obzirom da se funkcija `SetAttribute()` poziva nad jednim konkretnim objektom, oblast važenja je taj jedan objekat.
8) Ukoliko nije poznat pokazivač na objekat čiju vrednost atributa treba promeniti, najbolje je koristiti funkciju `Set()`, koja pripada `Config` sistemu. U NS-3 simulatoru postoji lista svih kreiranih čvorova (`NodeList`), a u okviru svakog čvora postoji lista svih mrežnih uređaja (`DeviceList`) koje taj čvor koristi. Korišćenjem ove dve liste mogu se podešavati atributi mrežnih uređaja željenog čvora. Slično tome, svaki čvor poseduje i listu aplikacija (`ApplicationList`) koja omogućava podešavanje atributa aplikacija instaliranih u odgovarajućem čvoru. Oblast važenja kod ovog metoda je za jedan konkretan objekat. Za podešavanje nekog atributa u svim čvorovima i svim aplikacijama koje poseduju ovaj atribut, umesto indeksa čvora u listi čvorova i indeksa aplikacije u listi aplikacija koristi se *. Oblast važenja u ovom slučaju proširuje se na sve objekte koji su obuhvaćeni odgovarajućom komandom za podešavanje atributa.

5.2.3. Slučajne promenljive

NS-3 simulator sadrži ugrađeni generator pseudo slučajnih brojeva (*Pseudo-Random Number Generator*, PRNG). Ovaj generator je identičan kao i kod NS-2 simulatora – MRG32k3a (L'Ecuyer i ostali, 2001). PRNG sekvenca je podeljena na niz intervala koji se ne preklapaju i oni se nazivaju strimovi (*streams*). Generator obezbeđuje približno $1{,}8 \cdot 10^{19}$ nezavisnih strimova slučajnih brojeva, od kojih se svaki sastoji od $2{,}3 \cdot 10^{15}$ podstrimova (slika 5.3). Svaki podstrim ima dužinu (tj. broj slučajnih brojeva do sledećeg podstrima) $7{,}6 \cdot 10^{22}$. Podstrimovi u NS-3 simulatoru su indeksirani globalnom promenljivom koja se naziva `RngRun` i koja može imati vrednosti od 1 do $2{,}3 \cdot 10^{15}$. Dužina celog opsega slučajnih brojeva koje može da generiše generator je $3{,}1 \cdot 10^{57}$.

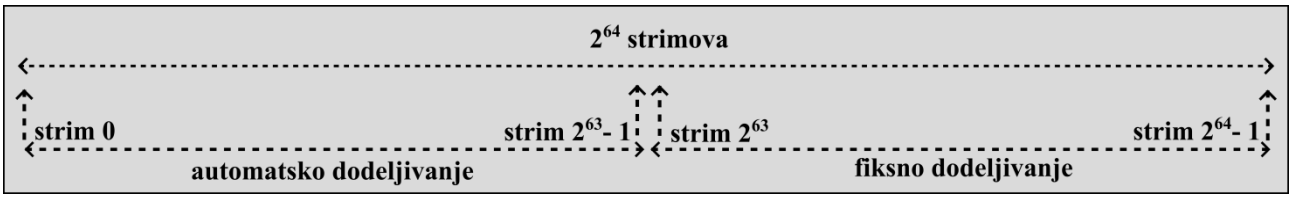


Slika 5.3. Podela opsega slučajnih brojeva u NS-3 simulatoru na strimove i podstrimove

Svaki strim se dodeljuje po jednoj slučajnoj promenljivoj i ovi strimovi su nekorelisani. Takođe, svi podstrimovi unutar jednog strima su nekorelisani. Podrazumevani opsezi iz kojih slučajne promenljive uzimaju vrednosti su prvi podstrimovi unutar svakog pojedinačnog strima. Za jednu simulaciju dovoljan je po jedan podstrim za svaku slučajnu promenljivu. Promena podstrima iz kog slučajna promenljiva uzima vrednost vrši se promenom parametra `RngRun` (podrazumevana vrednost je 1). Promena celokupne PRNG sekvence slučajnih brojeva vrši se promenom parametra `RngSeed`.

Ukoliko se menja `RngSeed`, nije garantovano da će strimovi pojedinih slučajnih promenljivih biti nekorelisani za različita pokretanja simulacije. S druge strane, ukoliko se fiksira `RngSeed`, a menja se `RngRun`, strimovi će biti nekorelisani. Zbog toga se kod višestrukih simulacija u NS-3 simulatoru preporučuje menjanje parametra `RngRun` uz fiksno `RngSeed`.

Celokupan prostor slučajnih brojeva, koji sadrži 2^{64} (približno $1{,}8 \cdot 10^{19}$) strimova, podeljen je na dva dela (slika 5.4). Prvih 2^{63} strimova se smešta u bazu za automatsko dodeljivanje, a drugih 2^{63} strimova je rezervisano za fiksno dodeljivanje (od strane korisnika).



Slika 5.4. Podela celokupnog prostora slučajnih brojeva u NS-3 simulatoru

Klase koje realizuju određenu raspodelu slučajnih promenljivih se nasleđuju iz bazne kalse `RandomVariableStream`. Ova klasa je naslednica NS-3 klase `Object`, tako da samim tim ima ugrađen sistem atributa, agregacije objekata i pametnih pokazivača. Neke od klasa koje realizuju određenu raspodelu slučajnih promenljivih su: `UniformRandomVariable`, `Constant RandomVariable`, `SequentialRandomVariable`, `ExponentialRandom`

`Variable`, `NormalRandomVariable`, itd. Sve ove klase su dokumentovane u skriptu `random-variable-stream.h`, koji je smešten u modulu `src/core/model`. Korisnik može kreirati i druge slučajne promenljive pod uslovom da su izvedene iz bazne klase `Random VariableStream`. Određena slučajna promenljiva, kao i bilo koja promenljiva klase `Object`, može da se kreira na više načina, korišćenjem mehanizma *object factory*, pozivom funkcije `CreateObject()` i slično.

Svaka od izvedenih klasa `RandomVariableStream` klase ima odgovarajući skup atributa. Na primer, klasa `UniformRandomVariable` ima dva atributa: `Min` i `Max`. Podrazumevane vrednosti ovih atributa su 0 i 1, respektivno, tako da slučajna promenljiva uzima vrednosti iz opsega [0,1]. Postavljanje željenih vrednosti ovih atributa može se vršiti na više načina, kao što je prethodno prikazano u tabeli 5.1. Bazna klasa `RandomVariableStream` obezbeđuje dve funkcije (koje nasleđuju i sve izvedene klase): `GetInteger()`, koja vraća celobrojnu vrednost, i `GetValue()`, koja vraća realni broj. Osim ovih funkcija, svaka slučajna promenljiva može definisati i druge specijalizovane funkcije za generisanje slučajnih brojeva.

5.2.4. *Callback* sistem

Ukoliko u toku simulacije postoji potreba za prenosom informacija između dva simulaciona modula, to je moguće izvršiti pomoću *callback* mehanizma. Ovaj mehanizam omogućava da se u nekom delu koda pozove funkcija iz drugog modula, bez ikakve intermodulske zavisnosti. Za to je potrebna jedna vrsta indirektnosti koja podrazumeva korišćenje pokazivača na tu funkciju. Adresa funkcije koja se poziva tretira se kao promenljiva. Ova promenljiva se naziva pokazivač na funkciju (*pointer-to-function variable*). Veza između funkcije i pokazivača na funkciju je identična kao veza između objekta i pokazivača na objekat. U modulu koji poziva funkciju nekog drugog modula definiše se samo pokazivač na funkciju određenog tipa, a koja će se tačno funkcija pozvati definiše se prosleđivanjem odgovarajuće adrese funkcije kada je to potrebno. Na taj način ne postoji direktna veza između modula, već se veza uspostavlja indirektno preko pokazivača.

Callback API obezbeđuje:

- deklarisanje tipa *callback* funkcije;
- prosleđivanje bilo kog poziva nekoj C++ funkciji članici klase ili funkciji koja nije članica klase pomoću šablonizovanog *callback* mehanizma.

Jedna od varijacija *callback* funkcija su *null callback* funkcije. Pozivanje *null callback* funkcije je identično pozivanju pokazivača na nulu i dovodi do neregularnog prekida izvršavanja programa. Zato se uvek preporučuje provera da li je neki pokazivač jednak nuli pre pokušaja da se pozove funkcija na koju pokazuje. Još jedna vrsta *callback* funkcija su *bound callback* funkcije, koje primaju i argumente za fiksno dodeljivanje. Vrlo važan deo *callback* sistema je *traced callback*, koji će biti objašnjen u odeljku 5.2.5.2.

Callback sistem se primenjuje kod:

- *Socket* API,
- *Layer*-2/*Layer*-3 API,
- *Tracing* podsistema i
- API između IP i podsistema za rutiranje.

5.2.5. Generisanje izlaznih podataka iz simulatora

Cilj svakog pokretanja simulacije je dobijanje odgovarajućeg izlaza koji će korisnici kasnije moći da koriste za dalja istraživanja. NS-3 simulator ovo obezbeđuje uz pomoć dva mehanizma: sistem za logovanje i sistem za praćenje. Sistem za logovanje obezbeđuje jednostavnu kontrolu informacija koje će biti štampane na ekran (informacije o greškama, upozorenja, itd.), dok sistem za praćenje ima pristup jezgru simulatora i omogućava upis u različite vrste datoteka ili jednostavno štampanje na ekran podataka o promenama određenih parametara ili o dešavanju događaja koji su interesantni za korisnika. Drugim rečima, sistem za praćenje omogućava izvlačenje podataka iz simulacionog modela koji su poželjni da budu izlaz simulacije. Kod sistema za logovanje svi podaci se izbacuju na ekran u tekstualnoj formi i taj tekst kontroliše projektant modula. Kod sistema za praćenje projektant modula generiše odgovarajuću informaciju korisniku, dok obradu i format zapisa te informacije kontroliše sam korisnik.

5.2.5.1. Sistem za logovanje

NS-3 sistem za logovanje projektovan je pre svega za nadgledanje i debagovanje rada simulacionih programa. Pomoću ovog sistema se određuje koje će se informacije štampati na ekran korisnicima. Tako se može izbeći štampanje nepotrebnih informacija, tako da se štampaju samo one informacije koje su potrebne korisniku. U ovom sistemu postoji osam nivoa informacija koje će se štampati i svaki od ovih nivoa (osim prvog i poslednjeg) ima odgovarajući makro kojim se vrši štampanje na ekran. Nivoi, od najnižeg do najvišeg, su:

- `LOG_NONE` – podrazumevani nivo, nema štampanja, ne postoji odgovarajući makro;
- `LOG_ERROR` – štampaju se samo poruke o greškama u toku rada, odgovarajući makro je `NS_LOG_ERROR();`
- `LOG_WARN` – štampaju se poruke sa upozorenjima, odgovarajući makro je `NS_LOG_ WARN();`
- `LOG_DEBUG` – štampaju se relativno retke poruke za debagovanje, odgovarajući makro je `NS_LOG_ DEBUG();`
- `LOG_INFO` – štampaju se informacije o radu programa, odgovarajući makro je `NS_LOG_ INFO();`
- `LOG_FUNCTION` – štampaju se informacije koje opisuju svaku funkciju koja se poziva, odgovarajući makro je `NS_LOG_ FUNCTION();`
- `LOG_LOGIC` – štampaju se informacije o svakoj logičkoj celini jedne funkcije, odgovarajući makro je `NS_LOG_ LOGIC();`
- `LOG_ALL` – štampa se sve gore navedeno, ne postoji odgovarajući makro.

Umesto prethodno navedenih pojedinačnih nivoa, moguće je koristiti notaciju kojom se obuhvata više nivoa. Na primer može se definisati `LOG_LEVEL_LOGIC`, `LOG_LEVEL_INFO`, `LOG_LEVEL_DEBUG`, itd., što znači da se štampaju poruke navedenog nivoa i svih nivoa ispod njega. Sistem logovanja sadrži i makro `NS_LOG_UNCOND()`, kojim se vrši bezuslovno štampanje poruke na ekran. Pošto je sistem za logovanje projektovan pre svega za debagovanje, on može da radi samo ako je kompajliranje NS-3 simulatora izvršeno u modu `debug`. Ako je korišćen mod `optimized`, onda sistem za logovanje uopšte ne može da se koristi, ali je izvršavanje programa značajno brže.

5.2.5.2. Sistem za praćenje

Cilj svake simulacije je generisanje određenih izlaznih rezultata potrebnih za dalja istraživanja, a NS-3 sistem za praćenje je primarni mehanizam za to. Ovaj sistem se sastoji od tri podsistema:

1) ASCII sistem za praćenje – služi za generisanje standardnih NS-2 kompatibilnih *trace* datoteka. Ove datoteke imaju ekstenziju `.tr`. To su obične tekstualne datoteke koje su obezbeđene, pre svega, zbog kompatibilnosti sa NS-2 simulatorom. Svaka linija u ovoj datoteci odgovara jednom *trace* događaju. Instaliranje ASCII sistema za praćenje u sve čvorove mreže vrši se pozivom funkcije `EnableAsciiAll()`. U pojedinim slučajevima nije praktično instalirati ASCII sistem za praćenje u sve čvorove, već je potrebno da se instalira samo u neke čvorove i mrežne uređaje. Ovo se može izvršiti pomoću funkcije `EnableAscii()`, koja može da prima različite argumente za zadavanje jednog ili više čvorova i mrežnih uređaja u koje će se instalirati ASCII sistem za praćenje. Mnogi korisnici se oslanjaju na ovaj sistem da generišu izlazne podatke koje kasnije moraju dalje da obrađuju i analiziraju pomoću spoljašnjih alata (alata koji nisu sastavni deo NS-3 simulatora).
2) *Packet capture* (PCAP) sistem za praćenje – služi za snimanje svih paketa koji se emituju u mreži. Paketi se smeštaju u *trace* datoteke u PCAP formatu sa ekstenzijom `.pcap`. Taj format može vrlo efikasno da se analizira u različitim programskim paketima. Među njima je svakako najpopularniji *WireShark*. Ovaj programski paket je besplatan, ali izuzetno bogat funkcijama, te se često koristi od strane administratora mreže jer pruža velike mogućnosti za statističku analizu paketskog saobraćaja, bilo da su podaci snimljeni u realnoj mreži ili da su nastali kao rezultat simulacije. Akronim PCAP označava "hvatanje" paketa i zapravo je API koji uključuje definiciju odgovarajućeg formata datoteke. Instaliranje PCAP sistema za praćenje u sve čvorove vrši se pomoću funkcije `EnablePcapAll()`. Za razliku od ASCII sistema za praćenje koji je mogao ceo da se smesti u jednu datoteku, kod PCAP sistema za praćenje to nije moguće. U ovom slučaju je obavezno zadavanje prefiksa za ime datoteke. Ime datoteke će imati oblik: `Prefiks-BrojČvora-BrojMrežnogUređaja.pcap`. Ovde takođe nije uvek optimalno instalirati PCAP sistem za praćenje u sve čvorove, već se može koristiti funkcija `EnablePcap()` za instaliranje ovog sistema u pojedine čvorove.
3) Opšti sistem za praćenje – korišćenje izvora podataka (*trace source*) koji generiše karakteristične podatke za praćenje i sakupljača podataka (*trace sink*) koji vrši obradu i prikaz prikupljenih podataka. Izvore podataka tipično definiše projektant modula, dok sakupljače podataka definiše korisnik. Zbog toga, za razliku od prethodna dva sistema koji generišu izlaz po unapred definisanom formatu, ovaj sistem omogućava veliku fleksibilnost korisniku koji u sakupljaču može da obrađuje podatke i da obrađene podatke snima u datoteku po proizvoljnom formatu. U NS-3 simulatoru postoje dve vrste izvora podataka koje su realizovane kroz šablonizovane klase `TracedValue` i `TracedCallback`. Klasa `TracedValue` obezbeđuje samo praćenje brojnih vrednosti (`int`, `double` i `bool`). Na primer, ako bi se posmatrao neki red za čekanje, korišćenjem ovog tipa izvora moguće je pratiti samo trenutni broj paketa u redu za čekanje. Klasa `TracedCallback` obezbeđuje praćenje kompleksnijih vrednosti ili događaja. Na primer, posmatrajući pomenuti red za čekanje, moguće je pratiti koji je tačno paket stigao, koji napustio red za čekanje i u kom trenutku, da li je došlo do prekoračenja maksimalnog broja paketa, itd. Uloga sakupljača podataka je da "sirove" podatke koje emituje izvor podataka obradi i upiše u datoteku ili ispiše na ekran u željenom formatu. Kod klase `TracedValue` tip funkcije za sakupljača podataka je striktno definisan, dok kod `TracedCallback` klase tip sakupljača podataka definiše projektant modula na proizvoljan način. Povezivanje izvora podataka sa sakupljačem podataka vrši se najčešće od strane korisnika u odgovarajućem skriptu, gde su

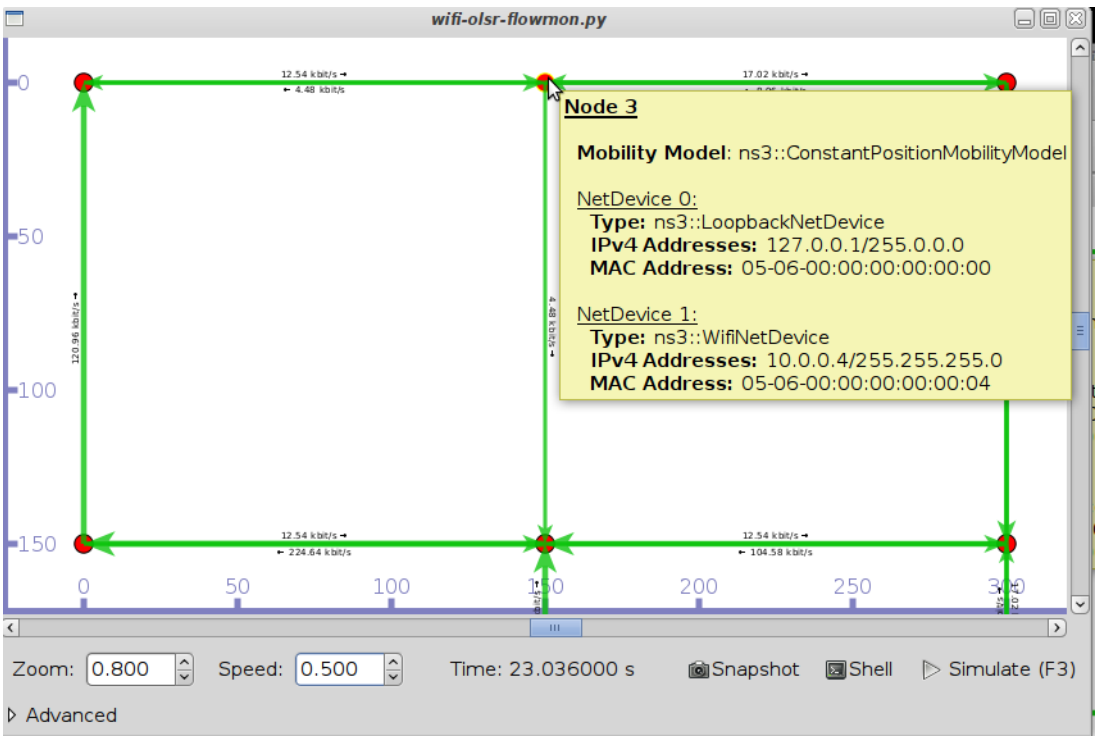
definisane i funkcije sakupljači podataka, korišćenjem funkcije `TraceConnectWithoutContext()` ili funkcije `TraceConnect()`. Potrebno je napomenuti da se na jedan izvor podataka ne moraju povezati sakupljači, što omogućava da projektant modela realizuje veći broj izvora koje smatra potencijalno korisnim, dok će se korisnici povezati samo na one izvore koji su im zaista potrebni u datom simulacionom scenariju. Takođe, na jedan izvor može se povezati i više različitih sakupljača podataka. Nakon povezivanja izvora i sakupljača podataka obezbeđeno je da se upisom novog podatka u izvor automatski pozivaju svi povezani sakupljači.

