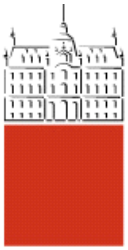
Marsetič, R. 2016. Uporaba metod spodbujevanega učenja za optimizacijo krmiljenja prometa na cestni arteriji. = Application of reinforcement learning methods for optimization of traffic control on arterial roads. Doctoral dissertation. Ljubljana, Univerza v Ljubljani, Fakulteta za gradbeništvo in geodezijo. (Mentor Žura, M.)

http://drugg.fgg.uni-lj.si/5446/

Univerza
v Ljubljani

Fakulteta za
gradbeništvo in
geodezijo



DOKTORSKI ŠTUDIJSKI PROGRAM III. STOPNJE
GRAJENO OKOLJE

Kandidat:
ROK MARSETIČ

UPORABA METOD SPODBUJEVANEGA UČENJA ZA OPTIMIZACIJO KRMILJENJA PROMETA NA CESTNI ARTERIJI

Doktorska disertacija štev: 33/GO

APPLICATION OF REINFORCEMENT LEARNING METHODS FOR OPTIMIZATION OF TRAFFIC CONTROL ON ARTERIAL ROADS

Doctoral thesis No.: 33/GO

Komisija za doktorski študij je na 28. seji, 16. maja 2012, po pooblastilu 30. seje Senata Univerze v Ljubljani z dne 20. januarja 2009, dala soglasje k temi doktorske disertacije.

Za mentorja je bil imenovan doc. dr. Marijan Žura.

Ljubljana, 15. februar 2016

Univerza
v Ljubljani

Fakulteta za
gradbeništvo in
geodezijo

Komisijo za oceno ustreznosti teme doktorske disertacije v sestavi:

- doc. dr. Marijan Žura, UL FGG,
- doc. dr. Tomaž Maher, UL FGG,
- izr. prof. dr. Drago Sever, UM FG,

je imenoval Senat Fakultete za gradbeništvo in geodezijo na 28. seji, 29. februarja 2012.

Poročevalce za oceno doktorske disertacije v sestavi:

- doc. dr. Peter Lipar, UL FGG,
- doc. dr. Tomaž Maher, UL FGG,
- izr. prof. dr. Drago Sever, UM FGPA,

je imenoval Senat Fakultete za gradbeništvo in geodezijo na 24. seji, 9. decembra 2015.

Komisijo za zagovor doktorske disertacije v sestavi:

- prof. dr. Matjaž Mikoš, dekan UL FGG, predsednik,
- izr. prof. dr. Marijan Žura, UL FGG, mentor,
- doc. dr. Peter Lipar, UL FGG,
- doc. dr. Tomaž Maher, UL FGG,
- izr. prof. dr. Drago Sever, UM FGPA.

je imenoval Senat Fakultete za gradbeništvo in geodezijo na 25. seji, 27. januarja 2016.

IZJAVA O AVTORSTVU

Podpisani Rok Marsetič, izjavljam, da sem avtor doktorske disertacije z naslovom: **Uporaba metod spodbujevanega učenja za optimizacijo krmiljenja prometa na cestni arteriji**.

Izjavljam, da je elektronska različica v vsem enaka tiskani različici in dovoljujem objavo elektronske različice v digitalnem repozitoriju UL FGG.

Ljubljana, 15. februar 2016

.....

(podpis)

POPRAVKI

Stran z napako	Vrstica z napako	Namesto	Naj bo

BIBLIOGRAFSKO – DOKUMENTACIJSKA STRAN IN IZVLEČEK

UDK: **656.05: 004.451.26:519.8:(043)**
Avtor: **mag. Rok Marsetič**
Mentor: **izr. prof. dr. Marijan Žura**
Naslov: **Uporaba metod spodbujevanega učenja za optimizacijo krmiljenja prometa na cestni arteriji**
Tip dokumenta: **doktorska disertacija**
Obseg in oprema: **109 str., 10 pregl., 54 sl., 6 en.**
Ključne besede: **spodbujevano učenje, cestna arterija, krmiljenje prometa, svetlobnosignalne naprave**

Izvleček

Današnja družba se zlasti v mestnih središčih pogosteje srečuje s prometnimi zastoji, višjimi stroški, nižjo prometno varnostjo in večjim onesnaženjem okolja. Prometne obremenitve vedno bolj presegajo kapaciteto cestne infrastrukture, posebno v koničnih urah. V mestih je velik delež križišč opremljenih s svetlobnosignalnimi napravami, ki imajo ključno vlogo pri varnem in učinkovitem vodenju prometa. Pri zagotavljanju čim krajših zamud je pomembna hitra prilagoditev krmilnih programov v realnem času. Z omejitvijo širjenja cestne infrastrukture in naraščanjem prometnih obremenitev čedalje težje ohranjamo zadovoljiv prometni nivo uslug s klasičnimi tehnikami krmiljenja. To še posebno velja ob nastanku nepredvidenih prometnih dogodkov. V takšnih situacijah so pomanjkljivosti zdajšnjih krmilnih sistemov še opaznejše. Promet je stohastične narave in spremembe v prometnem toku zahtevajo, da se krmilna strategija nenehno posodablja. Za reševanje tako zahtevnih in kompleksnih problemov optimizacije krmiljenja svetlobnosignalnih naprav potrebujemo sistem, ki se nenehno prilagaja in uči. S pomočjo umetne inteligence je mogoče razviti sisteme, ki so sposobni samostojnega učenja. V doktorski disertaciji smo predstavili sodoben pristop krmiljenja svetlobnosignalnih naprav na cestni arteriji, in sicer z uporabo algoritma spodbujevanega učenja. Predlagani algoritem omogoča hitro in samostojno učenje optimalne strategije krmiljenja v najrazličnejših prometnih razmerah. Uspešnost predlaganega algoritma smo preverili s pomočjo mikrosimulacijskega orodja, s katerim lahko z veliko natančnostjo simuliramo realne prometne razmere. Dokazali smo, da je učinkovitost predlaganega krmiljenja boljša v primerjavi s klasičnim prometno odvisnim krmiljenjem.

»Ta stran je namenoma prazna«

BIBLIOGRAPHIC – DOCUMENTALISTIC INFORMATION

UDC:	**656.05: 004.451.26:519.8:(043)**
Author:	**Rok Marsetič, M.Sc.**
Supervisor:	**assoc. prof. Marijan Žura, Ph.D.**
Title:	**Application of reinforcement learning methods for optimization of traffic control on arterial roads**
Document type:	**Ph.D. Thesis**
Notes:	**109 p., 10 tab., 54 fig., 6 eq.**
Keys words:	**reinforcement learning, road artery, traffic control, traffic lights**

Abstract

Nowadays, society faces several traffic related problems, such as traffic jams, time loss, lower traffic safety, increased pollution, etc., especially in urban areas. This is caused by high traffic volumes, which often exceed the capacity of the road infrastructure, particularly in peak hours. A common way of managing traffic in urban areas is traffic light control, which plays a key role in traffic safety and efficiency. To reduce delays the traffic light controllers should adjust to changing traffic volumes continuously and rapidly. Limited possibilities for road infrastructure extensions and growing traffic volumes represent a challenge for existent control techniques with increasing problem of maintaining suitable level of service. When unexpected events occur, the disadvantage of current traffic control system is even more evident. Stochastic nature of traffic and constant changes in traffic flow requires continuous adaption of traffic light controller. For solving complex problem of traffic lights optimization the system that continuously adapts and learns should be employed. Artificial intelligence approaches enable development of self-learning systems. The thesis presents a novel approach for solving problems of traffic light controller optimization with use of the reinforcement learning. The proposed algorithm enables fast and self-learning optimal strategy of traffic control in different traffic conditions. The efficiency of proposed algorithm was tested using a micro simulation tool, which simulates traffic conditions with great accuracy. The results of the performed experiments show that proposed algorithm outperforms the actuated signal controllers.

»Ta stran je namenoma prazna«

ZAHVALA

Za vodenje pri nastajanju doktorske disertacije se iskreno zahvaljujem mentorju izr. prof. dr. Marijanu Žuri. S svojo strokovnostjo, bogatim znanjem in hkrati preprostostjo mi je pomembno pomagal, da je disertacija zaključena.

Zahvaljujem se sodelavcema Darji Šemrov in Juriju Velkavrhu, za vse diskusije o spodbujevanem učenju ter nesebični pomoči pri programiranju algoritma. Hvala vsem preostalim sodelavcem na Prometnotehniškem inštitutu Fakultete za gradbeništvo in geodezijo za deljenje svojega znanja ter za prijetno delovno vzdušje.

Iz srca hvala staršem, da sta me na moji življenjski poti pravilno usmerjala, toliko stvari omogočila in ves čas verjela vame.

Daša, hvala za podporo in razumevanje, ko nismo mogli biti skupaj z najinima sinovoma Markom in Lukom.

»Ta stran je namenoma prazna«

KAZALO

KAZALO SLIK

LIST OF FIGURES

SEZNAM PREGLEDNIC

LIST OF TABLES

OKRAJŠAVE IN SIMBOLI

a_t	akcija v času t
a_{t+1}	akcija v času $t + 1$
C	ciklus
DP	dinamično programiranje
EAS	enoagentni sistem(i)
g_i	efektivni zeleni čas
G_i	zeleni čas
l_i	dolžina kolone na prometnem pasu
L_m	množica prometnih pasov na glavni smeri
L_s	množica prometnih pasov na prečni smeri
n	število prehodov agenta iz stanja s_t v stanje $s_{t+1,j}$ pri akciji a_t,
$n(s_t, a_t)$	število vseh že izvedenih akcij v stanju s_t
n_ε	faktor izkoriščanja znanja
q	vrednost zajezitvene dolžine
Q_p	pričakovana vrednost Q
r_t	nagrada v času t
SSN	svetlobnosignalna naprava
$s_{t+1,j}$	množica vseh možnih stanj v času $t + 1$
s_t	stanje v času t
s_{t+1}	stanje v času $t + 1$ ob izbiri akcije
SU	spodbujevano učenje
t	čas/korak
t_{va}	trajanje zelenega signala vodilnega agenta
v/c	faktor nasičenosti razmerje volumen proti kapaciteti križišča
VAS	večagentni sistem(i)
w_i	utež posameznega pasu
α	faktor stopnje učenja
γ	faktor diskontiranja
ε	parameter raziskovanja

1 UVOD

1.1 Opis obravnavane problematike

Rast motorizacije, povpraševanja po mobilnosti in posledično večje prometne obremenitve na cestni mreži predstavljajo velik izziv za prometno in računalniško stroko. Zaradi omejitve s prostorom za širjenje cestnega omrežja na eni strani in sprememb potovalnih navad na drugi strani se mesta vsakodnevno spopadajo s prometnimi zastoji, saj povpraševanje presega zmogljivosti cestne infrastrukture. Prometni zastoji se pojavijo, ko prometni volumen preseže kapaciteto infrastrukture. Posledice so nam dobro znane pod pojmi »nevarnost-zastoji-stroški-ogljični odtis«. Z omejitvijo širjena cest nam za povečanje kapacitete preostane optimizacija vodenja prometa na obstoječi infrastrukturi. Učinkovito krmiljenje svetlobnosignalnih naprav je pri tem ključnega pomena. Optimalno krmiljenje lahko dosežemo s pomočjo inteligentnih transportnih sistemov, ki so se sposobni prilagajati spreminjajočemu se prometu.

Inteligentni transportni sistemi, ki imajo vgrajene napredne tehnologije, imajo v prometnem sistemu vedno večjo vlogo pri reševanju prometnih težav, s katerimi se srečuje današnja družba. Z uporabo sodobne računalniške, informacijske in komunikacijske tehnologije je njihov cilj doseči povečanje mobilnosti, varnosti, varčevanja z energijo, zmanjšanje onesnaževanja okolja in nižje stroške za družbo. Za upravljanje s prometnimi tokovi v različnih tipih križišč, predvsem v mestnih središčih, so najbolj razširjene svetlobnosignalne naprave. Čas in razpored posameznih faz razporejajo tako, da so zamude najmanjše, sočasno pa prepustnost križišča oziroma sistema križišč največja. Obenem skrbijo za varnejše odvijanje prometa v križiščih. Jakost prometnih tokov nenehno niha, zato sta strategija in cilj vsakega upravljavca prometa odzivnost sistema v realnem času. Velik delež implementiranih krmilnih sistemov je časovno odvisnih. Časovno odvisni sistemi uporabljajo krmilne programe, ki so narejeni na podlagi predhodnega štetja prometa. Za prometni tok bi lahko rekli, da je živa, spreminjajoča se stvar. Pri tem se pojavlja vprašanje, ali so taki sistemi, ki temeljijo zgolj na zgodovinskih prometnih podatkih dovolj učinkoviti. Kljub nenehni rasti motorizacije in danes tudi fizičnim omejitvam pri izgradnji in širitvi nove infrastrukture je v uporabi še vedno veliko tovrstnih sistemov. Naprednejši sistemi krmiljenja signalnih naprav so prometno odvisni sistemi, ki se sproti prilagajajo prometu. Za razliko od časovno odvisnih sistemov, pri katerih so krmilni programi fiksni, se prometno odvisni sistemi prilagajajo glede na količino prometa, ki ga zaznajo z detektorji. Čas in razpored posameznih faz spreminjajo v odvisnosti od detektiranih vozil. Kljub podaljševanju potrebnega zelenega časa ali zamenjavi trenutne faze so prometno odvisni sistemi učinkoviti le deloma. Prilagajanje oziroma podaljševanje zelenega časa v posamezni smeri je prav tako omejeno in

lahko ne zadostuje ob večjih odstopanjih v prometnem toku (Bingham, 2001). Učinkovitost klasičnih prometno odvisnih sistemov lahko izboljšamo z inteligentnimi sistemi krmiljenja. Ti se nenehno prilagajajo prometnim spremembam, saj imajo zmožnost učenja. Za njihovo primerno delovanje ne zadostuje samo hitro prilagajanje, temveč mora biti sistem zasnovan tako, da se nenehno uči in izboljšuje. Z nenehnim učenjem se tudi spreminja in prilagaja logiko krmiljenja, ki je ključna za dosego zadovoljivih prometnih nivojev uslug.

Temeljno načelo ustreznega nadzora prometa je, da se naprave odzovejo na dinamične spremembe, ki vladajo v vsakodnevnem prometu. Ker je promet stohastične narave, ga je v realnem času težko nadzorovati s t. i. matematičnimi modeli. Vloženo je bilo veliko truda pri razvoju koristnejših modelov. Uporaba vedno kompleksnejših in podrobnejših modelov je prinesla tudi številne omejitve. Velike količine vhodnih podatkov v realnem času težko nadzorujemo in s tem težje upravljamo sistem. Konvencionalne metode trpijo zaradi t. i. »predimenzioniranosti«. Če želimo dobiti najboljšo rešitev, je treba poznati položaj, hitrost in pot vsakega vozila v sistemu. Poleg tega moramo poznati in upoštevati reakcije voznikov. Sistemi krmiljenja prometa z možnostjo učenja, ki na podlagi podatkov v realnem času uravnavajo svetlobne signale v križiščih, imajo prednost pred sedanjimi časovno in klasično prometno odvisnimi sistemi. Nekaj takih sistemov, ki se uporabljajo v današnjem času, to že potrjuje (Mirchandani in Head, 2001). Težave za optimalno kontrolo SSN povzročajo predvsem velike količine neznank, ki jih je treba vnesti, še prej pa analizirati. Problematika je rešljiva z uporabo umetne inteligence (ang. *artificial intelligence*). Novejši inteligentni sistemi so sposobni reševati probleme podobno, kot jih rešujemo ljudje. Omenjeni sistemi imajo zmožnost okarakteriziranja in izbire uporabljenega znanja, prepoznavanja, reševanja problemov, učenja, izločanja, obdelave in izmenjave znanja. Inteligentni sistemi se zmorejo prilagajati novim in neznanim razmeram ter spremembam (Russell in Norvig, 2003). Zato je smiselno, da se na teh osnovah razvijajo novi prilagodljivi sistemi prometa. Glavni poudarek raziskave je usmerjena uporaba inteligentnih tehnik krmiljenja, ki se lahko prilagajajo dinamični naravi prometnega toka.

1.2 Tema in cilji doktorske disertacije

Namen doktorske disertacije je predstaviti različne obstoječe metode krmiljenja prometa na cestni arteriji, ter preveriti možnost uporabe sodobnih principov umetne inteligence za reševanje problema krmiljenja svetlobnosignalnih naprav.

Klasični sistemi krmiljenja prometa so omejeni v smislu učenja in prilagajanja nenehnim spremembam prometnega toka v realnem času. Velike količine prometnih podatkov in hiter odziv na spremembe so pri tem največja težava. Z inženirskega vidika je namen raziskave razviti koncept

krmiljenja SSN v realnem času, ki bo koristnejši kot dosedanji klasični prometno odvisni sistemi. Menimo, da lahko z uporabo algoritma na osnovi spodbujevanega učenja krmilimo arterijo oziroma več križišč v sosledju učinkoviteje kot s klasičnim prometno odvisnim krmiljenjem.

Glavni cilj doktorske disertacije je razviti in preizkusiti algoritem spodbujevanega učenja, ki bo učinkovito in v realnem času krmilil SSN na cestni arteriji. Z raziskavami so avtorji dokazali uspešno uporabo spodbujevanega učenja za reševanje problema krmiljenja SSN. Sklepamo, da je s kombinacijo različnih predlaganih pristopov možno izboljšati učinkovitost krmiljenja, zato smo si zastavili naslednjo delovno hipotezo: *če v algoritmu pravilno upoštevamo stohastično naravo prometa in implementacijske elemente spodbujevanega učenja lahko dosežemo rezultate, ki so boljši od klasičnega prometno odvisnega krmiljenja.*

1.3 Pregled stanja na obravnavanem znanstvenem področju

V tem poglavju podajamo pregled stanja na področju prilagodljivega krmiljenja SSN ter raziskave novih sistemov krmiljenja tako za samostojna križišča kot tudi več križišč v prometni mreži. Za razvoj prilagodljivih krmilnih sistemov obstajata dva pristopa. Eden je z uporabo tehnik umetne inteligence na osnovi hevristike in drugi z uporabo spodbujevanega učenja.

Papis in Mamdani (1977) sta razvila metodo za krmiljenje samostojnega križišča na osnovi mehke logike. Za krmiljenje prometnih mrež pa so mehko logiko uporabili Nakamiti in Freitas (2002) ter Srinivasan in sod. (2006). Tehnike generičnega spodbujevanega učenja za optimizacijo prometnih mrež sta preučevala Mikami in Kakazu (1994). Spall in Chin (1997) sta uporabila umetne nevronske mreže (*ang. artificial neural network*) za določitev optimalnih semaforskih ciklusov. Skupna značilnost navedenih hevurističnih rešitev je ta, da so krmilniki SSN na podlagi zaznanega prometa sposobni prilagoditi krmilne programe. Proces učenja nevronskih mrež se lahko izvaja v realnem času. Robni pogoji prometnih obremenitev so navadno prednastavljeni.

S pomočjo spodbujevanega učenja (SU) sta med prvimi poskušala izboljšati krmiljenje SSN Thorpe in Anderson (1996). V študiji sta za učenje agentov (krmilnikov) uporabila metodo SARSA. Rezultati posameznih akcij oziroma naučene vrednosti sta shranjevala v posebno matriko. Za vsak korak se je vrednostna funkcija akcije posodabljala. Preizkus z metodo sta izvedla na enostavni mreži kvadratne oblike s šestnajstimi križišči. Vsako križišče je imelo štiri krake, vsak krak pa po en prometni pas. Razdalja med križišči je bila cca. 130 m. V križiščih nista upoštevala levih zavijalcev, desne pa sta združila na pas z vozili za smer naravnost. Vsako križišče je bilo krmiljeno z enim agentom in nista upoštevala nobene koordinacije med agenti. Simulacijo sta izvedla s pomočjo programa, ki sta ga samostojno razvila. Upoštevala sta dvesekundni rumeni signal in enosekundni rdeči signal v vseh smereh. Agenti so se lahko odločali za akcije vsako sekundo. V študiji sta raziskovala različne načine opisov stanj (glede na število vozil po smereh, število vozil na posameznem odseku ...). Na podlagi različnih opisov stanj sta imela v vsakem primeru različno število možnih stanj. V najkompleksnejšem opisu stanj sta imela več kot 2000 možnih stanj. Poleg različnih opisov stanj sta testirala tudi vpliv nagrajevanja. Pri izbiri akcij sta v raziskavi upoštevala požrešno strategijo. Predlagano krmiljenje sta primerjala s časovno odvisnim krmiljenjem. Rezultati z metodo SARSA so se izkazali za boljše tako v primeru potovalnih časov kot v povprečnih zamudah. Kljub temu je v raziskavi zaslediti nekaj problemov in nejasnosti. Primerjavo z novim krmiljenjem sta izvedla s časovno odvisnim krmiljenjem, ki ni tako odzivno in učinkovito kot prometno odvisno. Poleg tega nista zapisala, katere krmilne programe sta upoštevala za časovno odvisno krmiljenje, prav tako pa nista razložila, ali so bili programi optimizirani. Druga

pomanjkljivost je, da nista upoštevala levih zavijalcev. Vprašljiva je tudi izbira orodja za simulacije, saj bi avtorja lahko uporabila priznano simulacijsko orodje, ki omogoča natančnejše simuliranje prometa in s tem bi dobila prepričljivejše rezultate.