5.2.5.3. Flow monitor

Flow monitor predstavlja poseban alat za merenje performansi mrežnih protokola i analizu rezultata u NS-3 simulatoru. Ovaj alat koristi sonde (*probes*) instalirane u mrežnim čvorovima kako bi se pratili paketi koje oni razmenjuju i merili različiti mrežni parametri kao što su protok, gubici i kašnjenje paketa. *Flow monitor* se instalira na mrežnom sloju i tu prati saobraćaj IP paketa. Ukupan saobraćaj u mreži deli se na tokove (*flows*), koji se formiraju između čvorova koji generišu i čvorova koji primaju saobraćaj. Statistike koje se beleže ovim alatom odnose se upravo na ove tokove saobraćaja. Upotreba ovog alata je vrlo jednostavna, sve se obavlja korišćenjem pomoćnih klasa. Pre pokretanja simulacije potrebno je kreirati objekat tipa `FlowMonitorHelper`, a zatim pozivom funkcije `InstallAll()` instaliraju se sonde u svaki čvor mreže. Nakon završetka simulacije potrebno je rezultate upisati u fajl pozivom funkcije `SerializeToXmlFile()` objekta tipa `Ptr<FlowMonitor>`. Neke od mana ovog alata su to što loše funkcioniše sa nekim protokolima rutiranja, ne može pouzdano da prati *overflow* (jer nema ugrađenu podršku za detekciju slučajeva kada dođe do prepunjavanja bafera u mrežnim čvorovima), ne uračunava overhed, kašnjenje, i druge parametre koji se unose na slojevima iznad mrežnog sloja.

5.2.5.4. Vizuelizacija rezultata simulacije

Za vizuelni prikaz rezultata simulacije u NS-3 simulatoru koriste se alati *PyViz* i *NetAnim*.



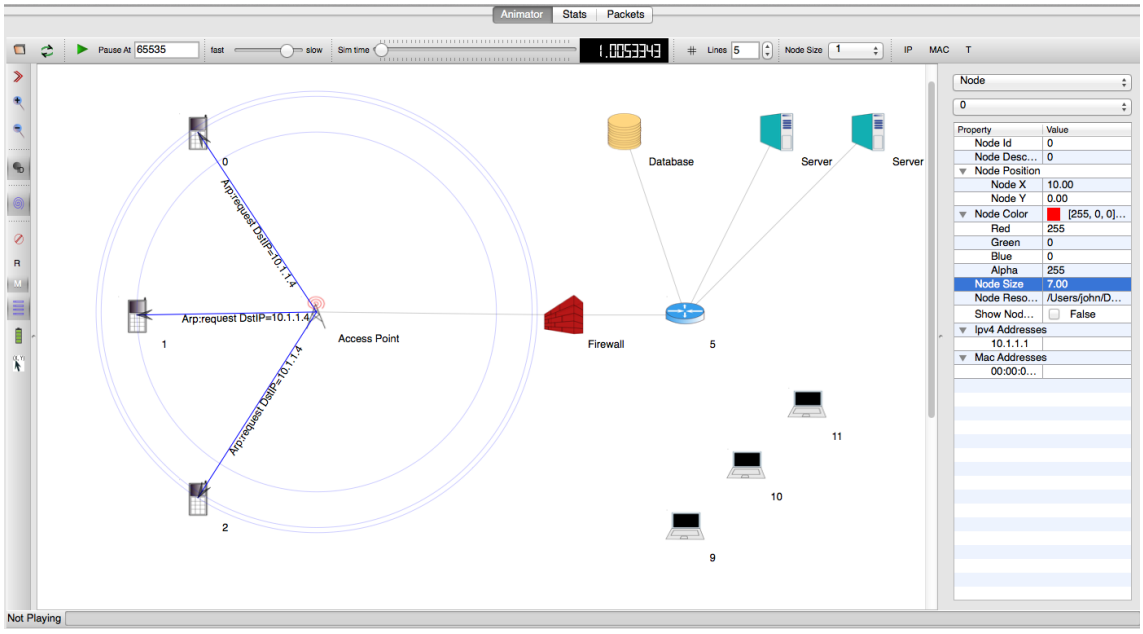
Slika 5.5. Vizuelizacija rada NS-3 simulatora pomoću alata *PyViz*

PyViz je vrlo jednostavan vizuelizator kojim se vrši vizuelizacija u toku izvršavanja simulacije, što znači da ne koristi *trace* datoteke. On se može koristiti za utvrđivanje da li su modeli mobilnosti onakvi kakve korisnik očekuje, gde dolazi do odbacivanja paketa, itd. Postoji takođe i ugrađena interaktivna *Python* konzola koja se može koristiti za debagovanje stanja određenih objekata. Na slici 5.5 predstavljen je vizuelni prikaz simulacije korišćenjem *PyViz* alata.

NetAnim je značajno kompleksniji *offline* alat za vizuelizaciju koji zahteva odgovarajuću *trace* datoteku za animaciju. Ovu *trace* datoteku kreira `AnimationInterface` u NS-3 simulatoru. Korišćenje *NetAnim* alata sastoji se iz dva koraka:

1) Generisanje XML *trace* datoteke za animaciju tokom simulacije korišćenjem `AnimationInterface` u jezgru NS-3 simulatora.
2) Učitavanje XML *trace* datoteke generisane u prethodnom koraku pomoću *offline* animatora (*NetAnim*).

NetAnim se pokreće komandom `./NetAnim` nakon čega se pojavljuje vizuelno okruženje kao na slici 5.6.



Slika 5.6. Vizuelizacija rada NS-3 simulatora pomoću alata *NetAnim*

Klikom na ikonicu u gornjem levom uglu pojavljuje se lista datoteka, od kojih treba izabrati odgovarajuću XML *trace* datoteku u koju je vršeno snimanje događaja tokom trajanja simulacije. Nakon toga se prikazuje vizuelizacija toka izvršavanja simulacije, gde je moguće pratiti šta se dešava u mreži paket po paket. *NetAnim* omogućava detaljno podešavanje vizuelnih karakteristika čvorova i prikaz saobraćaja u mreži. Ukoliko se u simulaciji koristi model za potrošnju energije čvora, te informacije se takođe mogu prikazati u ovom vizuelizatoru. Osim kao animator, *NetAnim* se može koristiti i za očitavanje XML datoteka koje je generisao *Flow monitor*. Na taj način se mogu analizirati mrežne statistike.

5.3. Elementi NS-3 simulatora za modelovanje *ad hoc* mreža

Elementi koji su potrebni za modelovanje određene *ad hoc* mreže u NS-3 simulatoru su mrežni uređaji, kanali, odgovarajući modeli mobilnosti, protokoli rutiranja i aplikacije za generisanje saobraćaja. U nastavku će svaki od ovih elemenata biti detaljnije opisan.

5.3.1. Wi-Fi mrežni uređaj i kanal

NS-3 čvorovi mogu sadržati jedan ili više mrežnih uređaja, slično kao što računari mogu sadržati više različitih mrežnih kartica. U ovom poglavlju detaljnije je objašnjen Wi-Fi model (klasa `WifiNetDevice` i pridružene klase) koji se koristi u NS-3 simulatoru za bežično povezivanje čvorova na bazi IEEE 802.11 familije standarda. Izvorni kôd Wi-Fi modela nalazi se u direktorijumu `src/wifi`. Pojedini elementi Wi-Fi modela nalaze se u posebnim modulima, odnosno odgovarajućim direktorijumima kao što je IEEE 802.11s (direktorijum `src/mesh`). Wi-Fi model u NS-3 simulatoru obezbeđuje:

- osnovni 802.11 *Distributed Coordination Function* (DCF) sa infrastrukturnim i *ad hoc* modovima;
- fizičke slojeve 802.11a, 802.11b, 802.11g, 802.11n (2,4 GHz, 5 GHz i 6 GHz opsezi), 802.11ac, 802.11ax (2,4 GHz i 5 GHz opsezi) i 802.11be modela;
- *MAC level Service Data Units* (MSDU) agregaciju i ekstenzije *MAC level Protocol Data Units* (MPDU) agregacije za 802.11n, obe se mogu kombinovati zajedno (*two-level* agregacija);
- 802.11ax *Downlink Orthogonal Frequency Division Multiple Access* (DL OFDMA) i *Uplink* OFDMA (UL OFDMA), uključujući podršku za *Multi-User Enhanced Distributed Channel Access* (MU EDCA) parametarski skup;
- 802.11be pronalaženje i podešavanje višestrukih veza;
- unapređeni 802.11e EDCA sa podrškom kvalitetu servisa (*Quality of Service*, QoS);
- mogućnost korišćenja različitih modela propagacionog slabljenja (*propagation loss*) i propagacionog kašnjenja (*propagation delay*);
- modele grešaka pri prenosu paketa i modele detekcije okvira koji su validirani kroz simulacije veza i druge reference;
- različite algoritme za kontrolu brzine, uključujući *Auto Rate Fallback* (ARF), *Adaptive Auto Rate Fallback* (AARF), *Collision-Aware Rate Adaptation* (CARA), ONOE, *Robust Rate Adaptation Algorithm* (RRAA), *ConstantRate*, *Minstrel* i *Minstrel High Throughput* (*Minstrel*-HT);
- 802.11s (*mesh*);
- 802.11p i *Wireless Access in Vehicular Environments* (WAVE).

IEEE 802.11 standard je dosta obiman i nisu svi aspekti implementirani u NS-3 simulator. Modeli fizičkog sloja funkcionišu na bazi paketa, bez frekvencijski selektivnih propagacija ili efekata interferencije, kada se koristi podrazumevani `YansWifiPhy` model. U slučaju aditivnog belog Gausovog šuma (*Additive White Gaussian Noise*, AWGN) ili širokopojasne interferencije, performanse se regulišu primenom analitičkih modela (zasnovanih na modulaciji i faktorima kao što je širina kanala).

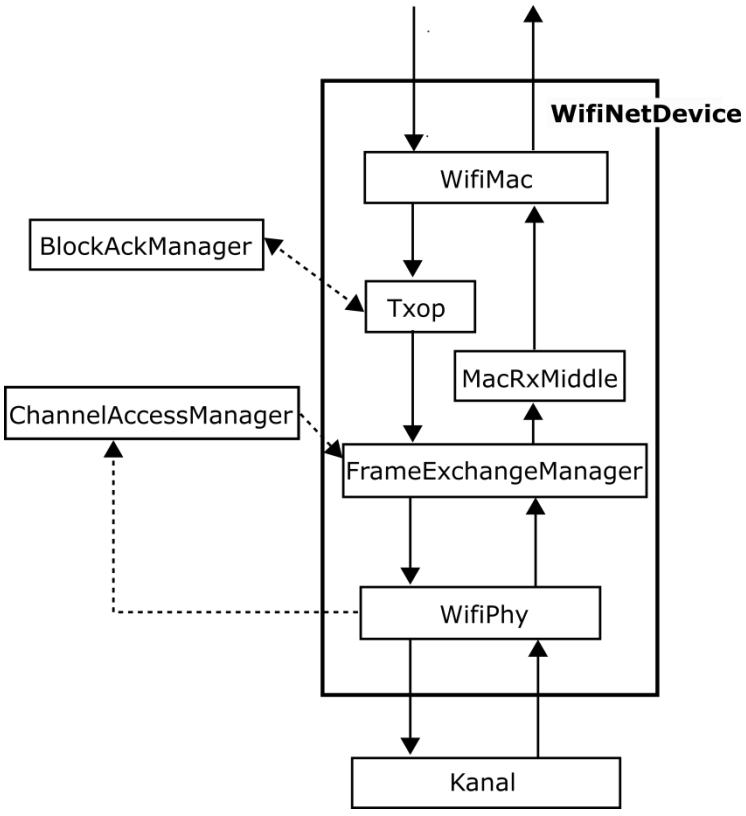
Implementacija Wi-Fi modela je modularna i sastoji se iz tri podsloja:

- model višeg MAC podsloja (*MAC high model*),
- model nižeg MAC podsloja (*MAC low model*) i
- model fizičkog sloja (*PHY layer model*).

U NS-3 simulatoru niži nivo MAC podsloja čine sledeći blokovi:

- menadžer razmene okvira (*FrameExchangeManager*),
- menadžer pristupa kanalu (*ChannelAccessManager*),
- *Transmission Opportunity* (*Txop*) objekat,
- menadžer blokova potvrda (*BlockAckManager*) i
- srednji MAC entitet (*MAC middle*).

Arhitektura Wi-Fi modela prikazana je na slici 5.7. Viši modeli MAC podsloja (*WifiMac*) vode računa o funkcionalnostima kao što su asocijacija, slanje *beacon* poruka, itd. Ovde se dodaje i MAC zaglavlje. *BlockAckManager* je odgovoran za upravljanje blokovima potvrda (*acknowledgments*), omogućavajući efikasnije upravljanje višestrukim paketima u okviru iste transmisije. *ChannelAccessManager* na bazi informacija iz fizičkog sloja (*WifiPhy*) određuje kada je zagarantovan pristup kanalu i obaveštava *FrameExchangeManager,* koji upravlja načinom na koji okviri pristupaju bežičnom kanalu i odlučuje kada i kako uređaj može da prenosi podatke. *MacRxMiddle* upravlja fragmentacijom paketa, dok *Txop* upravlja redosledom slanja paketa i čuva DCF/EDCA pristupne parametre. *WifiPhy* sloj vodi računa o slanju i prijemu okvira. Takođe, određuje verovatnoću uspešnog prijema okvira na osnovu odnosa signal šum, uključujući i interferenciju (*Signal to Interference plus Noise Ratio*, SINR). Kanal vodi računa o tome da signal bude prosleđen do svih konektovanih uređaja na fizičkom sloju, konsultovanjem modela propagacionog slabljenja i kašnjenja.



Slika 5.7. Arhitektura NS-3 Wi-Fi modela.

5.3.1.1. Model višeg MAC podsloja

U NS-3 simulatoru trenutno postoje tri tipa (izuzimajući *mesh* mreže) višeg MAC podsloja:

- model MAC podsloja za pristupne tačke (*Access Point*, AP), `ApWifiMac`,

- model MAC podsloja za radne stanice (*Station,* STA), `StaWifiMac`,
- model MAC podsloja za čvorove *ad hoc* mreže, `AdhocWifiMac`.

`ApWifiMac` implementira AP koja generiše periodične *beacon* signale i koja prihvata svaki pokušaj asocijacije. `StaWifiMac` implementira aktivan uređaj za sondiranje i asocijaciju koji upravlja automatskom reasocijacijom kad god se previše *beacon* signala neuspešno prenese. Najjednostavniji od ova tri modela je `AdhocWifiMac` koji podrazumeva implementaciju Wi-Fi MAC, koji ne vrši generisanje *beacon* signala, sondiranje ili asocijaciju.

Ova tri modela višeg MAC podsloja imaju zajedničku roditeljsku klasu `WifiMac`, koja obezbeđuje mnoge atribute za konfiguraciju modela. Na primer, atribut `QosSupported` omogućava podešavanje 802.11e QoS modela, atribut `HtSupported` omogućava podešavanje 802.11n *High Throughput* (HT) modela, atribut `VhtSupported` omogućava podešavanje 802.11ac *Very High Throughput* (VHT) modela, dok atribut `HeSupported` omogućava podešavanje 802.11ax *High Efficiency* (HE) modela. Podešavanje tipa višeg MAC podsloja uobičajeno se vrši korišćenjem pomoćne klase `WifiMacHelper`. Ova pomoćna klasa osim podešavanja tipa omogućava i podešavanje osnovnih parametara višeg MAC podsloja.

5.3.1.2. Model nižeg MAC podsloja

Model nižeg MAC podsloja podeljen je u tri komponente:

1) `FrameExchangeManager` hijerarhija klasa – implementira sekvence razmene okvira. Takođe upravlja agregacijom okvira, retransmisijom okvira, zaštitom i potvrdom prijema (*acknowledgment*).
2) `ChannelAccessManager` klasa – implementira DCF i EDCAF funkcije, koje upravljaju pristupom bežičnom kanalu u skladu sa IEEE 802.11 standardima.
3) `Txop` i `QosTxop` klase – upravljaju redom za slanje paketa. `Txop` klasa se koristi za MAC podslojeve koji ne podržavaju QoS, kao i za prenos okvira koji prema standardu koriste DCF funkcije za pristup kanalu. Klasa `QosTxop` se koristi za MAC podslojeve sa obezbeđenom QoS podrškom, omogućavajući bolju kontrolu nad saobraćajem i davanje prioriteta paketima kroz EDCA.

Često je za podešavanje funkcionalnosti Wi-Fi modela značajno napomenuti da niži MAC podsloj koristi algoritme kontrole brzine prenosa (*Rate Control Algorithms,* RCA). U NS-3 simulatoru implementiran je veći broj algoritama za kontrolu brzine prenosa, od kojih su neki preuzeti iz literature, dok se neki koriste u praksi u realnim Wi-Fi uređajima. Algoritmi koji se koriste u realnim uređajima su: `ArfWifiManager` (podrazumevani za pomoćnu klasu `WifiHelper`), `OnoeWifiManager`, `ConstantRateWifiManager`, `MinstrelWifiManager` i `MinstrelHtWifiManager`. Algoritmi koji su preuzeti iz literature i implementirani u NS-3 simulatoru su: `IdealWifiManager`, `AarfWifiManager`, `AmrrWifiManager`, `CaraWifiManager`, `RraaWifiManager`, `AarfcdWifiManager`, `ParfWifiManager`, `AparfWifiManager` i `ThompsonSamplingWifiManager`.

Ukoliko se za konfiguraciju funkcija srednjeg i nižeg sloja Wi-Fi modela koristi pomoćna klasa `WifiHelper`, tada se podrazumevano za kontrolu brzine prenosa koristi `MinstrelWifiManager`. Međutim, u praksi i u simulacijama se često koristi i algoritam sa konstantnom brzinom prenosa (`ConstantRateWifiManager`). Pri konfiguraciji Wi-Fi modela mogu se nezavisno podesiti željena brzina prenosa za sve *unicast* pakete podataka (funkcija `DataMode()`) i

željena brzina prenosa za *Request to Send* (RTS) kontrolne pakete (funkcija `ControlMode()`). Brzina prenosa preostalih kontrolnih paketa (*Acknowledgment* (ACK), *Clear to Send* (CTS) i dr.) propisana je standardom te se ne može podešavati u procesu konfiguracije. Osim toga, ako je potrebno slati *non-unicast* pakete, teba imati u vidu da se oni podrazumevano šalju nižom brzinom prenosa od *unicast* paketa.

5.3.1.3. Model fizičkog sloja

Fizički model je pre svega odgovoran za modelovanje prijema paketa i potrošnje energije čvora. Tipično se mogu izdvojiti tri važne komponente u procesu prijema paketa:

1) Svaki paket se podvrgava statističkim proračunima kako bi se odredilo da li je prijem paketa uspešan ili ne. Verovatnoća uspešnosti prijema paketa zavisi od odabrane modulacione šeme, odnosa signal-šum (uključujući i interferenciju), kao i statusa fizičkog sloja prijemnog uređaja (na primer, u režimu spavanja ili slanja prijem nije moguć).
2) Odgovarajući objekat u čvoru prati sve primljene signale tako da se u svakom trenutku može proceniti nivo interferencije i doneti odluka o uspešnosti prijema posmatranog paketa.
3) Korisnik može da odabere model proračuna greške pri prijemu paketa u skladu sa verzijom standarda i modulacionom šemom koja se koristi.

NS-3 simulator nudi korisnicima izbor između dva fizička modela sa osnovnim funkcionalnostima definisanim u baznoj klasi `WifiPhy`. Najčešće korišćena izvedena klasa koja se koristi u simulacijama je `YansWifiPhy`. Naziv "Yans" potiče iz vremena početka razvoja NS-3 simulatora, iz rada pod naslovom "*Yet Another Network Simulator*" (Lacage i Henderson, 2006), koji detaljno opisuje implementaciju ovog modela. Ovaj model fizičkog sloja omogućava isključivo analizu Wi-Fi signala i prenos ovih signala kroz odgovarajući Wi-Fi kanal (klasa `YansWifiChannel`). Dakle, samo objekti klase `YansWifiPhy` mogu biti povezani na kanal tipa `YansWifiChannel`, tako da se druge tehnologije koje bi mogle da imaju uticaj na nivo interferencije signala ne mogu ni na koji način pridružiti ovom kanalu. Štaviše, Wi-Fi signali koji ne rade na istoj frekvenciji ne mogu se međusobno povezati zajedničkim kanalom, čime se onemogućava simulacija inteferencije susednih Wi-Fi kanala, čak i u slučajevima kada se koriste kanali koji se delimično preklapaju. Takođe, korišćenje klase `YansWifiPhy` unosi i dodatna ograničenja u pogledu nedostatka podrške za simulaciju frekvencijski zavisnih propagacionih i feding modela.

Implementacija Wi-Fi kanala (klasa `YansWifiChannel`) oslanja se na NS-3 model propagacije za proračun slabljenja i kašnjenja u kanalu. Model propagacije realizovan je u okviru posebnog NS-3 modula `src/propagation`. Postoji veći broj modela slabljenja signala usled propagacije, među kojima su najpoznatiji `FriisPropagationLossModel`, `NakagamiPropagation LossModel`, `OkumuraHataPropagationLossModel`, `TwoRayGroundPropagation LossModel`, itd. Osim ovih modela postoje i modeli za proračun slabljenja u zatvorenom prostoru, koji se baziraju na modelima realizovanim u `src/buildings` modulu. Propagacioni modeli se mogu i nadovezivati jedan na drugi, kreirajući na taj način kompleksnije modele propagacije.