E. Bingham (2001) je predstavila krmiljenje prometa na samostojnem križišču s pomočjo SU na osnovi metode učenja akter-kritik (ang. *actor-critic*). Analizirala je uspešnost metode na zelo poenostavljenem samostojnem križišču. Križišče je bilo sestavljeno iz dveh enosmernih cest. Uspešnost algoritma krmiljenja je primerjala s klasičnim prometno odvisnim krmljenjem. Simulacije prometa je izvedla s pomočjo programa, ki so ga razvili na Tehniški univerzi v Helsinkih. Analizo je opravila na treh primerih z različnimi prometnimi obremenitvami. Eksperiment je pokazal, da je algoritem krmiljenja uspešen pri zmanjševanju zamud pri konstantnem pretoku vozil, kar pa je za stohastično naravo okolja v realnem svetu težko pričakovano. Podobno kot E. Bingham je tudi Wiering (2000) razvil svoj algoritem na osnovi SU in ga implementiral v svoj simulator prometa »green light district«. V nenasičenih razmerah so predstavljeni rezultati pozitivni, vendar je treba poudariti, da je simulator prometa zelo poenostavljen, saj ne upošteva pospeškov/pojemkov ter različnih hitrosti vozil. Nekoliko slabše rezultate je raziskovalka dobila v nasičenih prometnih razmerah in ob večjih nihanjih prometa.

Abdulhai in sod. (2003) so predstavili popolnoma prilagodljivo krmiljenje SSN na osnovi učenja Q. V študiji so predstavili uporabo učenja Q na samostojnem križišču kot tudi za krmiljenje cestne arterije. Rezultate analize so predstavili samo za samostojno križišče, za arterijo pa ne. Učinkovito krmiljenje cestne arterije je kompleksnejši problem in obenem veliko večji izziv. Avtorji niso upoštevali nobenih zavijalcev v samostojnem križišču. Upoštevali so dvofazni krmilni program s fiksnim ciklusom, kar je za prilagodljive sisteme krmiljenj, ki smo jih pregledali, redkost. Abdulhai in sod. (2003) so za nagrajevanje agenta izbrali skupno zamudo vozil med dvema semaforskima fazama. Stanja okolja so opisali s kolonami na krakih križišča in časom trajanja posamezne faze. V članku ni jasno opredeljena definicija stanj okolja oziroma dolžin kolon. Krmiljenje z učenjem Q so primerjali s časovno odvisnim krmiljenjem za različne prometne scenarije. Ob enakomernem prometu lahko iz analize razberemo, da so rezultati z obema načinoma krmiljenja enakovredni. Večje razlike tudi za več kot 50 % v korist učenja Q so ob spremenljivem prometnem toku. V študiji ni omenjeno, ali je bila narejena optimizacija časovnega krmiljenja. Kljub temu da Abdulhai in sod. (2003) niso predstavili rezultatov za arterijo, v delu predlagajo metodologijo krmiljena z učenjem Q za arterijo in mrežo križišč. Za lažjo koordinacijo med križišči predlagajo dodatne informacije o kolonah na sosednjih križiščih. Pri tem pa so dodali, da dodatne informacije znatno povečajo množice stanj. S tako povečanim številom možnih stanj se pojavi problem učenja agentov.

Za sinhronizacijo SSN je avtorica A. Bazzan (2005) uporabila evolucijsko teorijo igre. Analizo je izvedla na cestni arteriji desetih križišč. Vsak semafor je predstavljal enega agenta. Najvišja nagrada je bila podeljena, če je bila dosežena koordinacija med agenti, tj. z uskladitvijo zelenih signalov. Sistem je zasnovan tako, da omogoča izbiro različnih krmilnih programov glede na izbrano smer koordinacije ali za predhodno definiran čas v dnevu. Na podlagi izbire akcije okolje agentu podeli nagrado ali kazen (nagrada z negativnim predznakom). Vsak agent v sistemu je razpolagal samo z informacijami glede svojega prometnega stanja. Uspešnost pristopa je avtorica predstavila na podlagi števila agentov, ki so dosegli medsebojno koordinacijo, in potrebnega časa za usklajevanje.

Oliveira in sod. (2005) so z namenom izboljšanja prometnega nivoja uslug skupin semaforiziranih križišč uporabili metodo »cooperative mediation«. Optimizacija krmiljenja poteka centralizirano, kar pomeni, da skupina »mediatorjev« sporoča svoje odločitve agentom v svoji skupini, ki nato izvedejo posamezna opravila (akcije). V nasičenih prometnih razmerah pa lahko proces usklajevanja traja dlje časa, kar je neugodno za hiter proces koordinacije med SSN. Zaradi omenjenih težav sta v naslednji raziskavi Oliveira in A. Bazzan (2006) razvila decentraliziran model brez mediacije. Simulacije so izvedli na prometni mreži križišč (5 x 5), ki so jo uporabili že v prispevku Oliveira in sod. (2005). Mreža je bila sestavljena iz 25 vozlišč in 60 krakov. Kapaciteta posameznega kraka je bila 30 vozil v vsaki smeri. Agenti so imeli na razpolago dve možni akciji oziroma izbiro med dvema predhodno definiranima krmilnima programoma. Prvi krmilni program je favoriziral smer sever–jug, drugi pa vzhod–zahod. Razmerje med fazama je bilo 40 : 20. Oba krmilna programa sta imela 60-sekundni ciklus. Agenti so lahko zamenjali krmilni program na 10 minut. Mrežo so obremenjevali z različnimi prometnimi intenzitetami in za vsak primer so izvedli 100 simulacij. Prav tako so preizkušali vpliv spodbud med agenti. Dokazali so bistveno učinkovitejše prilagajanje med agenti v primerjavi s predhodno raziskavo.
Nato so vse analizirane podatke uporabili za učenje nevronske mreže, ki generira akcije za posamezne prometne razmere. Sistem uporablja zelo majhno količino procesorskega časa za izračun optimalnih vrednosti, saj predhodno učenje nevronske mreže ne poteka v realnem času. Obenem je slednje tudi največja pomanjkljivost pristopa, saj je treba ročno določiti merodajne prometne obremenitve. Poleg tega je treba vnaprej določiti zaporedje posameznih faz. S predlaganim pristopom so dobili rezultate, ki so primerljivi z uporabo DP.

Gregoire in sod. (2007) so optimizirali krmilne programe na prometni mreži, sestavljeni iz štirih križišč. Z uporabo algoritma, zasnovanega na podlagi strojnega učenja, so avtorji skušali izboljšali rezultate v primerjavi s časovno odvisnim krmiljenjem. Za prometni simulator so uporabili samostojno izdelan program. Prometna mreža je bila sestavljena iz ene daljše ceste, ki so jo križale štiri krajše. Štiri križišča v sosledju so bila razmaknjena 280 m. Horizontalne ceste so bile

dvopasovne, medtem ko so bile vertikalne štiripasovne. V križiščih niso upoštevali nobenih zavijalcev, tako da so vozila v horizontalni smeri morala prevoziti štiri križišča, vozila v vertikalni pa samo eno. Promet je na vseh krakih potekal dvosmerno. Na vseh križiščih so upoštevali dvofazne krmilne programe. Tako so imeli krmilniki oziroma agenti na razpolago dve akciji, in sicer podaljšanje faze ali preklop druge faze. Stanja so opisali s številom čakajočih vozil v horizontalni smeri, s številom čakajočih vozil v vertikalni smeri v trenutni fazi in s številom podaljševanja faze. Za nagrado so uporabili zamude vozil, ki so jo označili z negativnim predznakom. Testiranje algoritma so izvedli najprej na samostojnem križišču in nato še na celotni mreži. Prometne obremenitve so bile enakovredne kapaciteti križišča. Ob konstantnih prometnih obremenitvah so bili rezultati krmiljenja z algoritmom primerljivi z rezultati časovno odvisnega krmiljenja in nekoliko boljši v spremenljivih prometnih tokovih.

V svoji raziskavi je Cai (2007) za krmiljenje samostojnih križišč uporabil aproksimacijsko dinamično programiranje. S tem je reševal problem predimenzioniranosti in nepopolnih informacij, ki se pojavljajo pri metodi dinamičnega programiranja (DP) v realnem času. Tako je bistveno zmanjšal število možnih stanj okolja. Za načrtovanje in posodobitve krmilnih programov je uporabljal napoved prihodov vozil za naslednjih 10 sekund. Rezultati eksperimenta so pokazali, da je učinkovitost krmiljenja z aproksimacijskim DP tolikšna kot s klasičnim prometno odvisnim krmiljenjem.

Heydecker in sod. (2007) so prav tako kot Cai uporabili aproksimacijsko DP za krmiljenje. Razlika je bila le v kompleksnejši geometriji križišča, kjer so izvedli testiranja. Vendar je rezultate primerjal s časovno odvisnim krmiljenjem. Razvidno je, da so s pristopom aproksimacijskega DP zmanjšali zamude za polovico v primerjavi s časovno odvisnimi krmilnimi programi.

Wunderlich, Elhanany in Urbanik (2008) so za krmiljenje samostojnega križišča predlagali modificiran algoritem »longest-queue-first«. Predlagana metoda uporablja algoritem, ki minimizira zajezitvene dolžine na vsakem kraku in s tem znižuje povprečne zamude vozil v križišču. Analizo so opravili na štirikrakem križišču. Na vsakem kraku so predvideli leve zavijalne pasove in po en kombiniran pas za vožnjo naravnost in desno. Za krmiljenje semaforja so uporabili različne kombinacije dvofaznih krmilnih programov. Simulacije so opravili na podlagi srednje velikih in velikih (blizu kapaciteti) prometnih obremenitev. Upoštevali so tri tipe struktur prometa. Primerjali so dva predlagana tipa algoritma krmiljenja s časovno in prometno odvisnim krmiljenjem. Prvi tip krmiljenja ne omogoča prednosti tovornim vozilom. Drugi tip omogoča prednost tovornim vozilom z možnostjo podaljševanja zelenega signala. V prvem primeru so stanja opisali s kolonami vozil, v drugem primeru pa s številom vozil na pasu ter pri tem upoštevali večjo utež za tovorna vozila. Algoritem so zapisali s programskim orodjem Matlab ter z vmesnikom poganjali simulacije v

mikrosimulacijskem modelu. Dokazali so, da je s predlaganim algoritmom mogoče doseči nižje zamude vozil v primerjavi s klasičnimi metodami krmiljenja. Dodali so še, da je zadnja verzija algoritma zmožna krmiliti semaforje v realnem svetu.

Tako kot v svoji predhodni študiji so Cai in sod. (2009) za krmiljenje samostojnega križišča uporabili izboljšano verzijo algoritma na osnovi aproksimacijskega DP. Avtorji so za aproksimacijo vrednostne funkcije uporabili pristop z nevronskimi mrežami. Predlagan algoritem omogoča uporabo različnih tehnik strojnega učenja. V prispevku so predstavili dva primera, in sicer z učenjem po časovnih razlikah in s perturbacijskim učenjem. Z obema tehnikama učenja so dosegli enakovredne rezultate. Eksperiment so naredili na trikrakem križišču. Vsak krak je imel po eno enosmerno povezavo. Upoštevali so trifazni krmilni program (vsak krak je imel svojo fazo). Poleg različnih tipov učenja so analizirali vpliv frekvence spreminjanja krmilnih programov oziroma akcij. Interval odločanja algoritma je bil med 0,5 in 5 s. V študiji so dokazali, da je mogoče zmanjšati zamude v križišču v primerjavi s časovno odvisnim krmiljenjem za več kot 40 % (z najdaljšim intervalom) in več kot 60 % (z najkrajšim intervalom).

Na osnovi sistema TUC s ciljem krmiljenja večjih mrež sta de Olivera in Camponogara (2010) predlagala nov večagentni sistem (VAS) krmiljenja, tako da sta razčlenila model v manjša podobmočja. S tem vsak agent pridobiva podatke, krmili in koordinira svoje podobmočje v povezavi s sosednjimi agenti. S tem sta dosegla hitro konvergenco rezultatov. Rezultati s predlaganim pristopom so primerljivi z rezultati, pridobljenimi s sistemom TUC. Podoben pristop sodelovanja med sosednjimi agenti z uporabo SU sta uporabila Cahill in Salkham (2008). Pridobljeni rezultati so bili v primerjavi s časovno odvisnim krmiljenjem boljši, vendar primerjava s prometno odvisnim krmiljenjem, ki bi prikazala realnejšo učinkovitost, ni bila narejena.

Arel in sod. (2010) so s pristopom strojnega učenja na osnovi nevronskih mrež raziskovali krmiljenje SSN na mreži petih križišč. SU so uporabili samo za krmiljenje srednjega križišča. Preostala štiri križišča so bila krmiljena na podlagi algoritma »longest-queue-first«. Za vseh pet križišč so upoštevali enako štirikrako geometrijo. Vsak krak je imel po dva vhodna in dva izhodna pasova za 40 vozil. Sistem sodelovanja med križišči je zasnovan tako, da obodna štiri križišča pošiljajo prometne informacije centralnemu križišču. Stanja okolja so definirali z osemdimenzionalnim vektorjem. Vsak element je predstavljal relativni prometni tok na posameznem pasu. Relativni prometni tok je bil definiran s skupno zamudo vseh vozil na pasu deljen s povprečno zamudo na vseh pasovih. Zaradi velikega števila stanj okolja so avtorji uporabili metodo nelinearne aproksimacije funkcije. Na vseh križiščih so upoštevali štirifazne krmilne programe. Možne akcije agentov so bile podaljšanje trenutne kombinacije faz ali njihova zamenjava. Osnova za nagrade so bile zamude vozil. Če je bila

zamuda v trenutni fazi manjša kot v prejšnji, je sistem podelil pozitivno nagrado (1), v obratnem primeru pa negativno (– 1). Simulacije so bile narejene v okolju Matlab. Avtorji so zapisali, da je razvit algoritem zmožen krmiliti tudi večje mreže z več križišči.

Houli in sod. (2010) so v svoji raziskavi za krmiljenje SSN predlagali nov algoritem na osnovi SU, imenovan »multi-RL«. Uporabili so dve vrsti agentov, in sicer so bila prva vrsta agentov vozila, druga pa SSN. Prometna mreža je bila sestavljena iz sedmih križišč. Na vseh križiščih so upoštevali vse možne manevre. Za vsa vozila v sistemu so upoštevali enako hitrost. Cilj krmiljenja je bil zmanjšati zamude, število ustavljanj in skrajšati kolone. Za izmenjavo podatkov med vozili in SSN so uporabili *ad hoc* brezžično omrežje. Pri tem so upoštevali, da so vsa vozila v mreži opremljena z oddajniki. Izmenjane informacije so bile: prometni tok skozi križišča v vsakem časovnem koraku, dolžine kolon na vseh krakih, vrste vozil (osebni avtomobili, tovornjaki, avtobusi in intervencijska vozila), pot vsakega vozila in manever vsakega vozila. Algoritem pri krmiljenju je upošteval štiri stanja na posameznih krakih: prosti prometni tok, srednje zgoščen prometni tok, zgoščen prometni tok in krmiljenje z nudenjem prednosti avtobusom ter intervencijskim vozilom. Ob prihodu avtobusa ali intervencijskega vozila se je v najkrajšem možnem času preklopil zeleni signal. Prosti prometni tok so določili, ko je vstopilo v mrežo manj kot 90 vozil/minuto, med 90 in 180 vozili/min za srednje zgoščen tok in zgoščen prometni tok za več kot 180 vozil/min. Prihode vozil v mrežo so variirali med 30 in 270 vozili, ki so jih spreminjali vsako minuto. Za simuliranje so uporabili uveljavljeno simulacijsko orodje Paramics. Trajanje simulacije je bilo 10.000 časovnih enot, od katerih je bilo prvih 40 % namenjenih učenju algoritma, preostalih 60 % pa za shranjevanje rezultatov. Rezultate svoje metode so primerjali s časovno odvisnim krmiljenjem in krmiljenjem, ki ga je predlagal Wiering s sodelavci (2004). Rezultati, pridobljeni s predlaganim algoritmov, so bili za različne prometne razmere in merila učinkovitosti boljši od običajnih tehnik krmiljenja. Kljub dobrim rezultatom so avtorji poudarili, da je implementacija takega sistema ekonomsko vprašljiva, saj bi za primerno delovanje morali z oddajniki opremiti vsa vozila na terenu.

Kot trdita v svoji raziskavi Prashanth in Bhatnagar (2011), sta prva, ki sta za krmiljenje semaforiziranih križišč uporabila pristop SU v kombinaciji z aproksimacijo funkcije. V raziskavi sta predstavila algoritem učenja Q modeliranega kot enoagentnega sistema. Algoritem omogoča uspešno krmiljenje večjega števila križišč. Problem predimenzioniranosti glede možnih stanj okolja so rešili tako, da so stanja opisali s tremi stopnjami (nizko, srednje visoko in visoko) zasedenosti prometne mreže. Rezultate so primerjali s časovno odvisnim krmiljenjem, algoritmom najdaljše kolone in posebnim algoritmom učenja Q, ki ne uporablja aproksimacije. Uporabljeno razmerje prometnih obremenitev glavnih smeri proti stranskim smerem je bilo 100 : 5. Kot sta zapisala, je predlagani algoritem v vseh testih in primerjavah prekosil ostale načine krmiljenja. Analizo sta izvedla v arteriji

dveh križišč, mreži štirih križišč, mreži devetih križišč in arteriji z osmimi križišči v sosledju. Algoritem sta za potrebe simulacij implementirala v program na osnovi Jave.

Avtorja Khamis in Gomaa (2014) sta za krmiljenje semaforjev uporabila SU, s katerim sta skušala zmanjšati zamude in skupni čas potovanja. Z upoštevano komunikacijo med agenti jima je uspelo doseči tudi linijsko koordinacijo na prometni mreži in s tem zmanjšati porabo goriva. Problem je bil oblikovan kot VAS. Simulacije prometa sta izvedla s simulatorjem »green light district«. Svoj algoritem sta primerjala s prometno odvisnim krmiljenjem (Self-Organizing Traffic Lights – SOTL) na osnovi umetne inteligence in Wieringov algoritem krmiljenja (2004). Algoritem se je izkazal za učinkovitega tako v nasičenih prometnih razmerah kot tudi v situacijah prostega prometnega toka.

Prabuchandran in sod. (2014) so oblikovali problem krmiljenja prometa tako, da so uporabili markovske procese in uporabili algoritme na osnovi večagentnega spodbujevanega učenja (ang. *multi-agent reinforcement learning MARL*). S tem pristopom so dosegli, da se lahko strategija krmiljenja dinamično spreminja glede na spreminjajoče se razmere. Vsako križišče v mreži oziroma semafor je predstavljal en agent. Vsak agent se določa o trajanju posamezne faze na podlagi požrešne strategije. Informacije o izvedenih akcijah in stanjih okolja so se zapisovale v matriko, iz katere so agenti črpali znanje. Problem velikega števila možnih stanj so rešili tako, da je vsak agent razpolagal samo z informacijami stanj na svojem križišču. Stanja so opisali kot kolone vozil na posameznem pasu križišča. Kolone vozil so razdeli v tri razrede. Dodeljena spodbuda je bila različna za vsakega agenta. Pogojena je bila glede na število sosednjih križišč in vsote kolon na krakih. Eksperimente so izvedli na prometni mreži devetih križišč in v mreži dvajsetih križišč. Pristop se je izkazal za učinkovit pri zmanjšanju povprečnih zamud. Za primerjavo učinkovitosti algoritma so uporabili mikrosimulacijsko orodje. Svoj algoritem so primerjali s časovno odvisnim in prometno odvisnim krmiljenjem. Za nadaljnje delo so si avtorji zastavili dokaz konvergence algoritma, saj je v sklopu tega prispevka niso dokazali.

1.4 Vsebina doktorske disertacije

Doktorska disertacije je razdeljena na šest poglavij. V prvem poglavju je opisana obravnavana problematika, zastavljeni cilji in pregled stanja na obravnavanem znanstvenem področju.