Paketi poslati sa fizičkog sloja u Wi-Fi kanal poseduju pridružen podatak o snazi signala na predaji. Model kanala ima ulogu da isporuči paket svim Wi-Fi uređajima (odnosno odgovarajućim fizičkim slojevima tih uređaja) koji su povezani na isti kanal, pri čemu se za svaki prijemni uređaj izračunava nivo snage signala na prijemu i kašnjenje korišćenjem pridruženog propagacionog modela. Kašnjenje uključuje vreme koje je potrebno za slanje/prijem paketa (i zavisi od brzine signaliziranja i dužine paketa), kao i vreme propagacije signala koje je zadato u propagacionom

modelu. Najčešće se postavlja da je brzina prostiranja talasa jednaka brzini svetlosti, te se kašnjenje dobija deljenjem ove brzine sa rastojanjem predajnika i prijemnika.

U novijim verzijama simulatora pojavio se alternativni model fizičkog sloja, `SpectrumWifiPhy` (novi modul implementiran u modulu `src/wifi`), koji omogućava da se na bazi frekvencijske analize signala modelira međusobni uticaj različitih tehnologija koje koriste isti frekvencijski opseg (na primer Wi-Fi, *Bluetooth* i slične tehnologije). Ovaj model omogućava i proučavanje različitih efekata za koje se koriste frekvencijski zavisni modeli. Bez obzira na razlike, oba fizička modela (`SpectrumWifiPhy` i `YansWifiPhy`) izvedena su iz kalse `WifiPhy` i konfigurišu se na sličan način.

5.3.2. Modeli mobilnosti

Modeli mobilnosti dizajnirani su da opišu pravilo kretanja mobilnih korisnika, kao i kako se njihova lokacija, brzina kretanja i ubrzanje menjaju sa vremenom. Imajući u vidu da modeli mobilnosti imaju veoma značajnu ulogu u određivanju performansi protokola, bilo bi poželjno da oponašaju pravila kretanja iz realnih uslova u što je moguće većoj meri. U suprotnom, simulacioni rezultati mogu biti pogrešno protumačeni i na osnovu toga doneti pogrešni zaključci. Na primer, ukoliko se analizira mreža kod koje se čvorovi kreću međusobno nezavisno, bilo bi pogrešno koristiti modele mobilnosti koji simuliraju grupno kretanje čvorova. Zbog svega navedenog neophodno je imati dublje razumevanje modela mobilnosti i njihovog uticaja na performanse mreže.

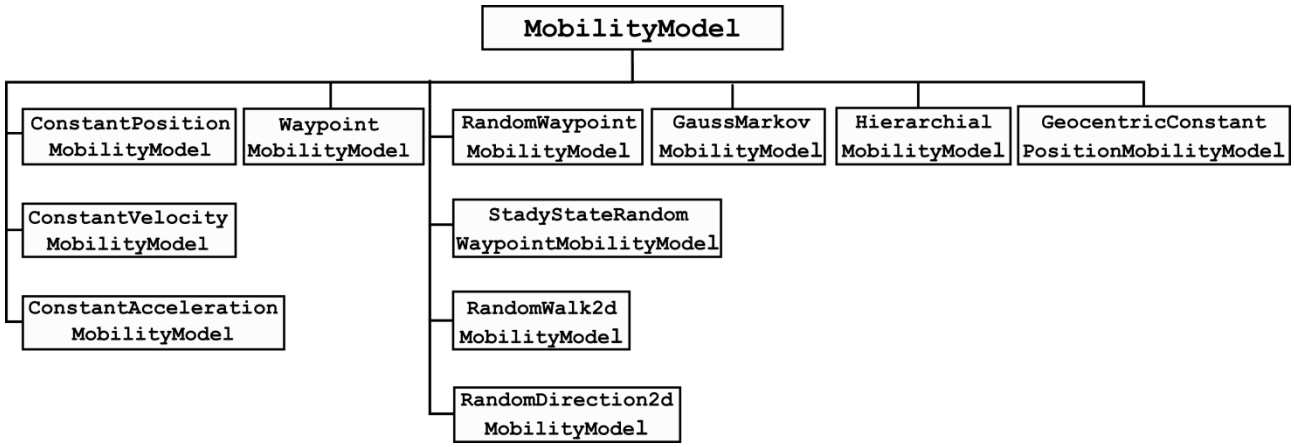
Modeli mobilnosti mogu se podeliti na osnovu specifičnosti karakteristika kretanja u četiri osnovne kategorije (Bai i Helmy, 2004). Najčešća grupa modela mobilnosti je ona kod koje je kretanje čvorova u najvećoj meri slučajno (*random mobility models*). Za neke modele mobilnosti, trenutno kretanje čvorova zavisi od kretanja čvora u prethodnom vremenskom periodu. Takvi modeli nazivaju se modeli mobilnosti sa vremenskom zavisnošću (*mobiliy model with temporal dependancy*). U nekim scenarijima mobilnosti, mobilni čvorovi teže korelisanom kretanju. Takvi modeli nazivaju se modelima mobilnosti sa prostornom zavisnošću (*mobility models with spatial dependency*). Postoje i modeli mobilnosti koji imaju neko geografsko ograničenje (*mobility model with geographic restriction*), kod kojih je kretanje čvorova ograničeno ulicama, saobraćajnicama ili preprekama.

Modele mobilnosti moguće je podeliti i na sintetičke (*synthetic* modeli) i modele zasnovane na praćenju (*tracebased* ili skraćeno *trace* modeli). Sintetički modeli predstavljaju analitičke modele koji su razvijeni na osnovu analiza specifičnih scenarija kretanja. S druge strane, modeli zasnovani na praćenju daju mogućnost da se razumeju realna pravila kretanja i predstavljaju osnovu za analizu statističkih osobina modela kretanja. Međutim, *trace* modeli su uglavnom ograničeni na specifične scenarije kretanja. Na primer, zaključak o kretanju na nekom specifičnom univerzitetskom kampusu sa malim brojem čvorova i pristupnih tačaka ne može se primeniti u gusto naseljenom gradu sa velikim brojem pristupnih tačaka. Zbog toga, u poslednje vreme postoje napori da se precizno prate kretanja u različitim okruženjima tokom dužeg vremenskog perioda (na primer nekoliko godina) kako bi se doneli zaključci o periodičnom ponašanju korisnika koji se mogu generalizovati (Ribeiro i Sofia, 2011).

Pojedini sintetički modeli pokušavaju da simuliraju realno ponašanje ljudi, imajući u vidu karakteristike koje su dobijene u okviru *trace* modela. Na taj način se formiraju i hibridni modeli kretanja koji kombinuju sintetičke modele i statističke podatke dobijene iz *trace* modela. Nezavisno da li su modeli sintetički, *trace* ili hibridni, svi modeli mobilnosti mogu se analizirati sa stanovišta kretanja jednog čvora - individualni modeli (*independent models*) i kretanja grupe čvorova - grupni modeli (*group models*).

5.3.2.1. Klasifikacija modela mobilnosti dostupnih u NS-3 simulatoru

Modeli mobilnosti realizovani u okviru NS-3 slimulatora prikazani su na slici 5.8. Izvorni kôd modela mobilnosti nalazi se u direktorijumu `src/mobility`. Prva grupa modela predstavlja jednostavne modele mobilnosti, kod kojih su unapred zadati neki parametri i tokom trajanja simulacije se ne menjaju. Na primer, `ConstantPositionMobilityModel` (CP) u stvari je statički model kod kog se pozicija čvora ne menja kada je jednom postavljena, sve dok se eventualno ne postavi na neku drugu vrednost. Kod `ConstantVelocityMobilityModel` i `ConstantAccelerationMobilityModel` modela brzina, odnosno ubrzanje čvora, ne menjaju se tokom trajanje simulacije, osim ukoliko se eventualno ne postave na neku novu vrednost.



Slika 5.8. Modeli mobilnosti realizovani u NS-3 simulatoru

Kod modela `WaypointMobilityModel` svaki čvor određuje svoju brzinu i poziciju iz skupa objekata tipa `Waypoint`. Svaki objekat tipa `Waypoint` (*waypoint* tačka) sadrži podatak o vremenskom trenutku (`waypointTime`) u kom čvor treba da stigne na narednu poziciju (`waypointPosition`). Ukoliko je trenutno vreme simulacije veće od vremena `waypointTime` neke *waypoint* tačke, ta tačka se odbacuje. Podrazumevano je da inicijalna pozicija svakog čvora odgovara poziciji prve *waypoint* tačke i da je inicijalna brzina svakog čvora jednaka nuli. Kada čvor dođe do poslednje *waypoint* tačke, njegova pozicija postaje statična i brzina ponovo jednaka nuli. Kretanje između dve susedne *waypoint* tačke ostvaruje se konstantnom brzinom. Ukoliko korisnik želi da se čvor nalazi u nekoj poziciji duži vremenski period (da miruje) neophodno je da se u dve susedne *waypoint* tačke unese ista pozicija, ali različito vreme.

Grupa modela mobilnosti zasnovanih na slučajnom kretanju najčešće je korišćena u simulacionim scenarijima primenjenim u javno dostupnoj literaturi. Osnovni model iz ove grupe je `RandomWaypointMobilityModel` (RW), kod kog se čvorovi kreću prema slučajno izabranom odredištu. Brzina kojom se čvor kreće prema odredištu takođe se bira slučajno, kao i pauza koja se pravi kada čvor dostigne željeno odredište. Kod RW modela brzina kretanja i pauza predstavljaju nezavisne slučajne promenljive čiju raspodelu je moguće konfigurisati po potrebi. Za razliku od njega, kod `StadyStateRandomWaypointMobilityModel` (SSRW) modela slučajne promenljive koje opisuju poziciju, brzinu i pauzu u kretanju uvek imaju uniformnu raspodelu. Razlika je u tome što početne vrednosti ovih parametara ne uzimaju vrednosti iz uniformne raspodele, već iz stacionarne raspodele (*stationary distribution*) RW modela mobilnosti (Navidi i Camp, 2004).

U okviru NS-3 simulatora dostupne su i dve varijante modela mobilnosti koje su izvedene iz RW modela: `RandomWalk2dMobilityModel` (RWK) i `RandomDirection2dMobility Model` (RD). Kod RWK modela svaki čvor se kreće slučajno izabranom brzinom i pravcem sve dok ne pređe fiksno rastojanje ili ne prođe neko određeno vreme. Za razliku od RW modela, kod RWK modela nema pauza u kretanju. Kod RD modela, svaki čvor bira slučajno vreme pauze, brzinu i pravac kretanja, međutim ne bira tačku ka kojoj će se kretati, već se kreće do ivice simulacione oblasti. Kada dostigne ivicu, čeka vreme pauze i nakon toga bira novi smer i brzinu kretanja i postupak se ponavlja.

Problemi koji se javljaju kod RW i sličnih modela mobilnosti su situacije kada se čvorovi kreću pod oštrim uglom i situacije kada dolazi do iznenadnog zaustavljanja čvorova, što ne odgovara situacijama iz realnog života. Ovi problemi mogu se eliminisati ukoliko se dozvoli da prethodna brzina i smer kretanja utiču na buduću brzinu i smer kretanja. Jedan od načina da se modelira ovakvo kretanje, koje je realnije nego kod RW modela, je korišćenje `GaussMarkovMobility Model` (GM) modela mobilnosti. Za razliku od drugih modela mobilnosti u NS-3, koji su bez memorije, GM model ima i memoriju i slučajnost.

Naredni model prikazan na slici 5.8 je `HierarchicalMobilityModel`, kojim je omogućeno da se pozicija objekta nižeg nivoa (*child* objekta) podešava u zavisnosti od pozicije objekta višeg nivoa (*parent* objekata). U osnovi ovaj model mobilnosti može da kombinuje dva druga modela mobilnosti: *parent* i *child* model. Pozicija čvora u ovom modelu predstavlja vektorski zbir pozicija čvora kod *parent* i *child* modela. Na taj način, ukoliko je za *parent* model izabran neki model kretanja, rezultat će biti relativno kretanje u odnosu na taj model. Ovaj model može biti koristan ukoliko je potrebno simulirati grupnu mobilnost čvora, kada se kretanje jednog čvora vezuje za kretanje drugog čvora, kao što se recimo može dogoditi sa vozilima. Podešavanje pozicija ovog modela uvek se vrši u apsolutnim koordinatama i menja samo poziciju *child* modela mobilnosti, nikad *parent* modela. *Child* model mobilnosti uvek koristi relativne koordinate u odnosu na *parent* model.

Konačno, `GeocentricConstantPositionMobilityModel` je jedini geografski model mobilnosti u NS-3 simulatoru, koji koristi geocentrične koordinate (*x*, *y*, *z*) umesto klasičnih lokalnih koordinata. To ga čini pogodnim za simulacije satelitskih mreža i drugih scenarija gde su čvorovi vezani za globalni referentni sistem. Ovaj model ne uključuje mobilnost čvorova, već oni ostaju na istoj poziciji tokom trajanja simulacije. Može se koristiti u kombinaciji sa drugim modelima mobilnosti koji omogućavaju kretanje kako bi se simulirali dinamički satelitski sistemi. Takođe, zbog geocentričnog pristupa, omogućava lakšu integraciju sa modelima koji koriste realne GPS koordinate ili astronomske proračune.

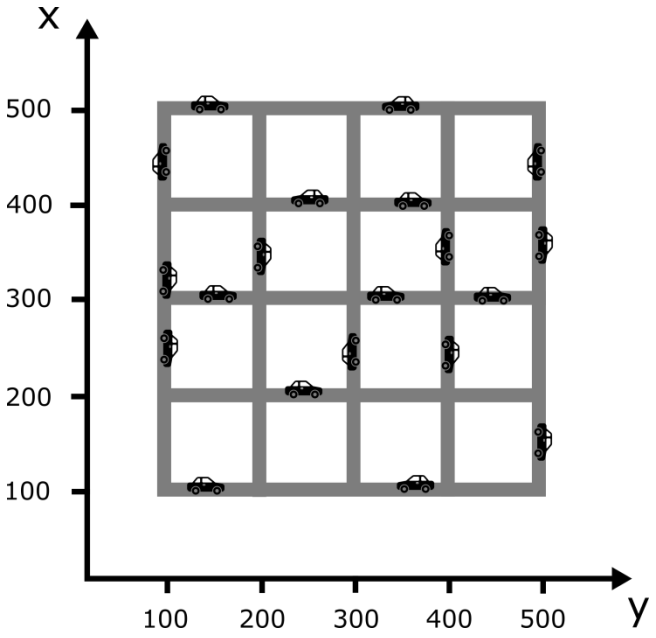
5.3.2.2. Slučajne promenljive i modeli mobilnosti

Analizom realizacija modela mobilnosti u okviru NS-3 simulatora, može se primetiti da se kretanje čvorova bazira na korišćenju slučajnih promenljivih. Kako je neophodno porediti rezultate simulacija pod ravnopravnim uslovima, u pojedinim slučajevima važno je da se model mobilnosti ponaša na isti način u različitim simulacionim scenarijima. Na primer, ukoliko bi se poredila dva protokola rutiranja poželjno je da se kod oba simulaciona scenarija čvorovi kreću na identičan način kako bi ovo poređenje bilo ravnopravno. Međutim, promenom konfiguracionog skirpta može se desiti da slučajna promenljiva dodeljena određenom parametru modela mobilnosti u različitim simulacijama uzima različite vrednosti, iako su globalni parametri `RngRun` i `RngSeed` izabrani na isti način. To potencijalno može da dovede do značajnih razlika u kretanju čvorova u simulacionom scenariju, čime se gubi ravnopravnost pri poređenju rezultata simulacija.

Ovaj problem moguće je rešiti na dva načina. Prvi je višestruko ponavljanje simulacija za različite vrednosti `RngRun` parametra i usrednjavanje dobijenih rezultata. Međutim, ovo rešenje je vremenski veoma zahtevno. Drugi način je korišćenje posebne funkcije `AssignStreams (streamIndex)`, obezbeđene u modelima mobilosti. Za izvestan broj slučajnih promenljivih mogu se rezervisati tačno određeni strimovi slučajnih brojeva definisanih indeksom strima (`streamIndex`). Ova funkcija omogućava da određeni model, za svaku slučajnu promenljivu koju koristi, rezerviše po jedan strim slučajnih brojeva, redom počev od indeksa strima (`streamIndex`) koji je prosleđen kao argument ove funkcije.

5.3.2.3. Manhattan grid model

Pored modela mobilnosti koji su direktno realizovani u NS-3 simulatoru, kretanje čvorova je moguće modelovati u nekom pomoćnom simulacionom alatu (kao što je SUMO simulator) i učitati u simulacioni skript NS-3 simulatora. Jedan od takvih modela je *Manhattan grid* (MG), koji je veoma pogodan za simulaciju kretanja vozila u urbanim sredinama. Ovo je urbani tip modela mobilnosti za VANET mreže, koji koristi mrežastu (*grid*) topologiju. Podrazumeva organizovano kretanje vozila ulicama koje se protežu u horizontalnom i vertikalnom pravcu. MG model koristi probabilistički pristup u odabiru kretanja vozila, jer na svakoj raskrsnici vozilo sa određenom verovatnoćom bira da li će skrenuti ili nastaviti da ide u istom pravcu. Na slici 5.9 prikazan je jednostavan MG model sa po 5 horizontalnih i vertikalnih ulica (rastojanje između raskrsnica je 100 m), kojima se na slučajan način kreće 20 vozila.



Slika 5.9. Ilustracija MG modela mobilnosti

5.3.3. Protokoli rutiranja i određivanje overheda u NS-3 simulatoru

Za modelovanje određene *ad hoc* mreže u NS-3 simulatoru, veoma je bitno izabrati odgovarajući protokol rutiranja kojim će biti definisane putanje za slanje podataka u mreži. U drugom poglavlju su predstavljeni različiti tipovi protokola rutiranja za WANET mreže, a detaljnije su opisani najznačajniji reaktivni i proaktivni protokoli (AODV, DSR i DSDV). Ova tri protokola su već implementirana u NS-3 simulator, tako da ih je moguće jednostavno koristiti za modelovanje procesa rutiranja paketa. Izvorni kôd AODV modela nalazi se u direktorijumu `src/aodv`. Klasa `aodv::RoutingProtocol` implementira sve funkcionalnosti servisa razmene paketa. Ova klasa definiše dve virtuelne funkcije za rutiranje i prosleđivanje paketa. Prva virtuelna funkcija

`aodv::RouteOutput()` koristi se za lokalne pakete, a druga `aodv::RouteInput()` se koristi za prosleđivanje i/ili isporučivanje primljenih paketa. Implementacija DSR protokola u NS-3 simulator izvršena je na bazi RFC 4728 preporuke, uz određene ekstenzije i modifikacije. Direktorijum u kom je smešten kôd DSR protokola je `src/dsr`. Slično tome, svi fajlovi za implementaciju DSDV protokola smešteni su u direktorijum `src/dsdv`.

Vrlo važan parametar u analizi svake *ad hoc* mreže je određivanje overheda protokola rutiranja. Često se smanjenje kašnjenja paketa aplikativnog saobraćaja može postići uz povećanje overheda, kao što je slučaj kod proaktivnih protokola. Najjednostavniji i praktično univerzalan način da se odredi overhed protokola rutiranja je korišćenjem PCAP sistema za praćenje (opisan u poglavlju 5.2.5.2.). Kako bi se ovo omogućilo, neophodno je omogućiti snimanje paketa koji se šalju preko Wi-Fi interfejsa u PCAP datoteku. Analizom ovih paketa u nekom pogodnom alatu, kao što je *WireShark*, mogu se filtrirati samo paketi protokola rutiranja i na taj način odrediti overhed. Određivanje overheda korišćenjem PCAP sistema za praćenje ima veliku prednost u tome što se veoma jednostavno instalira u NS-3 skript. Međutim, značajan nedostatak je što korisnik mora da poseduje instalaciju i solidno poznavanje nekog dodatnog alata kao što je *WireShark*, kako bi odredio overhed. Takođe, veoma veliki nedostatak ovakvog pristupa posebno dolazi do izražaja kada se za potrebe kvalitetne statističke analize u određenom eksperimentu izvršava veoma veliki broj simulacija. U tom slučaju potrebno je za svaku simulaciju učitati jedan ili više fajlova, te se obrada podataka dosta komplikuje.

Značajno efikasniji način za analizu overheda je svakako opšti sistem za praćenje, koji se zasniva na izvorima i sakupljačima podataka, kao što je objašnjeno u poglavlju 5.2.5.2. Izvori podataka su veoma efikasan način za dobijanje informacija jer se koriste samo po potrebi, odnosno samo ako je na izvor povezan odgovarajući sakupljač podataka. Različiti modeli u NS-3 simulatoru obezbeđuju realtivno veliki broj izvora podataka, ali na žalost modeli protokola rutiranja vrlo slabo podržavaju ili uopšte ne podržavaju ovaj sistem. AODV protokol, na primer, ne poseduje nijedan izvor podataka za praćenje.

Da bi se omogućilo merenje overheda neophodno je da se od AODV modela u NS-3 simulatoru dobije informacija o svakom paketu koji emituje protokol rutiranja, svakog čvora mreže. Ovaj izvor podrazumeva da se pri svakom generisanju kontrolnog paketa (klase `Packet`) protokola rutiranja, ovaj paket istovremeno prosledi i svim sakupljačima podataka. Za to je neophodno da se svaki put pri kreiranju novog kontrolnog paketa, on istovremeno prosledi preko izvora podataka za praćenje. Ovaj izvor podrazumeva da se sakupljači podataka projektuju tako da primaju podatak tipa pametnog pokazivača na paket (`Ptr<const Packet>`). Sakupljači podataka po prijemu paketa mogu dalje vršiti obradu na različite načine. Recimo, mogu brojati pristigle pakete, određivati ukupan ostvareni saobraćaj, definisati statistiku prema tipu paketa i slično. Da bi se obezbedio ispravan rad, potrebno je povezati izvor i sakupljač podataka korišćenjem funkcije `Config::Connect()`.

5.3.4. Aplikacije za generisanje saobraćaja

Kada su svi čvorovi i kompletna mreža konfigurisani na željeni način, potrebno je pokrenuti simulaciju. Međutim, da bi simulacija imala određeni smisao potrebno je osmisliti scenario u kome treba da se obezbedi da neki od čvorova generišu, a neki primaju saobraćaj generisan na aplikativnom sloju. U tu svrhu u NS-3 simulatoru se koriste generatori saobraćaja koje se izvode iz klase `Application`. Definicija ove klase nalazi se u modulu `src/network`, dok se konkretne aplikacije, naslednice klase `Application`, koje generišu konkretan profil saobraćaja nalaze u različitim modulima. Posebno su zanimljive aplikacije koje generišu test saobraćaj i koje se nalaze u modulu `src/applications` ili aplikacije koje su karakteristične za Internet mreže i koje se

nalaze u modulu `src/internet-apps`. Takođe, korisnik može i sam da kreira aplikacije koje će generisati neko specifično saobraćajno opterećenje.

Iako je u NS-3 simulatoru obezbeđen veći broj aplikacija za generisanje saobraćaja, korisnici često imaju potrebu da definišu sopstvene aplikacije. Razlozi za to mogu biti različiti, od potrebe da se definiše specifičan profil saobraćajnog opterećenja, pa do potrebe da se unesu specifični izvori za praćenje u cilju generisanja željenih statistika ili da se dodaju posebna zaglavlja za identifikaciju paketa kako bi se precizno pratio tok svakog paketa kroz mrežu. Da bi korisnik mogao da kreira sopstvenu aplikaciju potrebno je da ona bude naslednica klase `Application`. Od podataka članova, svaka aplikacija mora da poseduje pametni pokazivač na čvor kome pripada, vreme započinjanja i završetka slanja paketa i događaje koji su uneseni u planer (*scheduler*) i koji odgovaraju vremenima za početak i kraj slanja paketa. Osnovna funkcija bazne klase `Application` je da obezbedi funkcionalnosti koje se tiču započinjanja i završetka rada aplikacije. Sve druge detalje o načinu slanja paketa potrebno je realizovati u konkretnim klasama naslednicama. Treba napomenuti da se osim same klase `Application` u skriptovima često koriste i pomoćne klase koje olakšavaju proces instalacije određenog tipa aplikacije istovremeno u više čvorova, kao i klasa `ApplicationContainer` (definisana u `/src/networ/helper`) koja omogućava da se istovremeno zada početak i kraj rada grupi aplikacija, umesto da se to radi za svaku aplikaciju pojedinačno.

5.4. Simulatori mobilnosti čvorova

Simulatori mobilnosti saobraćaja su softverski alati koji omogućavaju modeliranje i simulaciju kretanja vozila, pešaka i drugih učesnika u saobraćaju u različitim okruženjima. Oni se mogu koristiti kao pomoćan alat mrežnim simulatorima, kako bi se generisali složeniji scenariji kretanja mrežnih čvorova. Neki od najpoznatijih simulatora mobilnosti čvorova su SUMO, "*Verkehr In Städten – SIMulationsmodell*" (VSSIM), *Multi-Agent Transport Simulation* (MATSim) i *Advanced Interactive Microscopic Simulator for Urban and Non-Urban Networks* (AIMSUN). U okviru ove disertacije, kao pomoćni alat za generisanje kretanja vozila korišćen je SUMO simulator.

SUMO je *open source*, prenosivi, višemodalni softverski paket za simulaciju saobraćaja, dizajniran za rad sa velikim saobraćajnim mrežama. U razvoju SUMO simulatora prvenstveno učestvuju istraživači Instituta za transportne sisteme pri Nemačkom centru za svemirska istraživanja (*German Aerospace Center*), a takođe razvoju pomaže i korisnička zajednica. Dostupan je besplatno kao otvoreni kod od 2001. godine, a od 2017. postao je projekat pod okriljem *Eclipse Foundation*. Licenciran je pod *Eclipse Public License* 2.0 (EPL 2.0). Aktuelna verzija simulatora je SUMO 1.22.0 (SUMO, 2025). Omogućava intermodalne simulacije, uključujući pešake, i dolazi s bogatim skupom alata za kreiranje saobraćajnih scenarija. Za realistično kreiranje gradskih i ruralnih mreža, SUMO omogućava učitavanje mapa iz realnog okruženja putem *Open Street Maps* (OSM). OSM je besplatna, otvorena kartografska baza podataka koja sadrži detaljne informacije o putevima, raskrsnicama, zonama i drugim geografskim elementima.

Takođe, SUMO simulator ima mogućnost modeliranje kretanja vozila kroz mrežu ulica koja je kreirana pomoću MG modela. U okviru ovog modela, SUMO omogućava definisanje broja horizontalnih i vertikalnih ulica, rastojanja između ulica, broja traka u ulicama, kao i tipova raskrsnica (sa semaforima ili bez). Za generisanje kretanja vozila, najpre je potrebno definisati njihovu početnu tačku i vreme polaska. Kretanje vozila može biti nasumično ili po unapred definisanim putanjama. Takođe je moguće definisati ukupan broj vozila koja se kreću u mreži, kao i maksimalnu dozvoljenu brzinu kretanja vozila. Upravo ovo će biti iskorišćeno u narednom poglavlju, kako bi bila izvršena simulacija kretanja vozila u VANET scenarijima u koima će biti testirani protokoli rutiranja.

Simulacija saobraćaja u SUMO simulatoru koristi softverske alate za simulaciju i analizu drumskog saobraćaja i sistema upravljanja saobraćajem. Nove strategije upravljanja saobraćajem mogu se implementirati u simulaciji radi analize pre nego što budu primenjene u realnim situacijama. SUMO je takođe predložen kao deo lanca alata za razvoj i validaciju funkcija automatizovane vožnje putem različitih pristupa kao što su *X-in-the-Loop* i digitalni blizanci (*digital twin*). SUMO se koristi i u istraživačke svrhe kao što su prognoza saobraćaja, evaluacija semafora, odabir putanja ili u oblasti sistema za komunikaciju između vozila. Korisnici SUMO simulatora mogu da prave izmene u izvornom kodu programa putem *open source* licence, što im omogućava da eksperimentišu s novim pristupima.

6. SIMULACIONA ANALIZA

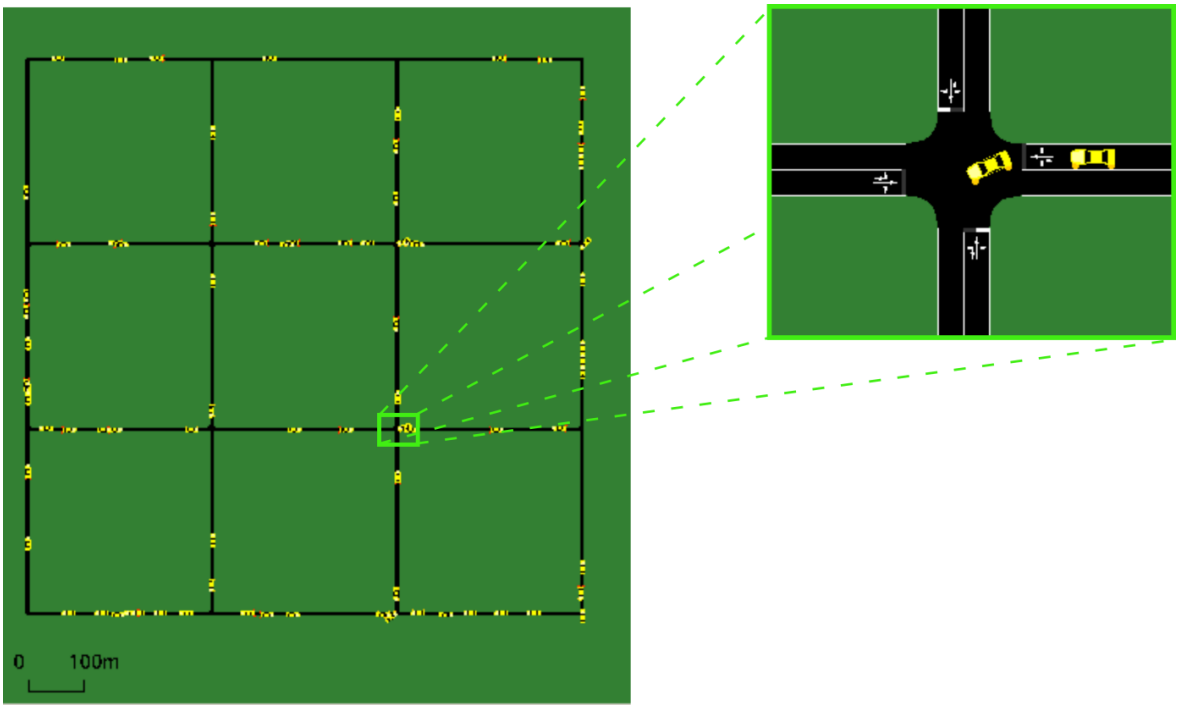
U ovom poglavlju je predstavljena simulaciona analiza novog Q-DRAV protokola rutiranja, koji je upoređen sa ranije opisanim QLAODV i ARPRL protokolima (koji su izabrani za predstavnike RL baziranih protokola rutiranja), kao i sa popularnim AODV protokolom (koji je izabran za predstavnika tradicionalnih protokola rutiranja). Cilj simulacione analize je da pruži uvid u ponašanje protokola u saobraćajnim uslovima koji približno odgovaraju realnim. Zato je jako bitno pogodno izabrati simulacione scenarije i pažljivo podesiti parametre simulacije. Za testiranje protokola izabran je NS-3 simulator, za koji je pregledom aktuelne literature utvrđeno da je najčešće korišćen simulacioni alat za evaluaciju i testiranje ranije publikovanih protokola rutiranja za VANET i FANET mreže. Još jedan od glavnih razloga zašto je izabran NS-3 simulator je taj što spada u *open source* simulatore, tako da je bilo moguće implementirati modele novih protokola. U okviru ovog simulatora već je implementiran model AODV protokola. S druge strane, model novog Q-DRAV protokola, kao i modele QLAODV i ARPRL protokola, bilo je neophodno naknadno implementirati u ovaj simulator kako bi bilo moguće izvršiti njihovo testiranje i poređenje (Jevtić i Bugarčuć, 2022; Jevtić i Bugarčić, 2023; Jevtić i ostali, 2023). Takođe, kako bi se drugim istraživačima olakšao razvoj budućih protokola rutiranja i omogućilo jednostavnije poređenje sa predloženim Q-DRAV protokolom rutiranja, implementacioni kôd ovog protokola u NS-3 simulatoru javno je dostupan (GitHub, 2024).

6.1. Simulacioni parametri

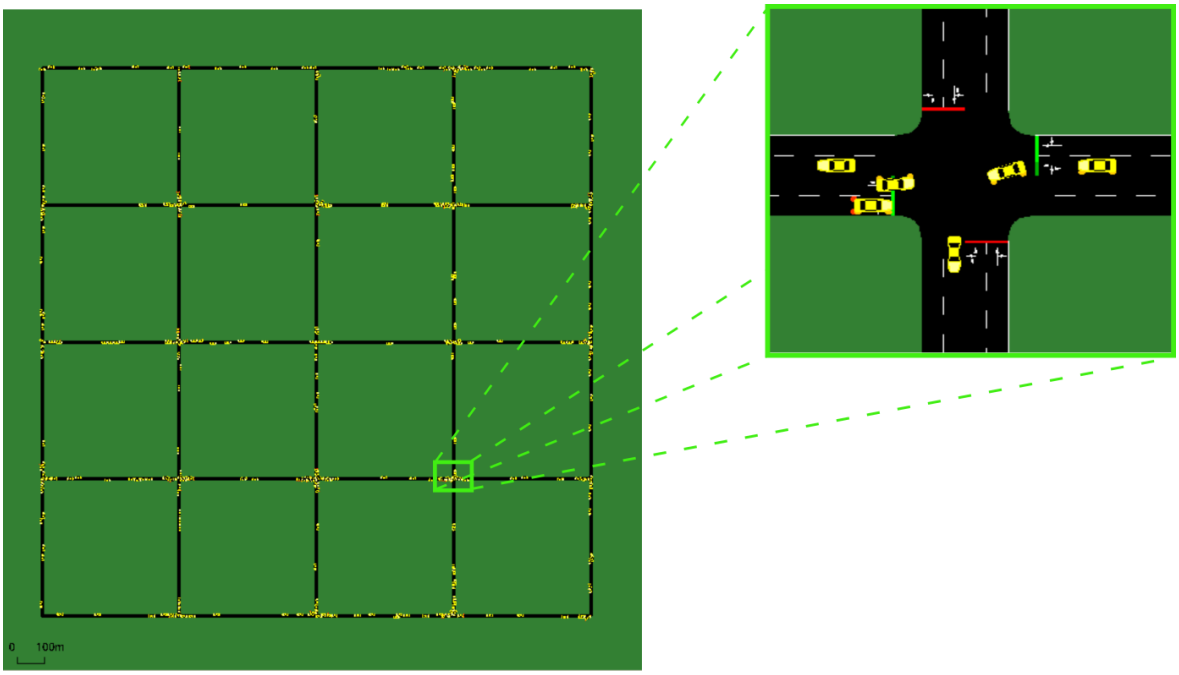
Imajući u vidu da se VANET mreže mogu implementirati u gradskim područjima koja zauzimaju manje ili veće oblasti, veoma je važno da protokoli rutiranja postižu dobre performanse u oba tipa ovih područja. Iako je QLAODV inicijalno testiran u (C. Wu i ostali, 2010) kako za mala, tako i za velika gradska područja, u (J. Wu i ostali, 2018) je poređen sa ARPRL protokolom samo za velike mreže. Takođe, QLAODV inicijalno nije testiran za veoma guste mreže (preko 300 čvorova), koje se često susreću u urbanim gradskim sredinama. Zbog toga su sprovedeni dodatni testovi (u malim i velikim mrežnim područjima, za varijabilne maksimalne dozvoljene brzine i gustine čvorova u mreži), kako bi se ukazalo na nedostatke postojećih protokola i predložio metod za njihovo unapređenje. Protokoli su testirani u dva različita scenarija, koji približno odgovaraju scenarijima u kojima su originalno testirani QLAODV i ARPRL protokoli u (C. Wu i ostali, 2010) i (J. Wu i ostali, 2018), respektivno. Ukupno je izvršeno 2400 simulacija u prvom scenariju (4 protokola, po 6 različitih maksimalnih dozvoljenih brzina kretanja vozila i po 100 ponavljanja za svaku tačku) i 2800 simulacija u drugom scenariju (4 protokola, po 7 različitih gustina vozila i po 100 ponavljanja za svaku tačku).

Prvi scenario podrazumeva manju simulacionu oblast sa fiksnim brojem vozila, a cilj je uvideti kako se menjaju mrežne performanse pri variranju maksimalne dozvoljene brzine kretanja vozila. Dimenzije simulacionog područja su 1000 m x 1000 m, sa po 4 horizontalne i vertikalne ulice i po jednom kolovoznom trakom u svakom smeru. Unutar ovog područja kreće se 80 vozila od kojih 30 (nasumično izabranih) generiše *Constant Bit Rate* (CBR) saobraćaj brzine 32 kb/s. Ograničenje maksimalne dozvoljene brzine kretanja vozila varira od 1 m/s do 25 m/s (3,6 km/h do 90 km/h). Za kreiranje ovog scenarija korišćen je *Manhattan Grid* model mobilnosti (opisan u poglavlju 5.3.2.3) i SUMO simulacioni alat (opisan u poglavlju 5.4), iz kog je model kretanja vozila ekstraktovan i

iskorišćen za simulacije u NS-3 simulatoru. Na slici 6.1 prikazana je ilustracija prvog scenarija u SUMO simulatoru.



Slika 6.1. Ilustracija prvog simulacionog scenarija



Slika 6.2. Ilustracija drugog simulacionog scenarija

Drugi scenario podrazumeva veću simulacionu oblast sa fiksnom maksimalnom dozvoljenom brzinom kretanja vozila, a cilj je uvideti kako se menjaju mrežne performanse pri variranju gustine vozila u mreži (odnosno ukupnog broja vozila u mreži). Dimenzije simulacionog područja u ovom scenariju su 2000 m x 2000 m, sa po 5 horizontalnih i vertikalnih ulica. Svaka ulica ima po dve kolovozne trake za kretanje vozila u oba smera, a svaka raskrsnica ima semafor. Maksimalna dozvoljena brzina kretanja vozila je ograničena na 15 m/s (54 km/h). Ukupan broj vozila u mreži

varira od 50 do 350 (sa korakom od 50 vozila), pri čemu uvek 20 nasumično izabranih vozila generiše CBR saobraćaj brzine 4 kb/s. Dakle, iako je ukupan broj vozila uglavnom veći u odnosu na prvi scenario, broj vozila koja generišu saobraćaj i brzina ovog saobraćaja su znatno manji. Kao i u prvom scenariju, za kreiranje i generisanje modela kretanja vozila korišćeni su *Manhattan Grid* model mobilnosti i SUMO simulacioni alat. Na slici 6.2 prikazana je ilustracija drugog scenarija u SUMO simulatoru.

Ostali simulacioni parametri su identični za oba scenarija. Na MAC podsloju se koristi IEEE 802.11p standard za bežične mreže, sa širinom kanala od 10 MHz i protokom od 6 Mb/s. Za model propagacije odabran je *Two Ray Ground*, a na transportnom sloju je korišćen UDP protokol. Ovi modeli su već implementirani u NS-3 simulator, tako da ih je jednostavno uključiti u simulacioni scenario. Kao što je već rečeno, na mrežnom sloju su primenjeni AODV, QLAODV, ARPRL i Q-DRAV protokoli rutiranja. Veličina paketa na aplikacionom sloju je 512 bajta. Trajanje jedne simulacije je ograničeno na 600 sekundi. Pregled najvažnijih parametara prvog i drugog simulacionog scenarija prikazani su u tabelama 6.1 i 6.2, respektivno.

Tabela 6.1. Pregled parametara prvog simulacionog scenarija

Parametar	*Vrednost*
Dimenzije prostora	1000 m x 1000 m
Model mobilnosti	*Manhattan Grid*
Broj ulica	po 4 horizontalne i vertikalne ulice
Trajanje simulacije	600 s
Ukupan broj vozila	80
Maksimalna brzina vozila	1 m/s, 5 m/s, 10 m/s, 15 m/s, 20 m/s, 25 m/s
Propagacioni model	*Two Ray Ground*
Protokol na MAC podsloju	IEEE 802.11p
Širina kanala na MAC podsloju	10 MHz
Protok na MAC podsloju	6 Mb/s
Protokol rutiranja	AODV, QLAODV, ARPRL, Q-DRAV
Transportni protokol	UDP
Aplikacioni protok po vozilu	32 kb/s
Broj vozila koja generišu aplikacioni saobraćaj	30
Veličina paketa na aplikacionom sloju	512 B

Tabela 6.2. Pregled parametara drugog simulacionog scenarija

Parametar	*Vrednost*
Dimenzije prostora	2000 m x 2000 m
Model mobilnosti	*Manhattan Grid*
Broj ulica	po 5 horizontalnih i vertikalnih ulica
Trajanje simulacije	600 s
Ukupan broj vozila	50, 100, 150, 200, 250, 300, 350
Maksimalna brzina vozila	15 m/s
Propagacioni model	*Two Ray Ground*
Protokol na MAC podsloju	IEEE 802.11p
Širina kanala na MAC podsloju	10 MHz
Protok na MAC podsloju	6 Mb/s
Protokol rutiranja	AODV, QLAODV, ARPRL, Q-DRAV
Transportni protokol	UDP
Aplikacioni protok po vozilu	4 kb/s
Broj vozila koja generišu aplikacioni saobraćaj	20
Veličina paketa na aplikacionom sloju	512 B

6.2. Rezultati simulacija

Poređenje protokola izvršeno je posmatrajući sledeće mrežne performanse: PLR, ostvareni protok na aplikacionom sloju, prosečan E2ED i džiter. Zbog verodostojnosti rezultata, za svaki parametar je izračunata prosečna vrednost na osnovu 100 rezultata simulacija, sprovedenih za različite vrednosti generatora slučajnih brojeva (menjan je `RngRun` parametar pri pokretanju simulacija, kao što je objašnjeno u poglavlju 5.2.3).