V drugem poglavju so predstavljene svetlobnosignalne naprave za krmiljenje prometa v križiščih. Opisana je osnovna terminologija krmiljenja ter časovno odvisno in prometno odvisno krmiljenje. Podrobneje so predstavljeni uveljavljeni prometno odvisni sistemi krmiljenja.

V tretjem poglavju so predstavljena teoretična izhodišča disertacije. Predstavljen je pristop SU kot relativno nov pristop pri reševanju prometne problematike. Nadalje so opisani vsi elementi SU in predstavljeni osnovni pristopi reševanja SU. Poudarek je na metodi učenja po časovnih razlikah. Poglavje je sklenjeno z opisom algoritma učenja Q, s pomočjo katerega je mogoče krmiliti SSN.

Četrto poglavje opisuje razvoj algoritma. V prvem delu poglavja so opisane razlike, prednosti in slabosti enoagentnih in večagentnih sistemov. Sledi opis okolja, stanja, akcije, nagrade in način izbiranja akcij sistema krmiljenja svetlobnosignalnih naprav. Na koncu poglavja je podrobno predstavljena še formalizacija predlaganega algoritma, zasnovanega na učenju Q.

V petem poglavju so predstavljeni rezultati algoritma na cestni arteriji. Poglavje je sestavljeno iz več podpoglavij. Začne se s predstavitvijo mikrosimulacijskega orodja, s katerim so bili narejeni testi učinkovitosti algoritma. Nadalje so predstavljene raziskave vpliva vhodnih prametrov na izhodne rezultate krmiljenja s pomočjo učenja Q. Podrobno je analizirana konvergenca algoritma v različnih prometnih razmerah. Uspešnost in učinkovitost algoritma je bila preverjena za nenasičene in nasičene prometne razmere. Rezultati, pridobljeni s predlaganim algoritmom, so nato primerjani s prometno odvisnim krmiljenjem prometa. Na koncu so podani še sklepi primerjave med klasičnim prometno odvisnim krmiljenjem ter krmiljenje z uporabo algoritma na osnovi učenja Q.

V šestem poglavju so predstavljene ugotovitve in prispevek doktorske disertacije. V sklepu so predstavljena še izhodišča za nadaljnje delo.

2 SVETLOBNOSIGNALNE NAPRAVE

2.1 Razvoj svetlobnosignalnih naprav

Glavna naloga SSN je varno in učinkovito vodenje prometa v križiščih. Začetek uporabe semaforjev za vodenje prometa v križiščih sega v leto 1868 v London. Podobne naprave so se v večjem številu začele pojavljati štirideset let kasneje v New Yorku. Kmalu zatem so leta 1914 prvič uporabili električno krmilno napravo. Po tem mejniku so v New Yorku postopoma začeli mehanske semaforske naprave zamenjevati z električnimi, avtomatskimi napravami. Zadnjo mehansko krmilno napravo so zamenjali z elektromehanskim semaforjem leta 1932. Elektromehanski krmilniki so prevladovali na trgu vse od dvajsetih do sedemdesetih let 20. stoletja (Bullock in Abbas, 2001). Vsi ti sistemi so bili časovno odvisni in s porastom motorizacije se je pokazala njihova pomanjkljivost, saj niso bili dovolj prilagodljivi. Za osveževanje programov so bila potrebna nova štetja prometa. Zaradi omenjene pomanjkljivosti so sčasoma razvili prometno odvisne krmilne sisteme, ki pa so potrebovali dodatno opremo – detektorje prometa.

Največji napredek krmilnih naprav se je pokazal z razvojem mikroprocesorjev, s katerim sta prišla nov zagon in napredek. V zgodnih šestdesetih letih prejšnjega stoletja so se začele uporabljati prometno signalne naprave v kombinaciji z računalniki. Pravi razmah v uporabi računalniško podprtih krmilnih naprav se je pojavil po letu 1970. Takrat so računalniki postali bolj razširjeni in standardizirali sta se strojna in programska oprema (Bullock in Urbanik, 2000). Velik potencial računalnikov na področju krmiljenja sta spoznala Ameriški oddelek za promet (*Department of Transportation – DOT*) in Ameriška zvezna uprava za ceste (*Federal Highway Administration – FHWA*), ki sta vložila veliko napora v razvoj prometno odvisnih sistemov. Razvoj se je nadaljeval in naprave so prerasle v kompleksne in zmogljivejše sisteme krmiljenja.

2.2 Osnove in terminologija krmiljenja

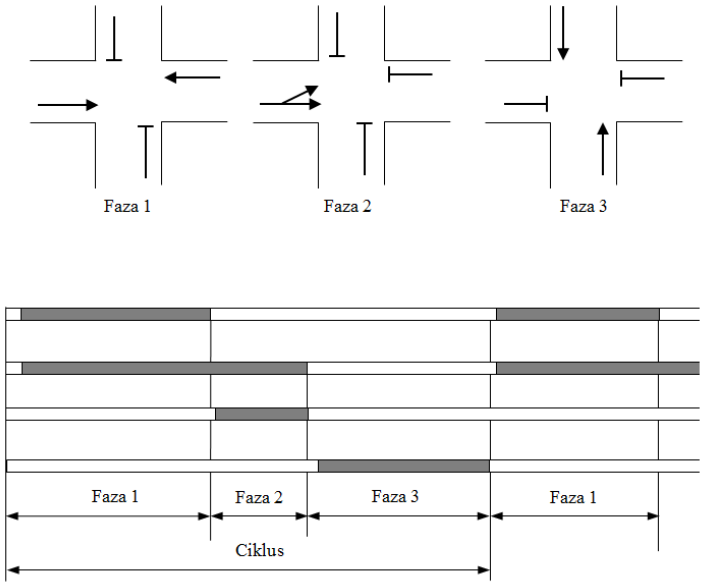
Krmiljenje SSN lahko opišemo kot krogotok različnih procesov, ki potekajo znotraj časovnega intervala. Posamezni procesi so: zbiranje podatkov, odločanje, izvrševanje, preverjanje in vrednotenje. Zbiranje podatkov je osnova za učinkovito krmiljenje in upravljanje s prometom. Lahko se izvaja na različne načine in z različnimi napravami. Glede na trenutne ali napovedane prometne razmere se izvede odločanje oziroma primerna reakcija sistema. Odločitve se oblikujejo na podlagi izbrane krmilne strategije. Ustrezna izbira krmilne strategije je temeljni pogoj za učinkovito

krmiljenje prometa. Izbira je lahko ročna (človek) ali avtomatska (naprava – algoritem). Na osnovi strategije sledi izvrševanje posameznih ukrepov ter preverjanje odločitev oziroma rezultatov. Nadzoruje se lahko z različnimi napravami, bodisi posamezno – večkratno ali konstantno. Za uspešen nadzor delovanja in morebitne spremembe se izvede še vrednotenje rezultatov.

Ob prihodu v križišče si vozniki kot uporabniki prometne infrastrukture želijo vožnjo brez ustavljanja oziroma s čim manj čakanja. To še posebej velja za voznike na glavnih vpadnicah, ki pričakujejo tako imenovano linijsko koordinacijo (Gordon in sod., 2005). Temu je treba prilagajati krmilne programe. Krmilni program vsebuje različne spremenljivke. Za analizo in določanje kapacitete semaforiziranega križišča je treba imeti vse podatke delovanja. Med te podatke sodijo dolžine ciklusov, zeleni časi, izgubljeni časi in shema faznih načrtov. Pogosti termini in definicije posameznih spremenljivk, ki so sestavni del krmilnega programa, so:

- ciklus (C): čas, ki je potreben za celotno sekvenco vseh signalnih intervalov oziroma faz;
- faza: del ciklusa za posamezen prometni pas (ali več prometnih pasov) v križišču;
- interval: določen del ciklusa, znotraj katerega signalni znak ostane nespremenjen;
- razdelitev ciklusa: delež celotnega ciklusa, ki je dodeljen vsaki fazi v ciklusu;
- časovni zamik: začetna časovna razlika začetka zelenih signalov v glavnih smereh križišč v sosledju;
- zeleni čas (G_i): čas zelenega signala za posamezno fazo i;
- izgubljeni čas: ko se križišče efektivno ne uporablja za nobeno skupino voznih pasov, sestavljen je iz izgube časa pri speljevanju in izpraznitve križišča oziroma zagotovitve potrebne varnosti med fazami;
- efektivni zeleni čas (g_i): čas posamezne faze i, ko lahko vozila (pešci) skupine voznih pasov prevozijo (prehodijo) križišče.
- delež zelene luči: razmerje med efektivnim zelenim časom posamezne faze in celotnim ciklusom;
- efektivni rdeči čas: čas, ko je onemogočeno prečkanje skupine voznih pasov za določeno fazo. Za posamezno fazo predstavlja celoten ciklus minus efektivni zeleni čas.

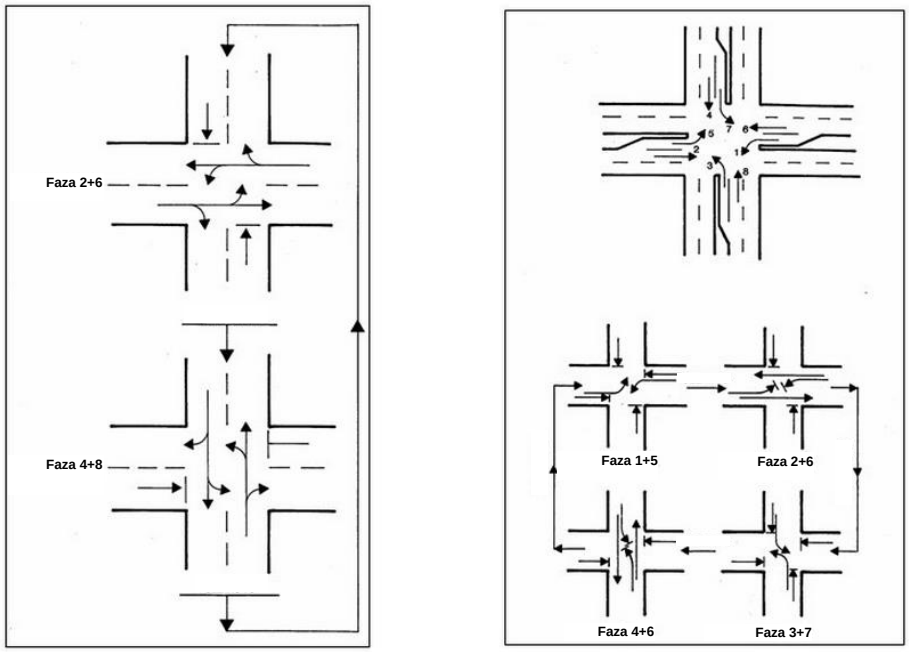
Na sliki 2.1 je predstavljeno razmerje med fazo in ciklusom.



Slika 2.1: Grafični prikaz razmerja med fazo in ciklusom

Figure 2.1: Graphical representation between phase and cycle

Pri semaforiziranih križiščih je ključnega pomena izdelava optimalnega krmilnega programa. Krmilni program je sestavljen iz posameznih faz in njihovega zaporedja. Večinoma so v uporabi dvofazni krmilni programi, razen če prometne obremenitve zavijalcev zahtevajo dodatne faze. Vendar se z večanjem števila faz veča tudi izgubljeni čas, saj prehodi med fazami predstavljajo izgubo. Dolžino zelenega časa posamezne faze narekuje prometni volumen, ki med posamezno fazo prečka križišče. Smer, ki ima največji delež vsega prometa, ima navadno tudi najdaljši zeleni čas. Če so v semaforiziranem križišču navzoči tudi pešci, moramo upoštevati minimalni zeleni čas posamezne faze čas, ki je potreben, da pešec varno prečka prehod (HCM, 2010). Pri razvoju algoritma smo to upoštevali z robnim pogojem minimalnega trajanja zelenega časa. Torej prva zamenjava faze je možna izključno po poteku minimalnega zelenega časa. Primer dvo- in štirifaznega krmilnega programa je prikazan na sliki 2.2.



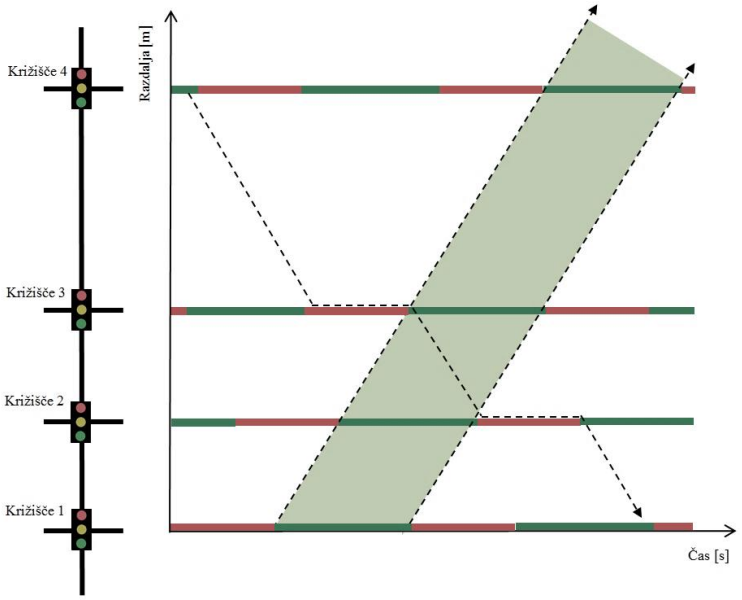
Slika 2.2: Primer dvo- in štirifaznega krmilnega programa (Gordon in sod., 2005)

Figure 2.2: Example of 2-phase and 4-phase signal sequence (Gordon et al., 2005)

Obstajajo trije osnovni tipi krmiljenja SSN: samostojno krmiljenje križišča, krmiljenje arterije in krmiljenje mreže. Za slednja dva primera je koordinacija SSN izjemnega pomena. Pri koordinaciji je treba biti pozoren na časovne zamike med križišči. Časovno-potni diagram (slika 2.3) prikazuje sinhroniziran oziroma koordiniran sistem cestne arterije. Iz časovno-potnega diagrama je dobro vidna linijska koordinacija med SSN na cestni arteriji. Na sliki je prikazan enostaven primer, ko širina zelenega signala izkorišča celoten zeleni čas za eno smer (pod predpostavko, da je v tej smeri bistveno večji prometni tok). V takem primeru imamo v drugi smeri več ustavljanj.

Časovno razliko med prvim in zadnjim vozilom, ki mu še ni treba ustaviti, imenujemo širina zelenega signala. Zamik ciklusa enega križišča glede na predhodno križišče je časovna razlika med začetkoma zelenih luči med dvema sosednjima SSN. Določevanje linijske koordinacije za enakomerne razdalje med križišči je bistveno lažje kot pri neenakih razdaljah. Pri slednjih je ročni proces določitve dolgotrajen, pri večjih mrežah skoraj neobvladljiv. S pomočjo računalniške tehnologije pa enostavnejši in mogoč. Na cestni arteriji je možno doseči tekoč in neoviran pretok vozil le s primerno koordiniranim krmiljenjem SSN. Za vsa koordinirana križišča na arteriji je treba izračunati ciklus ter minimalno in maksimalno fazo na primarni in sekundarni smeri. Osnova za določitev tega so prometne obremenitve. Navadno ima najbolj obremenjeno, geometrijsko največje križišče tudi najdaljši ciklus. Najpogosteje je tako križišče tudi primarno pri določanju koordinacije.

Klasičen problem koordinacije sistema je določitev optimalne širine linijske koordinacije za različne cikluse in hitrosti. Najbolj znani rešitvi sta »hill climbing« (Robertson, 1969) in mešano celoštevilčno programiranje (ang. *mixed-integer programming*) (Morgan in Little, 1964).



Slika 2.3: Časovno-potni diagram koordiniranih SSN na arteriji

Figure 2.3: Time-space diagram

2.3 Časovno odvisno krmiljenje

Pri časovno odvisnih napravah so krmilni programi vnaprej določeni in se spreminjajo le v odvisnosti od ure v dnevu. Na podlagi zgodovinskih podatkov prometa se določi krmilni program, ki ima fiksno dolžino ciklusa, zaporedje faz in zamik med križišči, če je govor o več križiščih. V prometnih konicah so navadno dolžine ciklusov daljše ali celo maksimalne, zunaj koničnih obremenitev pa so dolžine ciklusov krajše. Torej krmilni progami so vnaprej določeni za določen čas v dnevu, lahko tudi v tednu. Med prometnimi konicami časovno odvisne naprave nudijo dokaj učinkovite prometne razmere. Zunaj koničnih ur, predvsem ponoči, je promet na glavnih krakih večkrat ustavljen po nepotrebnem. Pri takem tipu krmiljenja je odprava omenjene pomanjkljivosti možna le s preklopom semaforja na utrip rumene luči, seveda če geometrija križišča oziroma prometna varnost to dopušča. Glavna pozitivna lastnost teh sistemov je predvsem v tem, da je so relativno poceni, saj ni potrebna dodatna infrastruktura v križišču (detektorji vozil). Vzdrževanje je nezahtevno. Slabost je v ažurnosti vhodnih podatkov, saj se v primeru rasti prometa ali spremembe strukture prometnih tokov krmilni programi samodejno ne prilagodijo.

2.4 Prometno odvisno krmiljenje

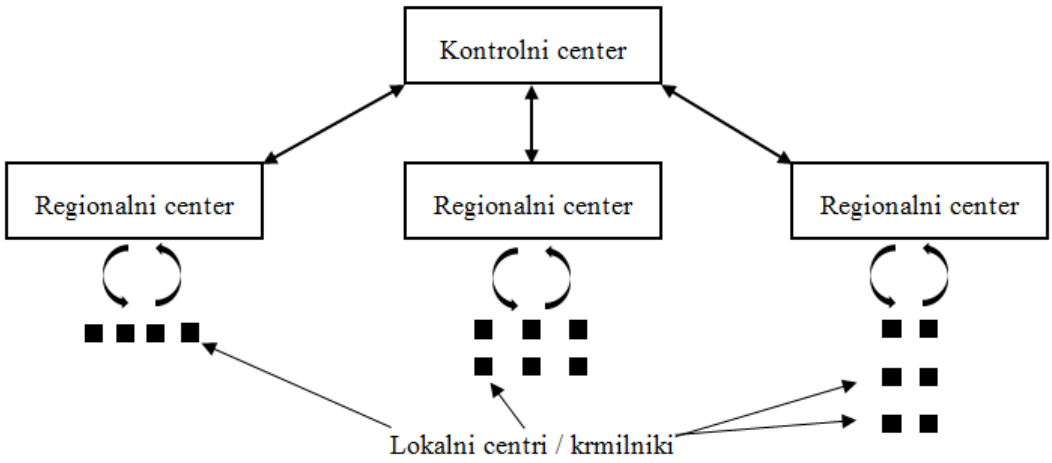
Prometno odvisne SSN spreminjajo krmilne programe na podlagi zaznanega prometa na posameznih krakih križišča. Za razliko od časovno odvisnih imajo prometno odvisni sistemi detektorje, s katerimi se zaznava promet. Najbolj razširjeni so zančni detektorji, vse bolj pa se uporabljajo tudi drugi načini (mikrovalovni detektorji, videodetekcija). Prometno odvisne semaforje lahko delimo v dve vrsti, in sicer v delno in polno prometno odvisne sisteme. Sistema se ločita v odvisnosti od vrste opreme in prometnih zahtev. Pri delno prometno odvisnih semaforjih so detektorji postavljeni samo na stranskih krakih križišča. Za razliko od delnega so pri polnem prometno odvisnem križišču detektorji postavljeni na vseh krakih križišča. Z detektorji se ugotavlja zasedenost oziroma razmik med vozili, nato pa vgrajena logika krmilnika določa, ali se trenutna faza podaljša ali zamenja druga faza. Ne glede na tip prometno odvisnega krmiljenja je treba določiti minimalne zelene čase ter maksimalne čase intervalov oziroma ciklusa. Prednosti prometno odvisnega sistema se kažejo v prilagoditvi krmilnih programov na kratkotrajna nihanja prometnega toka. Vzpostavitev takih sistemov zahteva višje stroške, saj potrebujejo dodatno strojno in programsko opremo.