Pored prosečnih vrednosti parametara, određen je i 95% interval poverenja za svaku izračunatu vrednost, kako bi se potvrdila pouzdanost rezultata simulacija. Ovaj interval predstavlja opseg vrednosti unutar kojeg se, sa 95% verovatnoće, može očekivati da se nalazi prosečna vrednost na osnovu podataka iz uzorka. Interval poverenja (*I*) se računa pomoću sledeće jednačine:

$$I = \bar{y} \pm Z \cdot \frac{\sigma}{\sqrt{n}}. \tag{6.1}$$

U ovoj relaciji *Z* zavisi od raspodele posmatranog parametra *y*, *n* označava veličinu uzorka (100 u našem slučaju), $\bar{y}$ predstavlja izračunatu prosečnu vrednost parametra *y* (na osnovu rezultata 100 simulacija), a σ predstavlja standradnu devijaciju. Poslednje dve veličine se računaju na osnovu sledećih jednačina:

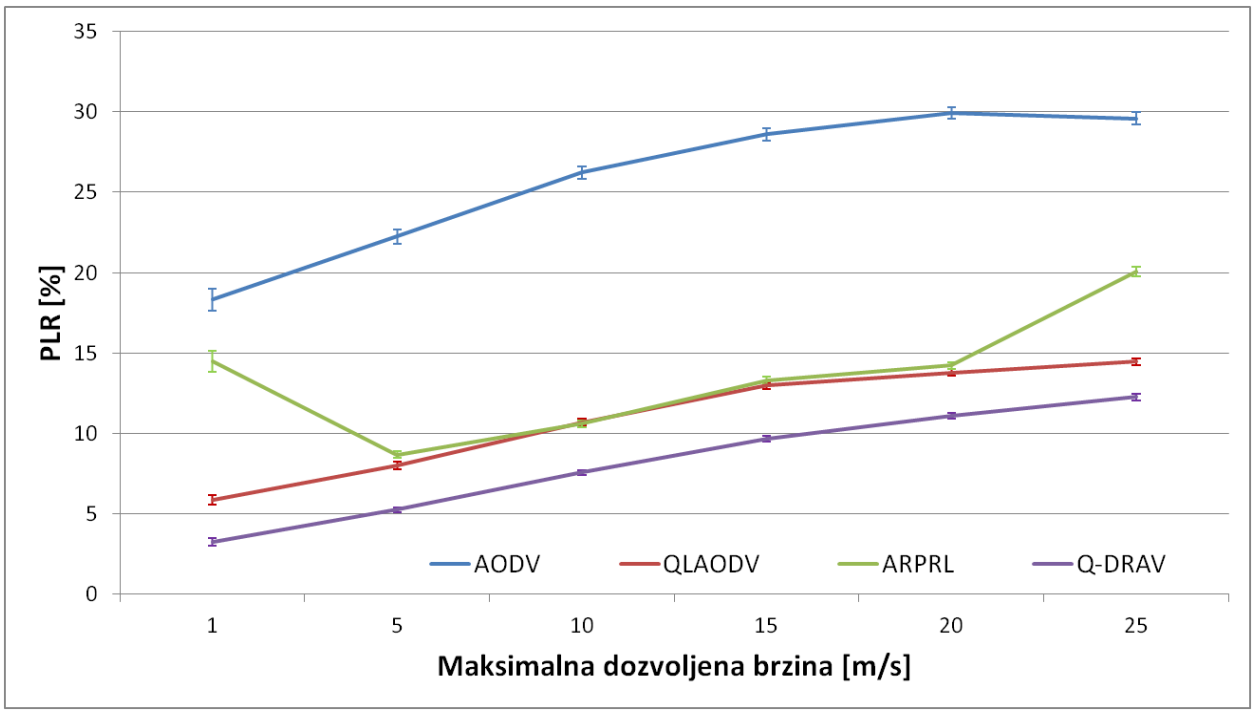
$$\bar{y} = \frac{\sum_{i=1}^{n} y_i}{n}, \tag{6.2}$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \bar{y})^2}{n}}. \tag{6.3}$$

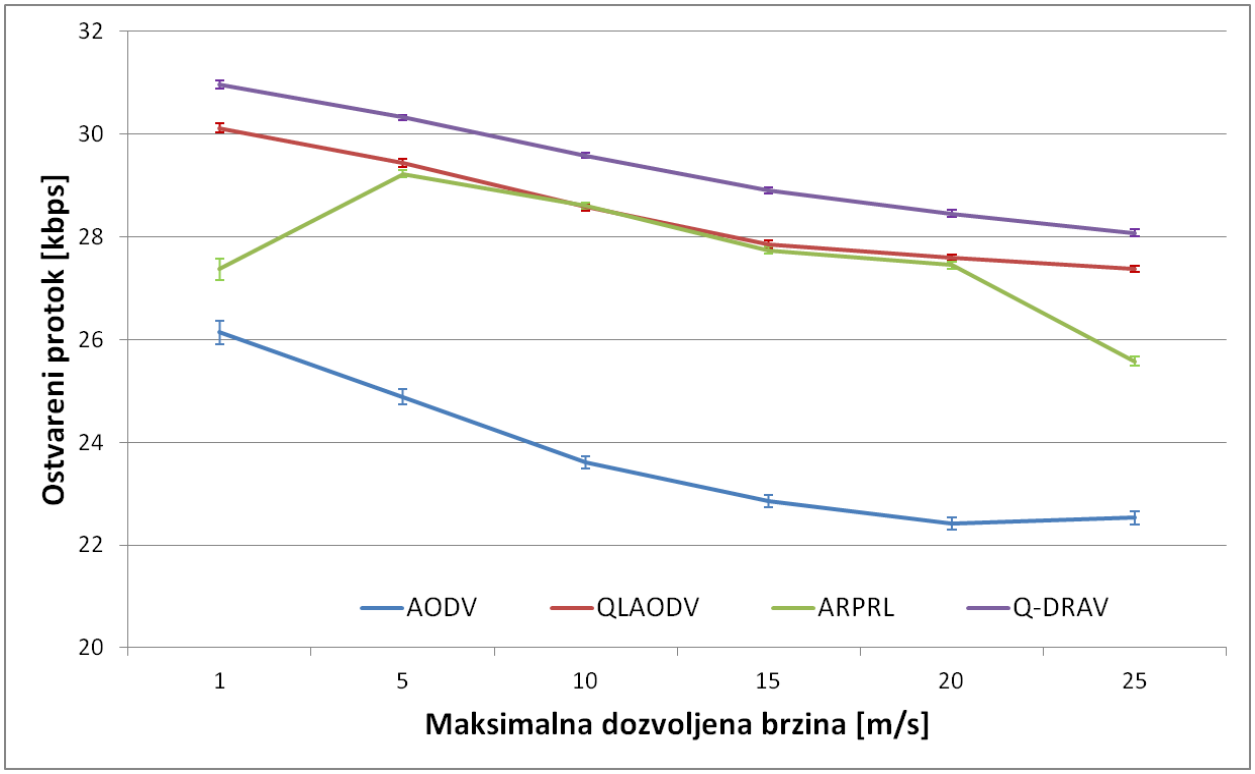
U jednačinama (6.2) i (6.3), y_i predstavlja vrednost posmatranog parametra dobijenu iz *i*-te simulacije (za `RngRun` = *i*). Interval poverenja za parametar *y* će onda biti u granicama od $\bar{y} - Z \cdot \frac{\sigma}{\sqrt{100}}$ do $\bar{y} + Z \cdot \frac{\sigma}{\sqrt{100}}$. Što je interval poverenja uži, to je preciznost rezultata simulacija veća.

Prosečne vrednosti parametara i njihovi odgovarajući 95% intervali poverenja prikazani su na slikama 6.3 do 6.6 za prvi i na slikama 6.7 do 6.10 za drugi scenario. Primetno je da se u oba scenarija interval poverenja nalazi unutar zadovoljavajućih granica, čime je potvrđena pouzdanost dobijenih rezultata simulacija. Može se videti da kada su rezultati bolji, tada je interval poverenja uži, tako da se sa velikom pouzdanošću može utvrditi koji protokol daje nabolje rezultate po pitanju pojedinih parametara.

Slike 6.3 i 6.4 prikazuju zavisnost PLR i ostvarenog protoka na aplikacionom sloju od maksimalne dozvoljene brzine kretanja vozila u prvom scenariju. Kao što je i očekivano, sa povećanjem brzine kretanja vozila svi parametri blago degradiraju, kod gotovo svih protokola. Može se zaključiti da QLAODV i ARPRL imaju značajno niže gubitke paketa i veći ostvareni aplikacioni protok u poređenju sa AODV protokolom. Primetno je ipak da ARPRL pokazuje priličnu degradaciju ovih parametara pri krajnje malim i velikim maksimalnim dozvoljenim brzinama kretanja vozila (od 1 m/s i 25 m/s). S druge strane, novi Q-DRAV protokol značajno nadmašuje QLAODV i ARPRL po ovim parametrima, kako pri nižim, tako i pri višim maksimalnim dozvoljenim brzinama vozila. Sve to Q-DRAV postiže uz izuzetno uzak interval poverenja za sve posmatrane maksimalne dozvoljene brzine, što govori o velikoj pouzdanosti ovih rezultata.



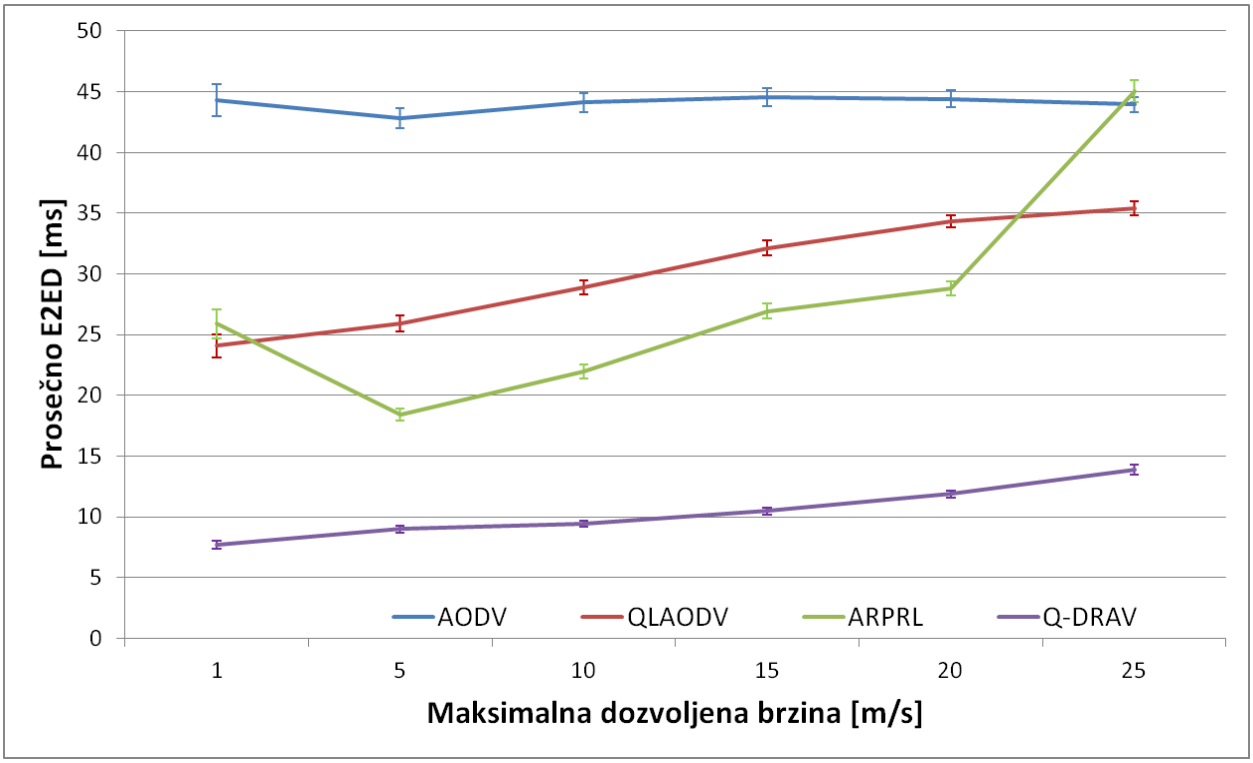
Slika 6.3. Zavisnost PLR od maksimalne dozvoljene brzine kretanja vozila u mreži



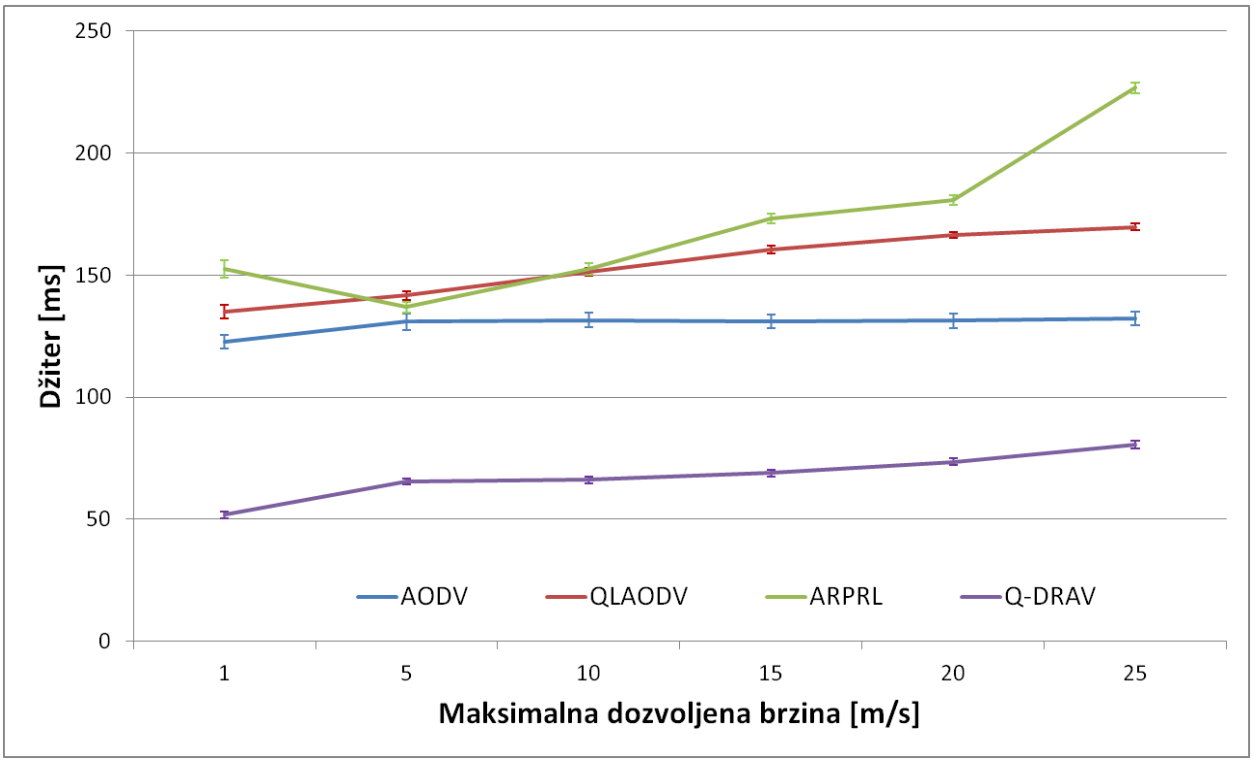
Slika 6.4. Zavisnost ostvarenog protoka od maksimalne dozvoljene brzine kretanja vozila u mreži

Slike 6.5 i 6.6 prikazuju zavisnost prosečnog E2ED i džitera od maksimalne dozvoljene brzine kretanja vozila u prvom scenariju. I ovde je primetno pogoršanje posmatranih parametara sa povećanjem maksimalne dozvoljene brzine, kod svih posmatranih protokola. Očigledno je da Q-DRAV pruža ubedljivo najbolje performanse po oba posmatrana parametra, a uzak interval poverenja i ovde potvrđuje visoku pouzdanost dobijenih rezultata.

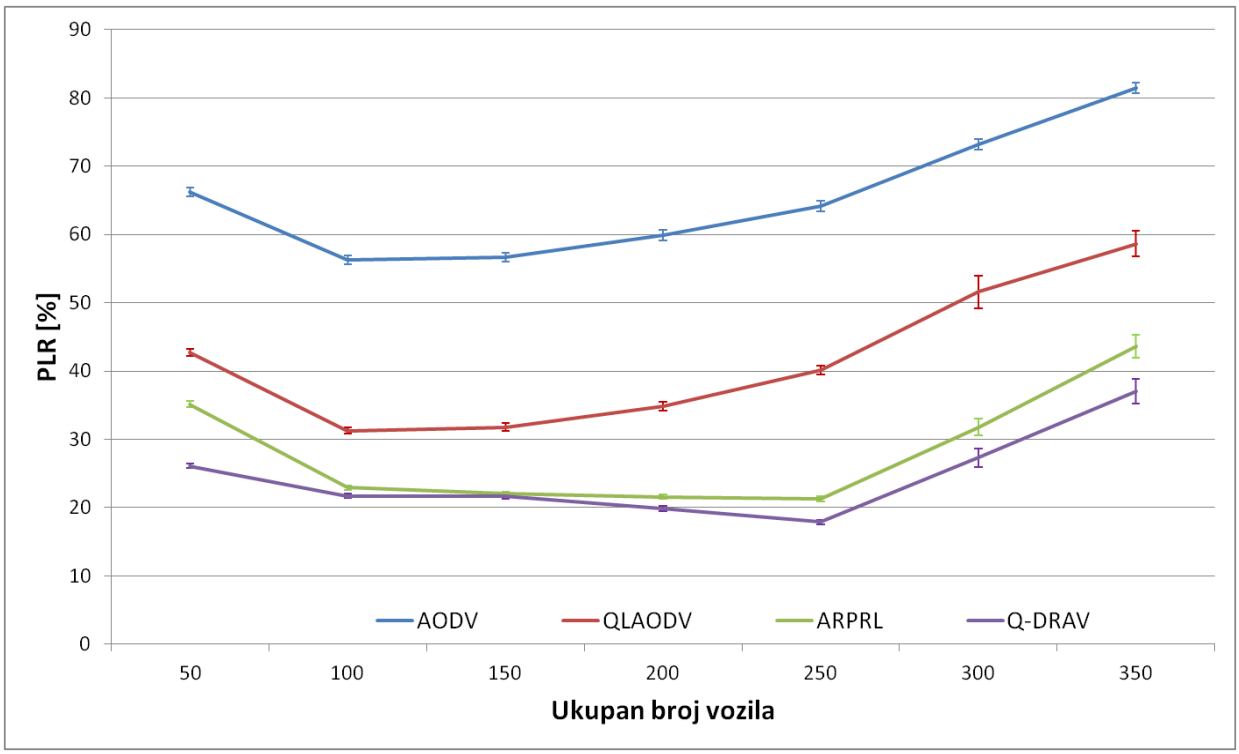
Može se videti da QLAODV i ARPRL imaju manje kašnjenje, ali veći džiter u poređenju sa AODV protokolom. Primetno je da i po ovim parametrima ARPRL pokazuje lošije rezultate pri krajnje malim i velikim maksimalnim dozvoljenim brzinama kretanja vozila.