Za optimizacijo pri prometno odvisnem krmiljenju imamo možnost spremembe dolžine faz ali spremembo dolžine ciklusa. Pri slednjem moramo paziti, da izbira dolgih ciklusov ne povzroči nepotrebnih čakalnih časov, predvsem na stranskih smereh. Po drugi strani kratki ciklusi povzročijo velik delež izgub zaradi izpraznitve križišča zaviranja in speljevanja. Tretja možnost optimizacije je določitev optimalnega časovnega zamika. Pri sistemih z možnostjo prilagajanja v realnem času je možna neprekinjena optimizacija časovnih zamikov. Pri zelo visokih nasičenostih, npr. nad 90 %, je določitev linijske koordinacije omejena. Razlog so daljše kolone med križišči. Pri prilagajanju zelenih časov je treba biti pozoren še posebej pri minimalnih časih zelenega signala. Navadno so minimalne vrednosti določene iz varnostnih razlogov. Za razliko od omenjenih postopkov, pri katerih spreminjamo spremenljivke krmiljenja samo v okviru vnaprej določenega ciklusa, obstajajo tudi aciklični postopki, ki niso omejeni z vnaprej določenim ciklusom in razporedom faz. Pri tem sta omejitvi le minimalna in maksimalna dolžina zelenih časov. Sistemi prometno odvisnega krmiljenja SNN so navadno aciklični. Taki postopki so fleksibilnejši in učinkovitejši pri optimizaciji.

2.5 Prometno odvisni sistemi krmiljenja

2.5.1 Sistem SCATS

Sistem SCATS (*Sydney Coordinated Adaptive Traffic System*) na podlagi prometnih podatkov, pridobljenih iz detektorjev, izbere najprimernejši krmilni program iz predhodno določenih programov (Bullock in Abbas, 2001). Sistem deluje na hierarhičnem sistemu, ki ima tri nivoje, in sicer lokalni, regionalni in kontrolni center. Zgradba sistema SCATS je prikazana na sliki 2.4. Lokalni krmilniki za vsako SSN predstavljajo najnižji nivo in skrbijo za zbiranje podatkov, obdelavo podatkov in zaznavo napake na detektorjih (Gordon in sod., 2005). Drugi nivo je regionalni oziroma taktični nivo, ki je obenem jedro sistema SCATS. Taktični nivo določa parametre posameznih križišč. Vsak regionalni nadzorni sistem nadzoruje po več sto kontrolerjev, ki so združeni v sisteme oziroma podsisteme. Podsistemi navadno tvorijo le nekaj križišč, ki obenem predstavljajo najmanjši del v celotnem sistemu. Iz aktualnih prometnih podatkov oziroma stopnje nasičenosti se določa dolžina zelenih časov, zamik in dolžina ciklusa. Zanke za detekcijo prometa so postavljene neposredno pred črto stop vsakega kraka. S pomočjo detektorjev se izvaja štetje vozil in merjenje časa nezasedenosti med zelenim časom. Krmilni algoritem na osnovi izmerjenih parametrov tekočega ciklusa iz predhodno pripravljenih programov izbere najboljši krmilni program za naslednji ciklus. Algoritem upošteva optimalni zamik v koordinirani skupini semaforjev, razdelitev zelenih časov in dolžino ciklusa. Če se dolžini ciklusov sosednjih skupin ujemata, upošteva tudi optimalno uskladitev. Najvišje je kontrolni center, ki ne izvaja nobene specifične nadzorne operacije, temveč le spremlja delovanje celotnega sistema. Pomanjkljivost sistema SCATS je draga investicija implementacije in slabša odzivnost v zasičenih prometnih razmerah.



Slika 2.4: Zgradba sistema SCATS (Lowrie, 1999)

Figure 2.4: Framework of SCATS system (Lowrie, 1999)

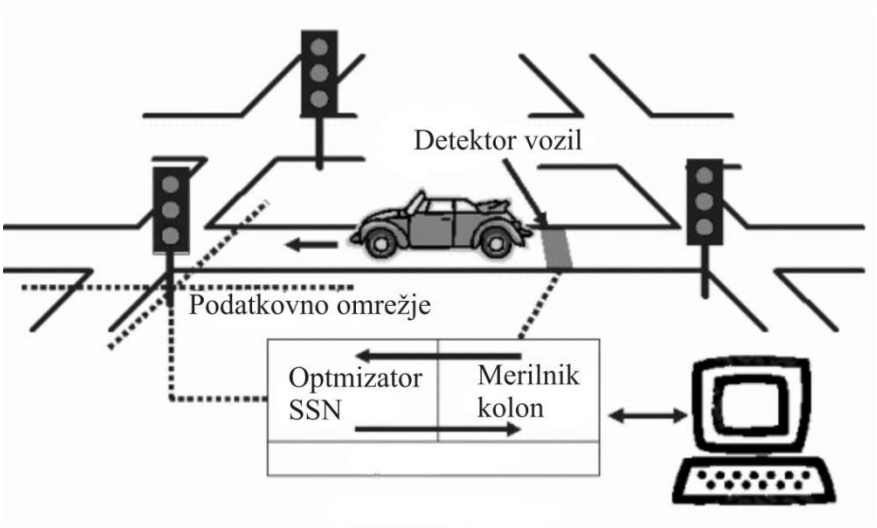
2.5.2 Sistem SCOOT

Sistem SCOOT (*Split Cycle and Offset Optimization Technique*) je danes razširjen po vsem svetu, najbolj pa v Veliki Britaniji. Glavna razlika med SCOOT in SCATS je v tem, da slednji nima sistema za optimiziranje krmilnih programov. Je eden najboljših sistemov za prometno odvisno krmiljenje križišč v mestih. Križišča, ki so krmiljena s sistemom SCOOT, so združena v več podobmočij. Vsako križišče iz istega podobmočja ima enako dolžino ciklusa. Osnova sistema je, da z manjšimi in pogostimi spremembami dolžin zelenih časov, zamikov ter dolžine ciklusa na predhodno izdelanih krmilnih programih zmanjšuje zamude in število ustavljanj. Naštete spremembe dela na podlagi aktualnega prometnega toka. Sam sistem ima svoj prometni model in optimizator. Sistem pri iskanju optimalne rešitve uporablja algoritem »hill climbing«. Prikaz delovanja sistema SCOOT je prikazan na sliki 2.5.

Koraki optimizacije so:

- sistem se po koncu zelene faze odloči, ali fazo podaljšati, skrajšati ali pustiti nespremenjeno;
- med potekom ciklusa se preračuna vpliv spremembe zamika glede na indeks učinka;
- vpliv trajanja ciklusa glede na indeks učinka se preračunava pri manjših obremenitvah na 5 minut oziroma ob velikih prometnih obremenitvah na 2,5 minute.

Prometni tokovi na obravnavanem križišču so podlaga za optimizacijo zelenih časov, medtem ko se za optimizacijo zamika upoštevajo vsi tokovi v bližini, ki prihajajo v križišče. Pri optimizaciji ciklusa je merodajno, ali nasičenost najbolj obremenjenega križišča v coni presega 90 %. Prometni podatki se pridobivajo iz detektorjev, ki so nameščeni na vsakem kraku posameznega križišča. Lokacija detektorjev je zelo pomembna in je navadno na začetku krakov. S tem je omogočeno pravočasno posredovanje podatkov ter zmanjšana možnost zasedbe detektorja, ki onemogoči napoved prometnega toka. Sistem SCOOT je primeren predvsem za mreže, pri katerih so križišča relativno blizu. Glavna prednost SCOOT-a je takojšen odziv na spremembe v prometu in s tem zmanjšanje zamud. Slabost pa predvsem v dragi investiciji opreme, težjem vzdrževanju in zapleteni nastavitvi sistema. Razvitih je bilo že več verzij. Med zadnjimi je možnost preskoka posameznih faz in s tem prednost javnemu potniškemu prometu ter pešcem (SCOOT, 2011).



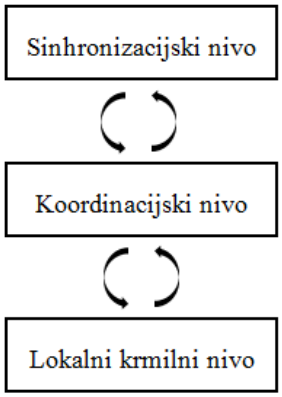
Slika 2.5: Prikaz delovanja sistema SCOOT (SCOOT, 2011)

Figure 2.5: How SCOOT works (SCOOT, 2011)

2.5.3 Sistem OPAC

Sistem OPAC (*Optimized Policies for Adaptive Control*) je namenjen optimizaciji krmilnih programov v realnem času. Med razvojem sistema je nastalo več verzij. Algoritem sistema, na podlagi DP, je bil sprva razvit za optimizacijo krmilnih programov za posamezna, izolirana križišča (OPAC I). Sistem določa optimalne točke preklopa med prvo in drugo fazo fiksnega krmilnega programa. OPAC pridobiva in uporablja prometne podatke iz postavljenih detektorjev na krakih križišča. Glavni cilj je zmanjševanje zamud in ustavljanj. Robni pogoji so določeni samo z minimalno in maksimalno dolžino posamezne faze. Dinamična optimizacija procesa poteka neprekinjeno tako, da se krmilni programi nenehno osvežujejo (Liao, 1998). Sčasoma se je sistem nadgrajeval in izboljševal. Gartner (2001) je predlagal zamenjavo pristopa DP in uvedel poenostavljen algoritem (OSCS – *Optimal Sequential Constrained Search*), ki je bil osnova za nove verzije. Z uporabo algoritma se je čas, potreben za izračun optimalnih programov, zmanjšal. Vendar so rezultati z uporabo OPAC II na ravni rezultatov, pridobljenih z DP. V primerjavi z DP je algoritem bolj zapleten. Za implementacijo sistema OPAC II je treba zadostiti določenim zahtevam. Optimizacija je razdeljena na časovne intervale. Intervali so dolgi od 50 do 100 sekund. Za vsak interval so dovoljene maksimalno tri zamenjave signalov. Sprememba signala z najmanjšo zamudo se shrani in uporabi kot optimalen rezultat za trenutno fazo. Pri delovanju v realnem času je primaren problem pridobitev točnih prometnih podatkov. Rešitev problema je uporaba metode potujočega horizonta. Verzijo, ki upošteva metodo potujočega horizonta, so poimenovali OPAC III. V kasnejši raziskavi je Gartner razvil še četrto generacijo sistema OPAC IV. Zadnja generacija ni omejena samo

na samostojna križišča, temveč je uporabna za cestne arterije in mreže. OPAC IV uporablja tehniko virtualnega fiksnega ciklusa (ang. *Virtual-Fixed-Cycle,* VFC). Sistem je razdeljen na tri nivoje (slika 2.6). Nivo za sinhronizacijo preračunava VFC s frekvenco nekaj minut. Glede na VFC nivo za koordinacijo optimizira zamike za vsako križišče. Lokalni krmilni nivo pa optimizira spremembe programa upoštevajoč parameter iz predhodnih nivojev. Kljub temu da je bila zadnja generacija testirana v realnem svetu, točnega procesa krmiljenja ni mogoče zaslediti v nobeni literaturi.