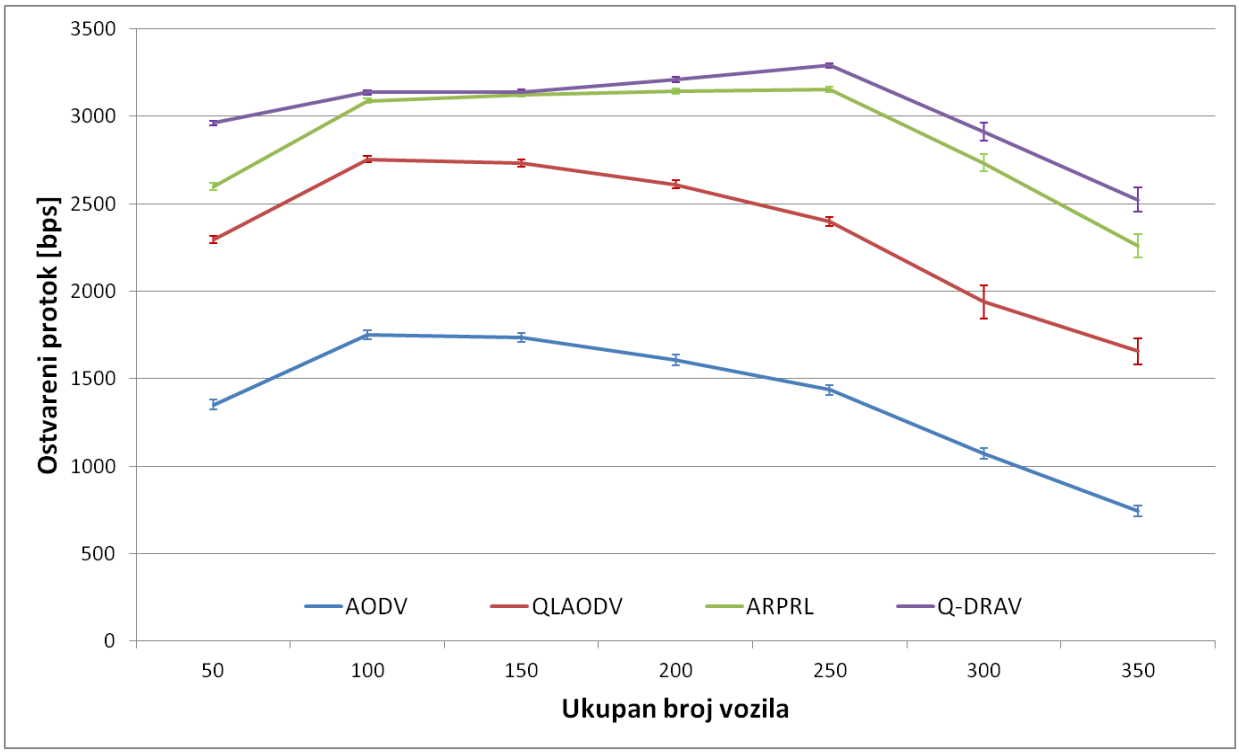
Slika 6.5. Zavisnost prosečnog E2ED od maksimalne dozvoljene brzine kretanja vozila u mreži



Slika 6.6. Zavisnost džitera od maksimalne dozvoljene brzine kretanja vozila u mreži



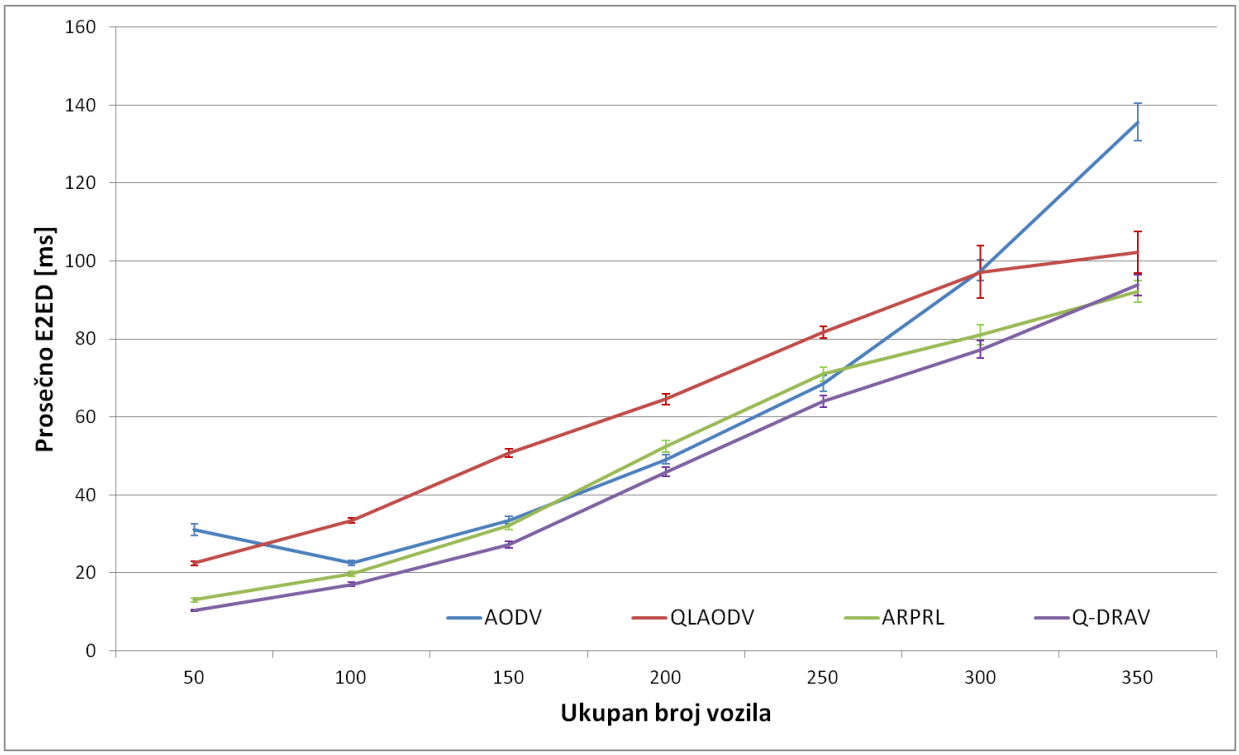
Slika 6.7. Zavisnost PLR od ukupnog broja vozila u mreži



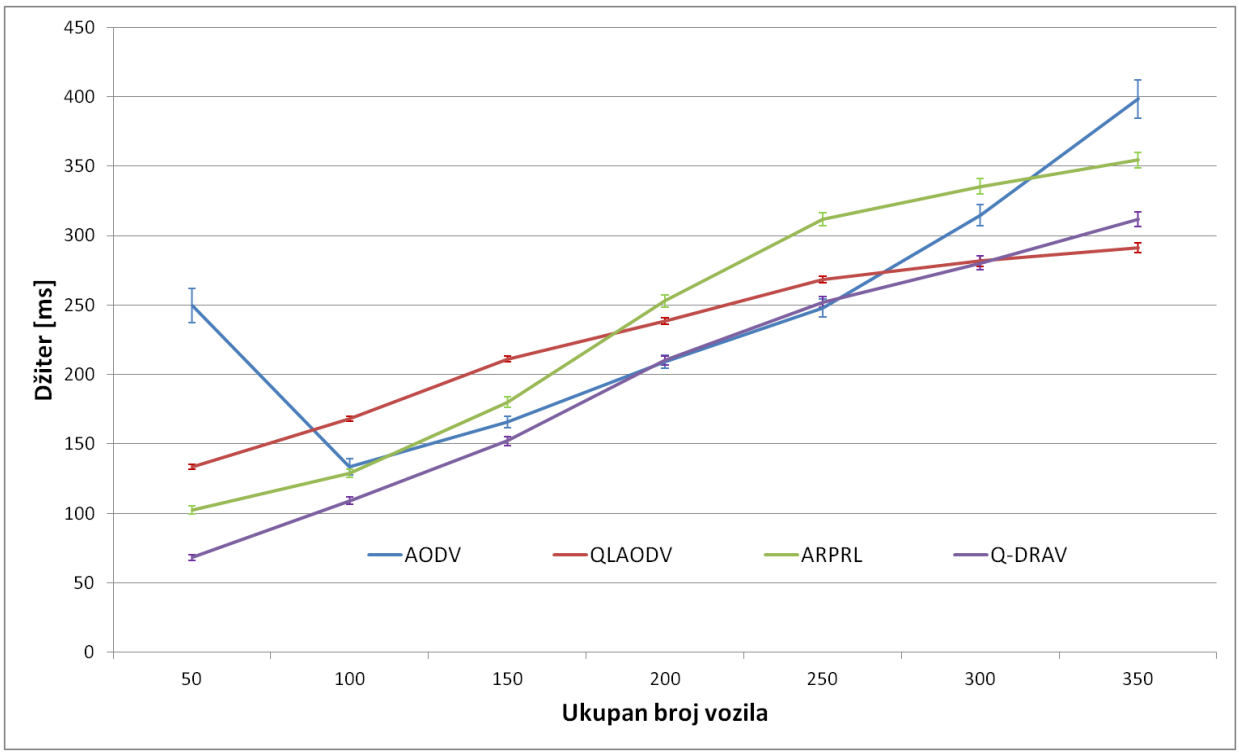
Slika 6.8. Zavisnost ostvarenog protoka od ukupnog broja vozila u mreži

Slike 6.7 i 6.8 prikazuju zavisnost PLR i ostvarenog protoka na aplikacionom sloju od ukupnog broja vozila u drugom scenariju. Kod svih protokola, ovi parametri se najpre popravljaju pri povećanju broja vozila u mreži (zbog bolje povezanosti mreže), a zatim degradiraju pri najvećim gustinama vozila (zbog preopterećenja u mreži). Primetno je da ARPRL daje značajno bolje

rezultate od AODV i QLAODV protokola, ali ga ipak Q-DRAV uvek nadmašuje, posebno u veoma retkim i veoma gustim mrežama.



Slika 6.9. Zavisnost prosečnog E2ED od ukupnog broja vozila u mreži



Slika 6.10. Zavisnost džitera od ukupnog broja vozila u mreži

Na kraju, slike 6.9 i 6.10 prikazuju zavisnost prosečnog E2ED i džitera od ukupnog broja vozila u mreži u drugom scenariju. Sa povećanjem broja vozila oba parametra rastu, zbog sve većeg broja paketa koji se šalju u mreži (što dovodi do većih zagušenja). Prema ovim parametrima, Q-DRAV

gotovo uvek pokazuje najbolje performanse, osim u slučaju veoma gustih mreža gde ARPRL daje gotovo identično kašnjenje, a QLAODV nešto manji džiter. QLAODV pokazuje konstantno veća kašnjenja od ARPRL i Q-DRAV protokola (čak i od AODV uglavnom), dok ARPRL sa druge strane daje veliki džiter u slučaju mreža sa većim brojem čvorova.

6.3. Analiza rezultata simulacija

Rezultati intenzivnih simulacija su pokazali da, uprkos značajnom unapređenju mrežnih performansi u poređenju sa AODV protokolom, postojeći protokoli rutiranja bazirani na RL pokazuju brojne nedostatke u VANET okruženju. QLAODV protokol je postavio dobre polazne osnove, koje su dalje unapređene ARPRL protokolom, ali još uvek postoji mnogo prostora za poboljšanje. Na osnovu ovih zapažanja predložen je novi protokol rutiranja (Q-DRAV) (Bugarčić i ostali, 2024), koji pokazuje veću prilagodljivost različitim mrežnim okruženjima i generalno pruža bolje performanse u VANET mrežama. Jedno od ograničenja studija u kojima su predloženi QLAODV (C. Wu i ostali, 2010) i ARPRL (J. Wu i ostali, 2018) je nedovoljno testiranje protokola. Stoga su u ovom poglavlju sprovedena dodatna testiranja (u malim i velikim mrežama, sa varijabilnim brojem vozila, sa varijabilnim maksimalnim dozvoljenim brzinama kretanja vozila, sa manjim i većim aplikacionim protocima), kako bi se pokazali nedostaci ranijih protokola i predložio pouzdan metod za njihovo unapređenje. Primarni cilj novog protokola je da reši ograničenja prethodnih protokola na što sveobuhvatniji način i poboljša mrežne performanse u različitim saobraćajnim uslovima (koji podrazumevaju V2V komunikaciju u gradskim sredinama).

Posmatrajući PLR i ostvareni protok na aplikacionom sloju u prvom scenariju, gde je testirana manja mreža i izvršena parametarska analiza variranjem maksimalnih dozvoljenih brzina kretanja vozila, postaje očigledno da postojeći protokoli bazirani na QL značajno nadmašuju AODV protokol. Međutim, ARPRL protokol pokazuje priličnu degradaciju posmatranih performansi pri najnižim i najvišim dozvoljenim brzinama kretanja vozila. Pošto ARPRL protokol prvobitno nije testiran u takvom okruženju, ovi nedostaci nisu primećeni u (J. Wu i ostali, 2018). Pretpostavka je da se ovaj problem javlja zato što Q-vrednosti, koje se koriste za određivanje optimalne putanje za slanje podataka, u velikoj meri zavise od brzine kretanja vozila. Pri vrlo niskim brzinama, ovaj uticaj nije dovoljno jak, što dovodi do spore konvergencije Q-vrednosti i uočljive degradacije mrežnih performansi. Pri visokim maksimalnim dozvoljenim brzinama u urbanom okruženju, vozila se između dva semafora/raskrsnice kreću veoma brzo i u relativno kratkom vremenu dolaze do sledećeg semafora/raskrsnice. Posledica toga je da vozila i u ovom slučaju veliki deo vremena provode u približno stacionarnom stanju (čekajući na semaforu/raskrsnici), što iz sličnih razloga kao kod niskih brzina rezultira degradacijom mrežnih performansi.

Q-DRAV protokol rešava ovaj problem uključivanjem drugih uticajnih faktora koji pomažu u izboru optimalne putanje, čak i kada su vozila stacionarna (NB, postojanje direktnog linka do odredišta, kreiranje petlji, LU, gubici paketa na MAC podsloju, RRB, HC i RTD), i dosledno pokazuje bolje rezultate od protokola sa kojima je upoređen za sve maksimalne dozvoljene brzine kretanja vozila. Konstantno uzak interval poverenja kod Q-DRAV protokola dodatno potvrđuje verodostojnost ovih rezultata.

Ovo poboljšanje mrežnih performansi još je očiglednije prilikom ispitivanja prosečnog E2ED i džitera, gde Q-DRAV pokazuje najbolje performanse za sve testirane maksimalne dozvoljene brzine kretanja vozila u prvom scenariju (takođe uz konstantno uzak interval poverenja). Posmatrajući samo prosečno E2ED, ubedljivo najlošije rezultate pokazuje AODV protokol. QLAODV konstantno daje značajno manja kašnjenja od AODV protokola, ali i značajno veća od Q-DRAV protokola. ARPRL i ovde pokazuje prilično povećanje kašnjenja za krajnje male i velike brzine (iz sličnih razloga kao kod PLR i ostvarenog protoka), dok u ostalim slučajevima daje značajno manja kašnjenja od AODV i QLAODV protokola. Ipak, ovi rezultati su i dalje daleko

lošiji od onih koji su postignuti primenom Q-DRAV protokola. Ako se posmatra samo džiter, QLAODV i ARPRL pokazuju konstantno lošije performanse čak i od AODV protokola. Posebnu degradaciju ovog parametra pokazuje ARPRL protokol, pri većim maksimalnim dozvoljenim brzinama kretanja vozila.

Veoma dobre performanse Q-DRAV protokola po ova dva parametra su, s jedne strane, posledica relativno brzog otkrivanja kraćih putanja za slanje paketa podataka, što je postignuto uzimanjem u obzir kašnjenja paketa i broja hopova duž cele putanje (putem RTD i HC parametara), dodeljivanjem maksimalnih Q-vrednosti direktnim putanjama do odredišta, kao i nagrađivanjem putanja koje imaju samo dva hopa do odredišta. S druge strane, izbegavanje kreiranja petlji, kroz kažnjavanje putanja na kojima se javljaju, značajno je uticalo na smanjenje kašnjenja. Ukoliko dođe do kreiranja petlje, paket kruži u njoj sve dok se ne pojavi putanja sa većom Q-vrednošću. U Q-DRAV protokolu to se veoma brzo rešava, zbog drastičnog kažnjavanja (smanjivanja) Q-vrednosti za putanje na kojima se pojavila petlja.

Posmatrajući PLR i ostvareni protok na aplikacionom sloju u drugom scenariju, koji uključuje veću mrežnu oblast i gde je izvršena parametarska analiza variranjem ukupnog broja vozila u mreži, može se zaključiti da ARPRL i Q-DRAV konzistentno pokazuju znatno bolje performanse od QLAODV i AODV protokola. Ipak, Q-DRAV konstantno nadmašuje ARPRL, posebno pri ekstremno niskim i ekstremno visokim gustinama vozila. Problem kod retkih mreža je nedovoljna povezanost, što otežava vozilima da pronađu putanju do odredišta, prouzrokujući gubitke i odbacivanje paketa podataka. S druge strane, u veoma gustim mrežama dolazi do preopterećenja mrežnih linkova, što rezultira povećanim zagušenjem mreže. QLAODV protokol generalno pokazuje veoma visoke gubitke i mali ostvareni protok paketa u drugom scenariju, posebno za veći broj čvorova, što ukazuje na to da ne uspeva da pronađe optimalne putanje u mrežama na većim oblastima i sa većim brojem vozila.

Q-DRAV smanjuje visoke gubitke paketa u mrežama sa manjim brojem vozila ubrzavanjem konvergencije kratkih putanja, kažnjavajući putanje na kojima dolazi do gubitka paketa na MAC podsloju, kao i kažnjavajući putanje preko kojih je poslat RPP paket, ali RPP-ACK paket nije stigao na vreme. Na ovaj način se smanjuje verovatnoća izbora visokorizičnih putanja, a samim tim se smanjuju gubici paketa i povećava ostvareni protok. Problem visokog opterećenja u mrežama sa velikim brojem vozila Q-DRAV prevazilazi uzimanjem u obzir zauzetosti propusnog opsega sledećeg hopa na putanji do odredišta (putem NB parametra) i zauzetosti propusnog opsega cele putanje (putem RRB parametra), kao i smanjenjem učestanosti slanja *Hello* paketa za visoko opterećene mreže.

U pogledu prosečnog E2ED i džitera u drugom scenariju, Q-DRAV daje najbolje rezultate za gotovo sve gustine mreže, osim u najgušćim mrežama gde ARPRL pokazuje gotovo identične rezultate u pogledu prosečnog E2ED, dok QLAODV pokazuje nešto bolje rezultate u pogledu džitera. Međutim, ova razlika nije značajna, posebno uzimajući u obzir da se prosečno kašnjenje i džiter računaju samo za pakete koji su uspešno stigli do odredišta. ARPRL protokol pri najvećim gustinama čvorova pokazuje vidno veći PLR od Q-DRAV protokola (blizu 20% veći gubici), dok QLAODV protokol pokazuje skoro dvostruko veće gubitke paketa u poređenju sa Q-DRAV protokolom. Takođe, QLAODV pokazuje konzistentno veliko kašnjanje, a ARPRL znatno veći džiter, posebno za gušće mreže, u poređenju sa Q-DRAV protokolom.

Ipak, treba naglasiti da je najveći prostor za unapređenje Q-DRAV protokola primetan upravo kod mreža sa velikim brojem čvorova. Performanse svih testiranih protokola značajno degradiraju kod ovih mreža, jer dolazi do prevelikog opterećenja mreže usled velikog broja kontrolnih paketa (i njihovih dimenzija) koji se koriste za ažuriranje Q-vrednosti. Može se reći da kod RL baziranih protokola veličina kontrolnih paketa i njihov broj eksponencijalno rastu sa porastom broja vozila u

mreži. Zato treba težiti daljem komprimovanju informacija koje se šalju putem ovih paketa, kao i uvođenju dodatnih tehnika za proređivanje slanja paketa u slučaju prevelikog opterećenja mreže. Kod Q-DRAV protokola ovo je izvršeno uvođenjem proređivanja slanja *Hello* paketa za guste mreže, preko koeficijenta verovatnoće slanja *p* (koji je manji ili jednak jedinici). Ovim je značajno smanjeno degradiranje mrežnih performasi u gustim mrežama, ali svakako postoji prostor za dalja unapređenja.

Može se zaključiti da predloženi Q-DRAV protokol dosledno pruža značajno bolje rezultate u oba razmatrana scenarija, čime se pokazao kao veoma kvalitetna zamena postojećim protokolima rutiranja u VANET mrežama. Dobri rezultati u različitim saobraćajnim uslovima pokazali su da Q-DRAV protokol ima perspektivu da značajno poboljša mrežne performanse u realnim VANET mrežama, koje su sklone stalnim promenama u gustini i brzini kretanja mrežnih čvorova. U budućnosti, protokol je moguće dalje razviti za V2V komunikaciju kroz uključivanje dubokih neuronskih mreža u RL proces, analizom različitih saobraćajnih scenarija (kao što su mreže vozila na pametnim autoputevima), uključivanjem drugih relevantnih mrežnih parametara koji bi korisno uticali na izbor optimalne putanje u RL procesu, kombinovanjem RL sa drugim tehnikama (SDN mreže, FL, BC, i druge), itd. Slični protokoli mogu se primeniti i u FANET mrežama, uz određena prilagođavanja uticajnih faktora karakteristikama ovih mreža.

7. ZAKLJUČAK

Razvoj inteligentnih transportnih sistema predstavlja jedan od glavnih pravaca unapređenja savremenih saobraćajnih mreža. Bitan segment ovih sistema mogu biti VANET i FANET mreže, tako da je od velikog interesa dalje unaprediti njihove performanse. Ovde se prvenstveno misli na smanjenje kašnjenja i procenta izgubljenih paketa u mrežama. Ove performanse u velikoj meri zavise od protokola rutiranja koji se koristi za izbor optimalne putanje za razmenu podataka između čvorova u mreži, tako da postoji potreba za njihovim stalnim unapređenjem.

Na početku disertacije predstavljeni su tradicionalni pristupi rutiranju podataka u WANET mrežama. Detaljnije su opisani principi funkcionisanja tradicionalnih AODV, DSR i DSDV protokola rutiranja. Ipak, pokazalo se da ovi protokoli ne pružaju zadovoljavajuće performanse kod visoko dinamičkih mreža (kao što su VANET i FANET), tako da ih je neophodno unaprediti korišćenjem nekih savremenih tehnika. Jedna od tih tehnika je mašinsko učenje, a posebno se ističe primena RL tehnike. U okviru ove disertacije pokazano je kako primenom RL tehnike u protokolima rutiranja mogu biti značajno unapređene ključne performanse dinamičkih WANET mreža. Zato je nakon tradicionalnih pristupa rutiranju, objašnjen i pristup rutiranju na bazi RL. Ovde su detaljnije opisani najznačajniji RL algoritmi i principi njihovog funkcionisanja. Kako bi se šire sagledali dosadašnji rezultati u ovoj oblasti, izvršen je pregled i klasifikacija aktuelnih RL baziranih protokola rutiranja za VANET i FANET mreže. Protokoli su klasifikovani u više kategorija na osnovu tipa mreže, tipa RL koji se primenjuje i kombinacije RL sa nekim drugim tehnikama. Takođe, izvršeno je i poređenje protokola rutiranja na osnovu uticajnih faktora koji određuju vrednost nagrade u RL procesu, posmatranih indikatora mrežnih performansi u postupku evaluacije protokola rutiranja, kao i korišćenog simulacionog alata za testiranje predloženih protokola. U svim radovima bilo je primetno značajno poboljšanje posmatranih mrežnih performansi, u poređenju sa performansama koje se postižu korišćenjem tradicionalnih protokola rutiranja. Ovo je pokazatelj efikasnosti primene RL tehnike, posebno u mrežama u kojima se topologija često menja i u kojima se čvorovi kreću velikom brzinom. Na osnovu izvršene klasifikacije i poređenja, za reprezentativne primere RL baziranih protokola, sa kojima je kasnije upoređen novi Q-DRAV protokol, izabrani su QLAODV i ARPRL protokoli. Oba protokola pokazuju značajno unapređenje mrežnih performansi u VANET mrežama, uz korišćenje relativno jednostavnog QL algoritma i bez zahteva za dodatnom mrežnom infrastrukturom.

Kako bi se dalje unapredio proces rutiranja u VANET mrežama, predložen je novi Q-DRAV protokol rutiranja za urbane VANET mreže, koji se bazira na primeni RL za izbor optimalne putanje za slanje podataka. Ovaj protokol uključuje brojne uticajne faktore u proces izbora optimalne putanje, a tu spadaju dostupnost propusnog opsega vozila, postojanje direktnog linka ka odredištu, pouzdanost linka, gubici paketa, rastojanje između vozila, dostupnost propusnog opsega celokupne putanje do odredišta, broj hopova do odredišta, kašnjenje paketa, potencijalno kreiranje petlji prilikom slanja podataka i gustina vozila u mreži. Protokol je implementiran i testiran u NS-3 simulacionom okruženju. Iz tog razloga su predstavljeni osnovni koncepti NS-3 simulatora i objašnjeni su elementi za modelovanje *ad hoc* mreža. Jedan od ključnih elemenata je upravo izbor odgovarajućeg protokola rutiranja.

Na osnovu intenzivnih simulacija u NS-3 simulatoru izvršena je analiza mrežnih performansi i poređenje novog Q-DRAV protokola sa rezultatima primene AODV, QLAODV i ARPRL protokola u dva VANET scenarija. Testiranje i poređenje protokola je najpre izvršeno na manjoj simulacionoj oblasti sa fiksnim brojem vozila i promenljivom maksimalnom dozvoljenom brzinom kretanja vozila. Nakon toga, protokoli su testirani i na većoj simulacionoj oblasti sa fiksnom maksimalnom dozvoljenom brzinom kretanja vozila, uz varijaciju ukupnog broja vozila u mreži. Na osnovu iscrpne simulacione analize, pokazalo se da predloženi protokol prevazilazi protokole sa kojima je poređen u pogledu ostvarenog protoka, PLR, prosečnog E2ED i džitera u oba simulaciona scenarija. Dobri rezultati u različitim saobraćajnim okolnostima pokazuju da Q-DRAV može značajno da unapredi mrežne performanse u realnim VANET mrežama, koje su sklone stalnim promenama gustine i brzine kretanja čvorova u mreži.

7.1. Doprinos disertacije

Osnovni cilj istraživanja u okviru ove disertacije je unapređenje performansi dinamičkih WANET mreža kroz adekvatno uključivanje RL u proces rutiranja paketa. Doprinosi disertacije potvrđuju ispunjenje postavljenog cilja. S tim u vezi, mogu se izdvojiti sledeći najvažniji rezultati i naučni doprinosi doktorske disertacije:

- izvršen je sveobuhvatan pregled i analiza aktuelnih istraživanja u oblasti primene RL tehnike za unapređenje protokola rutiranja u dinamičkim WANET mrežama;
- izvršena je klasifikacija aktuelnih protokola baziranih na RL za dinamičke WANET mreže, sa posebnim osvrtom na tip mreže, tip RL koji se koristi i primenu drugih tehnika u protokolima;
- izvršeno je poređenje aktuelnih protokola baziranih na RL za VANET i FANET mreže na osnovu uticajnih faktora u RL procesu, posmatranih parametara u postupku evaluacije protokola i korišćenih simulacionih alata;
- razvijen je novi protokol rutiranja za VANET mreže sa ciljem poboljšanja mrežnih performansi;
- novi protokol rutiranja i relevantni postojeći protokoli rutiranja bazirani na RL, sa kojima je novi protokol upoređen, implementirani su u NS-3 simulaciono okruženje;
- sprovedena je simulaciona analiza i evaluacija najvažnijih mrežnih performansi u slučaju primene novog i ranijih protokola rutiranja u VANET okruženju;
- izvršena je statistička obrada i komparativna analiza rezultata simulacija koja pokazuje da su unapređene ukupne mrežne performanse u pogledu ostvarenog protoka, PLR, E2ED i džitera.

Sveobuhvatan pregled, klasifikacija i poređenje aktuelnih istraživanja doprineli su boljem sagledavanju dosadašnje primene RL u protokolima rutiranja za dinamičke WANET mreže i omogućili su dobru osnovu za izbor pravca daljeg razvoja ovih protokola. Uzimajući u obzir prednosti i nedostatke prethodnih protokola, razvijen je novi protokol rutiranja koji je doprineo dodatnom unapređenju mrežnih performansi u VANET mrežama, u pogledu većeg ostvarenog aplikacionog protoka, kao i manjeg procenta izgubljenih paketa, kašnjenja paketa i džitera. Protokol je unapredio mrežne performanse za različite veličine mreža, promenljiv broj čvorova u mreži i varijabilno ograničenje brzine kretanja čvorova u mreži, što potvrđuje da može da doprinese unapređenju performansi realnih VANET mreža. Implementacioni kôd novog protokola javno je dostupan, tako da može biti iskorišćen za detaljnu analizu realizacije protokola i za jednostavno poređenje sa budućim protokolima u NS-3 simulatoru. Ovo bi moglo značajno olakšati i ubrzati nova istraživanja u ovoj oblasti.

7.2. Pravci budućih istraživanja

U okviru budućih istraživanja planiran je dalji razvoj protokola rutiranja za VANET mreže, što može podrazumevati primenu dubokih neuronskih mreža u RL proces, analizu drugačijih saobraćajnih scenarija (kao što su mreže vozila na pametnim autoputevima, uključivanje *building* modela u gradske mreže, veća simulaciona područja sa većim brojem vozila, itd.), uključivanje još nekih relevantnih parametara koji bi korisno uticali na izbor optimalne putanje, uključivanje spoljne infrastrukture (kao što su RSU jedinice) u proces rutiranja, itd. Slični protokoli mogu se primeniti i na druge tipove mreža, kao što su FANET mreže, čija popularnost u poslednje vreme sve više raste. Da bi dali dobre performanse, protokoli treba da se modifikuju tako da odgovore specifičnim zahtevima FANET mreža.

Zbog prednosti koje pružaju *ad hoc* mreže u odnosu na celularne mreže (na primer 5G) u brojnim okolnostima, po svemu sudeći VANET i FANET mreže će ostati važan deo inteligentnih transportnih sistema i u narednom periodu. Takođe, postoje i velike mogućnosti kombinovanja ovih mreža sa celularnim mrežama radi postizanja što bolje komunikacije između mobilnih korisnika (u koje se ubrajaju i vozila i bespilotne letelice). Samim tim i u budućem periodu postojaće velika protreba za razvojem novih protokola rutiranja za ove mreže, a RL se nameće kao jedan od njihovih najznačajnijih elemenata.

Literatura

Abbass H. A. (2001). Marriage in honey bees optimization (MBO): A haplometrosis polygynous swarming approach. In *Proceedings of the Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)*, Seoul, Korea, May 27-30, pp. 207-214. DOI: 10.1109/CEC.2001.934391

Ahmed A. A., Malebary S. J., Ali W., Barukab O. M. (2023). Smart traffic shaping based on distributed reinforcement learning for multimedia streaming over 5G-VANET communication technology. *Mathematics*, vol. 11(3): pp. 700-715. DOI: 10.3390/math11030700

An C., Wu C., Yoshinaga T., Chen X., Ji Y. (2018). A context-aware edge-based VANET communication scheme for ITS. *Sensors*, vol. 18(7). DOI: 10.3390/s18072022

Arafat M. Y., Moh S. (2021) A Q-learning-based topology-aware routing protocol for flying ad hoc networks. *IEEE Internet of Things Journal*, vol. 9(3): pp. 1985-2000. DOI: 10.1109/JIOT.2021.3089759

Ayub M. S., Adasme P., Melgarejo D. C., Rosa R. L., Rodríguez D. Z. (2022). Intelligent hello dissemination model for FANET routing protocols. *IEEE Access*, vol. 10: pp. 46513-46525. DOI: 10.1109/ACCESS.2022.3170066.

Bai F., Helmy A. (2004). *A survey of mobility models* in Book chapter in *Wireless Ad Hoc Networks*. University of Southern California, Los Angeles, California.

Bi X., Gao D., Yang M. (2020). A reinforcement learning-based routing protocol for clustered EV-VANET. In *Proceedings of the 2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, Chongqing, China, June 12-14, pp. 1769–1773. DOI: 10.1109/ITOEC49072.2020.9141805

Bugarčić P., Jevtić N., Malnar M. (2022). Reinforcement learning-based routing protocols in vehicular and flying ad hoc networks – a literature survey. *Promet-Traffic&Transportation*, vol. 34(6): pp. 893-906. DOI: 10.7307/ptt.v34i6.4159

Bugarčić P., Jevtić N., Malnar M., Stojanović M. (2024). Enhanced QL-based dynamic routing protocol for urban VANETs. *Advances in Electrical and Computer Engineering*, vol. 24(4): pp. 27-36. DOI: 10.4316/AECE.2024.04003

Clausen T. H., Jacquet P. (2003). *Optimized link state routing protocol (OLSR)*, RFC 3626.

Conner W. S., Kruys J., Kim K. J., Zuniga J. C. (2006). *Overview of the amendment for wireless local area mesh networking*, IEEE 802.11s Tutorial.

Cormen T. H., Leiserson C. E., Rivest R. L., Stein C. (2022). *Introduction to Algorithms, 3rd edition*. MIT press, Cambridge, Massachusetts

Da Costa L. A., Kunst R., De Freitas E. P. (2021). Q-FANET: Improved Q-learning based routing protocol for FANETs. *Computer Networks*, vol. 198. DOI: 10.1016/j.comnet.2021.108379

Dai C., Xiao X., Ding Y., Xiao L., Tang Y., Zhou S. (2018). Learning based security for VANET with blockchain. In *Proceedings of the 2018 IEEE International Conference on Communication*

Systems (ICCS), Chengdu, China, December 19-21., pp. 210–215. DOI: 10.1109/ICCS.2018.8689228

Das S. K., Manoj B. S., Murthy C. S. R. (2002). A dynamic core based multicast routing protocol for ad hoc wireless networks. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc '02)*, Lausanne, Switzerland, June 9-11, pp. 24-35. DOI: 10.1145/513800.513804

Daves R., Padhye J., Zill B. (2004). Comparison of routing metrics for static multi-hop wireless networks. *ACM SIGCOMM Computer Communication Review*, vol. 34(4): pp. 133-144. DOI: 10.1145/1030194.1015483

De Assis D. R., Wille E. C. G., Alves J. J. (2023). New results on the IC_AOMDV protocol for vehicular ad hoc networks in urban areas. *Advances in Electrical & Computer Engineering*, vol. 23(3): pp. 21-28. DOI:10.4316/AECE.2023.03003

Dube R., Rais C. D., Wang K. Y., Tripathi S. K. (1997). Signal stability-based adaptive routing (SSA) for ad hoc mobile networks. *IEEE Personal Communications*, vol. 4(1): pp. 36-45. DOI: 10.1109/98.575990

Garcia-Luna-Aceves J. J., Spohn M. (1999). Source-tree routing in wireless networks. In *Proceedings of the Seventh International Conference on Network Protocols*, Toronto, Canada, October 31-November 3, pp. 273-282. DOI: 10.1109/ICNP.1999.801950

GitHub (2024), [Online]. Dostupno na: https://github.com/pavlebugarcic/qdrav

Gunes M., Sorges U., Bouazizi I. (2002). ARA-the ant-colony based routing algorithm for MANETs. In *Procedings of the International Conference on Parallel Processing Workshop*, Vancouver, Canada, August 21, pp: 79-85. DOI: 10.1109/ICPPW.2002.1039715

Gupta S. G., Ghonge M. M., Thakare P. D., Jawandhiya P. M. (2013). Open-source network simulation tools: An overview. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, vol. 2(4): pp. 1629-1635.

Hass Z. J., Pearlman M. R. (1998). *The zone routing protocol (ZRP) for ad hoc networks (Internet-Draft*), Mobile Ad-hoc Network (MANET) Working Group, IETF.

He C., Liu S., Han S. (2020). A fuzzy logic reinforcement learning-based routing algorithm for flying ad hoc networks. In *Proceedings of the 2020 International Conference on Computing, Networking and Communications (ICNC)*, Big Island, USA, February 17-20, pp. 987-991. DOI: 10.1109/ICNC47757.2020.9049705

Hosseinzadeh M., Ali S., Ionescu-Feleaga L., Ionescu B. S., Yousefpoor M. S., Yousefpoor E., Ahmed O. H., Rahmani A. M., Mehmood A. (2023). A novel Q-learning-based routing scheme using an intelligent filtering algorithm for flying ad hoc networks (FANETs). *Journal of King Saud University - Computer and Information Sciences*, vol. 35(10). DOI: 10.1016/j.jksuci.2023.101817

Iwata A., Chiang C. C., Pei G., Gerla M., Chen T. W. (1999). Scalable routing strategies for ad hoc wireless networks. *IEEE Journal on Selected Areas in Communications*, vol. 17(8): pp. 1369-1379. DOI: 10.1109/49.779920

Jafarzadeh O., Dehghan M., Sargolzaey H., Esnaashari M. M. (2022). A model based reinforcement learning protocol for routing in vehicular ad hoc network. *Wireless Personal Communications*, vol. 123(1): pp. 975–1001. DOI: 10.1007/s11277-021-09166-9

Jevtić N., Bugarčić P. (2022). Analiza protokola rutiranja baziranih na učenju potkrepljivanjem za VANET mreže. In *Procedings of the XL Simpozijum o novim tehnologijama u poštanskom i telekomunikacionom saobraćaju – PosTel 2022*, Belgrade, Serbia, November 29-30, pp. 375-384. DOI: 10.37528/FTTE/9788673954165/POSTEL.2022.039

Jevtić N., Bugarčić P. (2023). Comparison of traditional and reinforcement learning based routing protocols in VANET scenario. *International Journal for Traffic & Transport Engineering*, vol. 13(1): pp. 125-137. DOI: 10.7708/ijtte2023.13(1).10

Jevtić N., Malnar M., Bugarčić P. (2021). Primena učenja potkrepljivanjem u protokolima rutiranja za dinamičke bežične *ad hoc* mreže. In *Procedings of the XXXIX Simpozijum o novim tehnologijama u poštanskom i telekomunikacionom saobraćaju – PosTel 2021*, Belgrade, Serbia, November 30-December 1, pp. 249-258. DOI: 10.37528/FTTE/9788673954455/POSTEL.2021.026

Jevtić N., Malnar M., Bugarčić P. (2023). Unapređenje protokola rutiranja za VANET mreže korišćenjem mašinskog učenja. In *Procedings of the XLI Simpozijum o novim tehnologijama u poštanskom i telekomunikacionom saobraćaju – PosTel 2023*, Belgrade, Serbia, November 28-29, pp. 191-200. DOI: 10.37528/FTTE/9788673954752/POSTEL.2023.019

Ji X., Xu W., Zhang C., Yun T., Zhang G., Wang X. (2019). Keep forwarding path freshest in VANET via applying reinforcement learning. In *Proceedings of the 2019 IEEE First International Workshop on Network Meets Intelligent Computations (NMIC)*, Dallas, USA, July 7-9, pp. 13-18. DOI: 10.1109/NMIC.2019.00008

Jiang S., Huang Z., Ji Y. (2021). Adaptive UAV-assisted geographic routing with Q-learning in VANET. *IEEE Communications Letters*, vol. 25(4): pp. 1358-1362. DOI: 10.1109/LCOMM.2020.3048250

Jiang Y., Zhu J., Yang K. (2023). Environment-aware adaptive reinforcement learning-based routing for vehicular ad hoc networks. *Sensors*, vol. 24(1): pp. 40-70. DOI: 10.3390/s24010040

Johnson D., Hu Y., Maltz D. (2007). *The dynamic source routing protocol (DSR) for mobile ad hoc networks for IPv4*, RFC 4728.

Khan A. S., Balan K., Javed Y., Tarmizi S., Abdullah J. (2019). Secure trust-based blockchain architecture to prevent attacks in VANET. *Sensors*, vol. 19(22). DOI: 10.3390/s19224954

Khan M. F., Yau K. L. A. (2020). Route selection in 5G-based flying ad-hoc networks using reinforcement learning. In *Proceedings of the 2020 10th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*, Penang, Malaysia, August 21-22, pp. 23-28. DOI: 10.1109/ICCSCE50387.2020.9204944

L'Ecuyer P., Simard R., Chen E. J., Kelton W. D. (2001). An object-oriented random number package with many long streams and substreams. *Operations Research*, vol. 50(6): pp. 1073-1075. DOI: 10.1287/opre.50.6.1073.358

Lacage M., Henderson T. (2006). Yet Another Network Simulator. In *Proceding of the 2006 Workshop on ns-3*, Pisa, Italy, October 10, pp. 12-21. DOI: 10.1145/1190455.1190467

Layuan L., Chunlin L. (2007). A QoS multicast routing protocol for clustering mobile ad hoc networks. *Computer Communications*, vol. 30(7): pp. 1641-1654. DOI: 10.1016/j.comcom.2007.01.017

Li F., Song X., Chen H., Li X., Wang Y. (2019). Hierarchical routing for vehicular ad hoc networks via reinforcement learning. *IEEE Transactions on Vehicular Technology*, vol. 68(2): pp. 1852-1865. DOI: 10.1109/TVT.2018.2887282

Li G., Gong C., Zhao L., Wu J., Boukhatem L. (2020). An efficient reinforcement learning based charging data delivery scheme in VANET-enhanced smart grid. In *Proceedings of the 2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*, Busan, South Korea, February 19-22, pp. 263-270. DOI: 10.1109/BigComp48618.2020.00-64

Li J., Chen M. (2020). QMPS: Q-learning based message prioritizing and scheduling algorithm for flying ad hoc networks. In *Proceedings of the 2020 International Conference on Networking and Network Applications (NaNA)*, Haikou City, China, December 10-13, pp. 265-270. DOI: 10.1109/NaNA51271.2020.00052

Liu B., Xu G., Xu G., Wang C., Zuo P. (2023). Deep reinforcement learning-based intelligent security forwarding strategy for VANET. *Sensors*, vol. 23(3): pp. 1204-1218. DOI: 10.3390/s23031204

Liu J., Wang Q., He C., Hu Y. (2020). ARdeep: Adaptive and reliable routing protocol for mobile robotic networks with deep reinforcement learning**.** In *Proceedings of the 2020 IEEE 45th Conference on Local Computer Networks (LCN)*, Sydney, Australia, November 16-19, pp. 465-468. DOI: 10.1109/LCN48667.2020.9314848

Liu J., Wang Q., He C., Jaffres-Runser K., Xu Y., Li Z., Xu Y. (2020). QMR: Q-learning based multi-objective optimization routing protocol for flying ad hoc networks. *Computer Communications*, vol. 150: pp. 304–316. DOI: 10.1016/j.comcom.2019.11.011

Liu X., Amour B. S., Jaekel A. (2023). A Reinforcement learning-based congestion control approach for V2V communication in VANET. *Applied Sciences*, vol. 13(6). DOI: 10.3390/app13063640

Lolai A., Wang X., Hawbani A., Dharejo A. F., Qureshi T., Farooq M. U., Mujahid M., Babar A. H. (2022). Reinforcement learning based on routing with infrastructure nodes for data dissemination in vehicular networks. *Wireless Networks*, vol. 28: pp. 2169-2184. DOI: 10.1007/s11276-022-02926-w

Luo L., Sheng L., Yu H., Sun G. (2022). Intersection-based V2X routing via reinforcement learning in vehicular ad hoc networks. *IEEE Transactions on Intelligent Transportation Systems*, vol. 23(6): pp. 5446-5459. DOI: 10.1109/TITS.2021.3053958

Malnar M., Jevtic N. (2022). An improvement of AODV protocol for the overhead reduction in scalable dynamic wireless ad hoc networks. *Wireless Networks*, vol. 28(3): pp. 1039-1051. DOI: 10.1007/s11276-022-02890-5

Marina M. K., Das S. R. (2001). On-demand multi path distance vector routing in ad hoc notworks. In *Proceedings of the Ninth International Conference on Network Protocols (ICNP)*, Riverside, USA, November 11-14, pp. 14-23. DOI: 10.1109/ICNP.2001.992756

Mohta A., Ajankar S. I., Chandane M. M. (2010). Network Simulator-3: A Review. *International Journal of IT & Knowledge Management*, vol. 3(1).

Mowla N. I., Tran N. H., Doh I., Chae K. (2020). AFRL: Adaptive federated reinforcement learning for intelligent jamming defense in FANET. *Journal of Communications and Networks*, vol. 22(3): pp. 244-258. DOI: 10.1109/JCN.2020.000015

Mubarek F., Aliesawi S., Ali Alheeti K., Alfahad N. (2018). Urban-AODV: An improved AODV protocol for vehicular ad-hoc networks in urban environment. *International Journal of Engineering & Technology*, vol. 7(4): pp. 3030-3036. DOI: 10.14419/ijet.v7i4.15031

Murthy S., Garcia-Luna-Aceves J. J. (1996). An efficient routing protocol for wireless networks. *Mobile Networks and Applications*, vol. 1(2): pp. 183-197. DOI: 10.1007/BF01193336

Nahar A., Das D. (2020). Adaptive reinforcement routing in software defined vehicular networks. In *Proceedings of the 2020 International Wireless Communications and Mobile Computing (IWCMC)*, Limassol, Cyprus, June 15-19, pp. 2118–2123. DOI: 10.1109/IWCMC48107.2020.9148237

Nahar A., Das D. (2020). SeScR: SDN-enabled spectral clustering-based optimized routing using deep learning in VANET environment. In *Proceedings of the 2020 IEEE 19th International Symposium on Network Computing and Applications (NCA)*, Cambridge, USA, November 24-27, pp. 1-9. DOI: 10.1109/NCA51143.2020.9306690

Navidi W., Camp T. (2004). Stationary distributions for the random waypoint mobility model. *IEEE Transactions on Mobile Computing*, vol. 3(1): pp. 99-108. DOI: 10.1109/TMC.2004.1261820

NS-3 (2025), [Online]. Dostupno na: https://www.nsnam.org/

Ogier R., Templin F., Lewis M. (2004). *Topology dissemination based on reverse path forwarding (TBRPF)*, RFC 3684.

Pei G., Gerla M., Chen T. W. (2000). Fisheye state routing: a routing scheme for ad hoc wireless networks. In *Proceedings of the IEEE International Conference on Communications (ICC 2000)*, New Orleans, USA, June 18-22, pp. 70-74. DOI: 10.1109/ICC.2000.853066

Perkins C. E., Bhagwat P. (1994). Highly dynamic destination sequenced distance vector routing (DSDV) for mobile computers. *ACM SIGCOMM Computer Communication Review*, vol. 24(4): pp. 234-244. DOI: 10.1145/190809.190336

Perkins C., Belding–Royer E., Das S. (2003). *Ad hoc on demand distance vector (AODV) routing*, RFC 3561.

Rajagopalan S., Shen C. C. (2006). ANSI: A swarm intelligence-based unicast routing protocol for hybrid ad hoc networks. *Journal of Systems Architecture*, vol. 52(8-9): pp. 485-504. DOI: 10.1016/j.sysarc.2006.02.006

Reddy L. R., Raghavan S. V. (2007). SMORT: Scalable multipath on-demand routing for mobile ad hoc networks. *Ad Hoc Networks*, vol. 5(2): pp. 162-188. DOI: 10.1016/j.adhoc.2005.10.002

Reddy T. B., Sriram S., Manoj B. S., Murthy C. S. R. (2006). MuSeQoR: Multi-path failure-tolerant security-aware QoS routing in ad hoc wireless networks. *Computer Networks*, vol. 50(9): pp. 1349-1383. DOI: 10.1016/j.comnet.2005.05.035

Ribeiro A., Sofia R. C. (2011). *A survey on mobility models for wireless networks*, SITI Technical Report SITI-TR-11-01.

Rodriguez-Bocca P. (2008). *Quality-centric design of peer-to-peer systems for live-video broadcasting*, PhD theses, Facultad de Ingenieria, Universidad de la Republica, Rennes, France.

Roh B. S., Han M. H., Ham J. H., Kim K. I. (2020). Q-LBR: Q-learning based load balancing routing for UAV-assisted VANET. *Sensors*, vol. 20(19): pp. 1-17. DOI: 10.3390/s20195685

Rui L., Yan Z., Tan Z., Gao Z., Yang Y., Chen X., Liu H. (2023). An intersection-based QoS routing for vehicular ad hoc networks with reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems***,** vol. 24(9): pp. 9068-9083. DOI: 10.1109/TITS.2023.3271456

Saini T. K., Sharma S. C. (2020). Recent advancements, review analysis, and extensions of the AODV with the illustration of the applied concept. *Ad hoc Networks*, vol. 103: pp. 102-148. DOI: 10.1016/j.adhoc.2020.102148

Saravanan M., Ganeshkumar P. (2020). Routing using reinforcement learning in vehicular ad hoc networks. *Computational Intelligence*, vol. 36(2): pp. 682–697. DOI: 10.1111/coin.12261

Sarker O., Shen H., Babar M. A. (2023). Reinforcement learning based neighbour selection for VANET with adaptive trust management. In *Proceedings of the 2023 IEEE 22nd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, Exeter, United Kingdom, November 1-3, pp. 585-594. DOI: 10.1109/TrustCom60117.2023.00091

Sivakumar R., Sinha P., Bharghavan V. (1999). CEDAR: A core-extraction distributed ad hoc routing algorithm. *IEEE Journal on Selected Areas in Communications*, vol. 17(8): pp. 1454-1465. DOI: 10.1109/49.779926

Sliwa B., Schuler C., Patchou M., Wietfeld C. (2021). PARRoT: Predictive ad-hoc routing fueled by reinforcement learning and trajectory knowledge. In *Proceedings of the 2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, Helsinki, Finland, April 25-28, pp. 1-7. DOI: 10.1109/VTC2021-Spring51267.2021.9448959

Smida E. B., Fantar S. G., Youssef H. (2020). Link efficiency and quality of experience aware routing protocol to improve video streaming in urban VANETs. *International Journal of Communication Systems*, vol. 33(3): e4209. DOI: 10.1002/dac.4209

Song B., Xu L., Wang P., Qiu X., Ke Y., Gu J., Yang Y., Chen A., Zhang F. (2024). DRL-AdCAR: Adaptive coding-aware routing with maximum coding opportunities and high-quality via deep reinforcement learning in FANET. *IEEE Transactions on Vehicular Technology (Early Access)*. DOI: 10.1109/TVT.2024.3461161

SUMO (2025), [Online]. Dostupno na: https://eclipse.dev/sumo/

Sutton R., Barto A. (2018). *Reinforcement learning: An introduction, second edition.* MIT Press, Cambridge, Massachusetts.

Toh C. K. (1997). Associativity-based routing for ad-hoc mobile networks. *Wireless Personal Communications*, vol. 4(2): pp. 103-139. DOI: 10.1023/A:1008812928561

Upadhyay P., Marriboina V., Goyal S. J., Kumar S., El-Kenawy E. S. M., Ibrahim A., Alhussan A. A., Khafaga D. S. (2023). An improved deep reinforcement learning routing technique for collision-free VANET. *Scientific Reports*, vol. 13(1). DOI: 10.1038/s41598-023-48956-y

Vaishampayan R., Garcia-Luna-Aceves J. J. (2004). Efficient and robust multicast routing in mobile ad hoc networks. In *Proceedings of the 2004 IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE Cat. No.04EX975)*, Fort Lauderdale, USA, October 25-27, pp. 304-313. DOI: 10.1109/MAHSS.2004.1392169

Wang J., Osagie E., Thulasiraman P., Thulasiram R. (2009). HOPNET: A hybrid ant colony optimization routing algorithm for mobile ad hoc network. *Ad Hoc Networks*, vol. 7(4): pp. 690-705. DOI: 10.1016/j.adhoc.2008.06.001

Wang Y., Giruka V., Singhal M. (2008). Truthful multipath routing for ad hoc networks with selfish nodes. *Journal of Parallel and Distributed Computing*, vol. 68(6): pp. 778-789. DOI: 10.1016/j.jpdc.2008.01.007

Wu C., Kumekawa K., Kato T. (2010). Distributed reinforcement learning approach for vehicular ad hoc networks. *IEICE Transactions on Communications*, vol. 93(6): pp. 1431-1442. DOI: 10.1587/transcom.E93.B.1431

Wu C., Yoshinaga T., Bayar D., Ji Y. (2019). Learning for adaptive anycast in vehicular delay tolerant networks. *Journal of Ambient Intelligence and Humanized Computing*, vol. 10: pp. 1379-1388. DOI: 10.1007/s12652-018-0819-y

Wu C., Yoshinaga T., Ji Y., Zhang Y. (2018). Computational intelligence inspired data delivery for vehicle-to-roadside communications. *IEEE Transactions on Vehicular Technology*, vol. 67(12): pp. 12038-12048. DOI: 10.1109/TVT.2018.2871606

Wu J., Fang M., Li H., Li X. (2020). RSU-assisted traffic-aware routing based on reinforcement learning for urban VANETs. *IEEE Access*, vol. 8: pp. 5733-5748. DOI: 10.1109/ACCESS.2020.2963850

Wu J., Fang M., Li X. (2018). Reinforcement learning based mobility adaptive routing for vehicular ad-hoc networks. *Wireless Personal Communications*, vol. 101: pp. 2143-2171. DOI: 10.1007/s11277-018-5809-z

Xie J., Talpade R. R., McAuley A., Liu M. (2002). AMRoute: Ad hoc multicast routing protocol. *Mobile Networks and Applications*, vol. 7(6): pp. 429-439. DOI: 10.1023/A:1020748431138

Xu K., Hong X., Gerla M. (2003). Landmark routing in ad hoc networks with mobile backbones. *Journal of Parallel and Distributed Computing*, vol. 63(2): pp. 110-122. DOI: 10.1016/S0743-7315(02)00058-8

Yang Q., Jang S., Yoo S. (2020). Q-learning-based fuzzy logic for multi-objective routing algorithm in flying ad hoc networks. *Wireless Personal Communications*, vol. 113(1): pp. 115-138. DOI: 10.1007/s11277-020-07181-w

Yang Q., Yoo S. (2024). Hierarchical reinforcement learning-based routing algorithm with grouped RSU in urban VANETs. *IEEE Transactions on Intelligent Transportation Systems*, vol. 25(8): pp. 10131-10146. DOI: 10.1109/TITS.2024.3353258

Yang X., Zhang W., Lu H., Zhao L. (2020). V2V routing in VANET based on heuristic Q-learning. *International Journal of Computers, Communications and Control*, vol. 15(5): pp. 1-17. DOI: 10.15837/ijccc.2020.5.3928

Yang Y., Zhao R., Wei X. (2019). Research on data distribution for VANET based on deep reinforcement learning. In *Proceedings of the 2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM)*, Dublin, Ireland, October 16-18, pp. 484-487. DOI: 10.1109/AIAM48774.2019.00102

Ye S., Xu L., Li X. (2021). Vehicle-mounted self-organizing network routing algorithm based on deep reinforcement learning. *Wireless Communications and Mobile Computing*, vol. 2021(1). DOI: 10.1155/2021/9934585

Zhang D., Yu F. R., Yang R. (2018). A machine learning approach for software-defined vehicular ad hoc networks with trust management. In *Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM)*, Abu Dhabi, United Arab Emirates, December 9-13, pp. 1-6. DOI: 10.1109/GLOCOM.2018.8647426

Zhang D., Yu F. R., Yang R. (2019). Blockchain-based distributed software-defined vehicular networks: A dueling deep Q-learning approach. *IEEE Transactions on Cognitive Communications and Networking*, vol. 5(4): pp. 1086–1100. DOI: 10.1109/TCCN.2019.2944399

Zhang D., Yu F. R., Yang R., Tang H. (2018). A deep reinforcement learning-based trust management scheme for software-defined vehicular networks. In *Proceedings of the 8th ACM Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications (DIVANet'18)*, Montreal, Canada, October 28 - November 2, pp. 1-7. DOI: 10.1145/3272036.3272037

Zhang D., Yu F. R., Yang R., Zhu L. (2020). Software-defined vehicular networks with trust management: A deep reinforcement learning approach. *IEEE Transactions on Intelligent Transportation Systems*, vol. 23(2): pp. 1400-1414. DOI: 10.1109/TITS.2020.3025684

Zhang D., Zhang T., Liu X. (2019). Novel self-adaptive routing service algorithm for application in VANET. *Applied Intelligence*, vol. 49(5): pp. 1866-1879. DOI: 10.1007/s10489-018-1368-y

Zhang W., Yang X., Song Q., Zhao L. (2021). V2V routing in VANET based on fuzzy logic and reinforcement learning. *International Journal of Computers, Communications & Control*, vol. 16(1): pp 1-19. DOI: 10.15837/ijccc.2021.1.4123

Zheng Z., Sangaiah A. K., Wang T. (2018). Adaptive communication protocols in flying ad hoc network. *IEEE Communications Magazine*, vol. 56(1): pp. 136-142. DOI: 10.1109/MCOM.2017.1700323

BIOGRAFIJA AUTORA

Pavle Bugarčić, mast. inž. saobraćaja, rođen je 17.10.1994. godine u Čačku. Nakon završene osnovne škole u Guči, upisao je prirodno-matematički smer Gimnazije u Čačku, koju je završio 2013. godine. Iste godine upisao je Saobraćajni fakultet Univerziteta u Beogradu. Diplomirao je u septembru 2017. godine na modulu za Telekomunikacioni saobraćaj i mreže, sa prosečnom ocenom 9,69. Master akademske studije na Saobraćajnom fakultetu u Beogradu, na Modulu za telekomunikacioni saobraćaj i mreže, upisao je školske 2017/18. godine i završio iste školske godine sa maksimalnom prosečnom ocenom 10,00. U oktobru 2018. godine upisao je doktorske akademske studije na studijskom programu "Saobraćaj" Saobraćajnog fakulteta Univerziteta u Beogradu i položio sve ispite predviđene nastavnim planom sa prosečnom ocenom 10,00. Tokom studija, na osnovu izuzetnog uspeha, bio je dobitnik brojnih stipendija, između ostalih i dve stipendije "Dositeja". Takođe, za školsku 2016/17. godinu, dobio je nagradu Saobraćajnog fakulteta za uspeh u završnoj (četvrtoj) godini studija, tokom koje je ostvario prosek 10,00.

Od maja 2018. godine zaposlen je na Saobraćajnom fakultetu Univerziteta u Beogradu, najpre kao saradnik u nastavi, a zatim kao asistent na Katedri za telekomunikacioni saobraćaj i mreže. U okviru svog zaposlenja održava računske i laboratorijske vežbe iz sledećih predmeta na osnovnim akademskim studijama: Osnovi elektronike, Telekomunikaciona elektronika, Telekomunikaciona merenja, Slučajni procesi u telekomunikacijama, Eksploatacija komunikacionih sistema, Prognoziranje u komunikacionom saobraćaju i Planiranje i prognoziranje u telekomunikacijama, a u jednom periodu bio je angažovan i na predmetima: Statistička teorija telekomunikacija, Osnovi telekomunikacionih sistema i Elementi razvoja informacionog društva. Na master akademskim studijama angažovan je na predmetu Prognoziranje novih servisa. U svom dosadašnjem radu učestvovao je u realizaciji jednog naučnog projekta. Bio je član brojnih komisija za odbranu završnih radova. Autor je jednog pomoćnog udžbenika, dvanaest radova na domaćim i međunarodnim konferencijama i četiri rada u domaćim i međunarodnim časopisima. Član je organizacionog odbora simpozijuma Postel. Uža oblast naučno-stručnog interesovanja mu je optimizacija rutiranja saobraćaja u bežičnim komunikacionim mrežama primenom veštačke inteligencije.

Izjava o autorstvu

Ime i prezime autora: Pavle Bugarčić

Broj indeksa: DS18D013

Izjavljujem

da je doktorska disertacija pod naslovom:

UNAPREĐENJE PROTOKOLA RUTIRANJA ZA DINAMIČKE BEŽIČNE *AD HOC* MREŽE KORIŠĆENJEM MAŠINSKOG UČENJA

- rezultat sopstvenog istraživačkog rada;
- da disertacija u celini ni u delovima nije bila predložena za sticanje druge diplome prema studijskim programima drugih visokoškolskih ustanova;
- da su rezultati korektno navedeni i
- da nisam kršio autorska prava i koristio intelektualnu svojinu drugih lica.

Potpis autora

U Beogradu, _____

_____

Izjava o istovetnosti štampane i elektronske verzije doktorskog rada

Ime i prezime autora: Pavle Bugarčić

Broj indeksa: DS18D013

Studijski program: Saobraćaj

Naslov rada: **UNAPREĐENJE PROTOKOLA RUTIRANJA ZA DINAMIČKE BEŽIČNE *AD HOC* MREŽE KORIŠĆENJEM MAŠINSKOG UČENJA**

Mentori: dr Marija Malnar, redovni profesor,

dr Nenad Jevrić, vanredni profesor

Izjavljujem da je štampana verzija mog doktorskog rada istovetna elektronskoj verziji koju sam predao radi pohranjivanja u **Digitalnom repozitorijumu Univerziteta u Beogradu**.

Dozvoljavam da se objave moji lični podaci vezani za dobijanje akademskog naziva doktora nauka, kao što su ime i prezime, godina i mesto rođenja i datum odbrane rada.

Ovi lični podaci mogu se objaviti na mrežnim stranicama digitalne biblioteke, u elektronskom katalogu i u publikacijama Univerziteta u Beogradu.

Potpis autora

U Beogradu, _____

_____

Izjava o korišćenju

Ovlašćujem Univerzitetsku biblioteku „Svetozar Marković“ da u Digitalni repozitorijum Univerziteta u Beogradu unese moju doktorsku disertaciju pod naslovom:

UNAPREĐENJE PROTOKOLA RUTIRANJA ZA DINAMIČKE BEŽIČNE *AD HOC* MREŽE KORIŠĆENJEM MAŠINSKOG UČENJA

koja je moje autorsko delo.

Disertaciju sa svim prilozima predao sam u elektronskom formatu pogodnom za trajno arhiviranje.

Moju doktorsku disertaciju pohranjenu u Digitalnom repozitorijumu Univerziteta u Beogradu i dostupnu u otvorenom pristupu mogu da koriste svi koji poštuju odredbe sadržane u odabranom tipu licence Kreativne zajednice (Creative Commons) za koju sam se odlučio.

1. Autorstvo (CC BY)

2. Autorstvo – nekomercijalno (CC BY-NC)

3. Autorstvo – nekomercijalno – bez prerada (CC BY-NC-ND)

4. Autorstvo – nekomercijalno – deliti pod istim uslovima (CC BY-NC-SA)

5. Autorstvo – bez prerada (CC BY-ND)

6. Autorstvo – deliti pod istim uslovima (CC BY-SA)

Potpis autora

U Beogradu, _____

_____