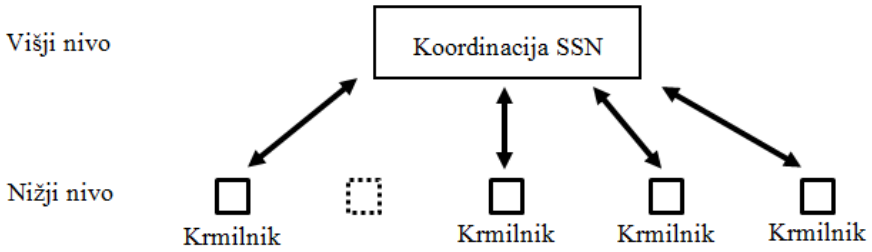
Slika 2.6: Nivojska struktura krmilnega sistema OPAC

Figure 2.6: Layer structure of OPAC system

2.5.4 Sistem PRODYN

Sistem PRODYN krmiljenja SSN je razvil Henry in spada v skupino prometno odvisnih sistemov krmiljenja mreže. Delovanje je zasnovano na podlagi DP. Tako kot ostali sistemi, ki temeljijo na DP, PRODYN nima fiksnega zaporedja faz, dolžine faz in dolžine ciklusa. Sistem uporablja pri krmiljenju mreže hierarhičen algoritem. Razviti sta bili dve verziji sistema. Prva ali začetna verzija ima hierarhično strukturo z dvema nivojema. Nižji nivo zajema krmiljenje križišča, medtem ko je višji nivo namenjen za koordinacijo arterije in mreže. Spodnji nivo sestavlja večje število krmilnikov križišč, ki generirajo začetne krmilne programe. Začetni krmilni programi so nato poslani na višji nivo. Tam se dodatno obdelajo in koordinirajo. Višji nivo nato zagotovi vsakemu križišču povratno informacijo, ki jo nato krmilnik uporabi za izboljšanje prvotnih krmilnih programov. Iterativni proces se izvaja, dokler se ne doseže ravnovesja med obema nivojema. Proces optimizacije sistema PRODYN je prikazan na naslednjem prikazu (slika 2.7). Hierarhična struktura se je v novejši verziji opustila. Začela se je uporabljati naprednejša metoda DP za posamezna križišča ter decentralizirano usklajevanje različnih krmilnikov. Sistem deluje podobno kot sistem OPAC. Optimizacija je razdeljena na veliko krajših časovnih intervalov. Vsak interval predstavlja določeno fazo. Stanja so

opisana kot trenutna faza in dolžina kolon. Odločitev v vsakem stanju je podaljšanje trenutne zelene faze ali preklop druge faze. Sistem prav tako kot OPAC uporablja metodo potujočega horizonta. Uporabljena metoda decentralizirane koordinacije sistema PRODYN poteka po korakih, in sicer: izbira križišča ter optimizacije, simulacija prometnih obremenitev, pošiljanje simuliranih prometnih obremenitev naslednjim križiščem ter premik do naslednjega križišča, izbira optimizacije trenutnega križišča glede na pridobljene obremenitve ter ponovno premik na drugi korak. Poglavitna težava procedure je primerna izbira prvega, začetnega križišča. Na dvosmernih cestah sotočno križišče lahko postane protitočno (Henry in Farges, 1990).



Slika 2.7: Nivojska struktura krmilnega sistema PRODYN

Figure 2.7: Layer structure of PRODYN system

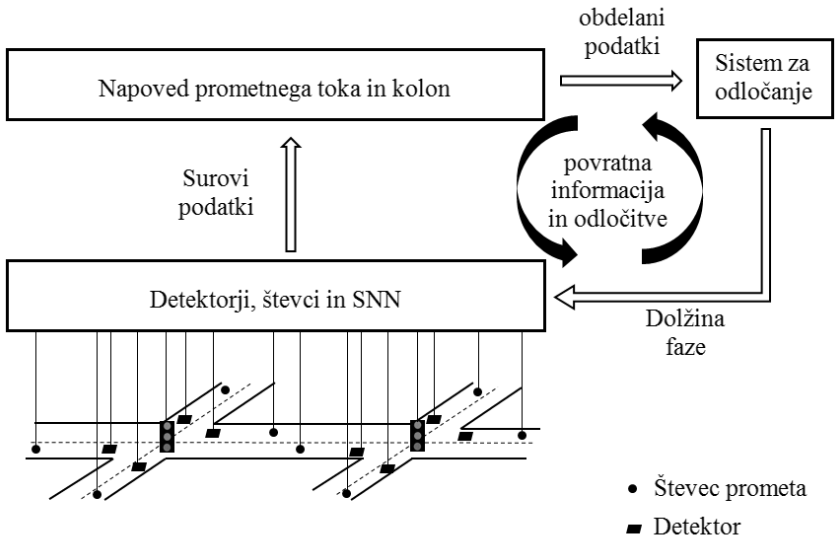
2.5.5 Sistem UTOPIA

Sistem UTOPIA (*Urban Traffic OPtimization of Integrated Automation*) sta leta 1989 predstavila Mauro in Di Taranto. Sistem je v uporabi v Italiji in na Nizozemskem. Je decentraliziran in hierarhično razdeljen na dva nivoja (področni nivo in nivo križišč). V prvem nivoju je krmilna strategija, ki se določa na podlagi napovedi prometa. Napoved se računa na podlagi vhodnih podatkov, pridobljenih s pomočjo detektorjev. Krmiljenje semaforjev je zasnovano na modelu, iz katerega sistem pridobiva prometne podatke. Ti podatki so delež zavijalcev, potovalni časi, gostota prometnega toka in dolžine kolon. Nadalje se podatke procesira z mikroskopskim modelom, na podlagi katerega se določa krmiljenje s pomočjo indeksa učinka v optimizacijskem modelu. Pri optimizaciji se poleg podatkov sosednjih križišč upošteva tudi minimalna in maksimalna dolžina ciklusa. Ker optimizacija poteka za celotno mrežo in ne samo na posamezna križišča, morajo potekati informacije o optimizaciji med vsemi križišči. Za celotno obravnavano področje se potovanja na glavnih poteh napovedujejo na področnem nivoju. Za napoved se uporablja makroskopski model. Vhodni podatki za makroskopski model predstavljajo količine, pridobljene z detektorji (Mauro in Di Taranto, 1989). Sistem je razvit za optimizacijo krmljenja z nudenjem delne ali popolne prednosti javnemu prometu brez podaljševanja potovalnih časov osebnih vozil. Na podlagi podatkov

izdelovalcev je z uporabo sistema mogoče prihraniti do 15 % potovalnih časov osebnih vozil ter 10 % pri emisijah in porabi goriva.

2.5.6 Sistem RHODES

Sistem RHODES (*Real-Time Hierarchical Optimized Distributed Effective System*) sodi med prometno odvisne signalne sisteme prav tako s hierarhično strukturo. Sestavljen je iz dveh osnovnih modulov, in sicer iz napovednega ter nadzornega. Prvi modul skrbi za napoved prometa, kdaj in koliko vozil bo prišlo v mrežo, medtem ko drugi za nadzor križišča in prometnega pretoka na cestni mreži. Arhitektura krmilnega sistema RHODES je prikazana na sliki 2.8. Določevanje krmilnih programov poteka na podlagi dveh algoritmov (REALBAND in COP – *Controlled Optimization of Phases*), ki sta bila razvita na osnovi DP. Prvi algoritem služi za ustvarjanje flot vozil na cestni mreži. Tako ustvarjen prometni tok uporabi drugi algoritem za določitev optimalne strategije krmiljenja na posameznih križiščih. Čeprav je algoritem COP zasnovan na DP, uporablja precej drugačne definicije za stanja v primerjavi s sistemoma DYPIC in OPAC. Pri algoritmu COP so stanja določena kot sekvence posameznih faz. Cilj optimizacije je doseči čim manjše zamude s čim manj ustavljanji in kar najbolj zmanjšati zajezitvene dolžine. Učinkovitost sistema RHODES je odvisna predvsem od napovedane natančnosti prometa za celoten časovni horizont, kar pa je v realnem svetu zelo težko doseči (Mirchandani in Head, 2001).



Slika 2.8: Arhitektura krmilnega sistema RHODES (Mirchandani in Head, 2001)

Figure 2.8: Architecture structure of RHODES system (Mirchandani and Head, 2001)

2.5.7 Sistem TUC

Sistem TUC (*Traffic-responsive Urban Traffic Control*) so razvili z namenom krmiljenja velikih prometnih mrež. Strategija koordinacije temelji na spremembi časovnih zamikov, dolžin ciklusov ali razdelitvah ciklusa oziroma vseh spremenljivk. Sistem daje prednost javnemu prevozu. Diakaki s sodelavci (2002) je predstavil rezultate raziskave v dveh scenarijih, in sicer za majhne in večje mreže v realnem svetu. Simulacije so izvedli za obe dnevni konici (jutranjo in popoldansko). Učinkovitost sistema TUC je bila v primerjavi s časovno odvisnim krmiljenjem boljša. Vendar primerjava s klasičnim prometno odvisnim krmiljenjem ni bila narejena. Pomanjkljivost sistema je v tem, da optimizacija programov poteka centralizirano. Ob konfliktih se težave rešujejo ročno ali z določenimi prednastavljenimi vrednostmi, kar je popolnoma v nasprotju z vidika VAS.

2.5.8 Sistem DYPIC

Robertson in Bretherton (1974) sta razvila metodo za optimalno krmiljenje DYPIC (*Dynamic Programmed Intersection Control*), zasnovano na DP za samostojno križišče. Avtorja sta predstavila metodo na enostavnem križišču s samo dvema konfliktnima manevroma, zato sta bili samo dve možnosti, in sicer podaljšanje ali zamenjava zelenega signala. V študiji sta predpostavila, da je v naslednjih nekaj minutah poznano točno število vozil, kar pa je v realnem svetu nemogoče. Zaradi tega je bila metoda DYPIC uporabljena samo na teoretični nivoju in za primerjavo z drugimi praktičnimi primeri krmiljenja SSN. Metoda krmiljenja DYPIC nima fiksne dolžine faz in fiksnega zaporedja faz, prav tako nima fiksne dolžine ciklusa. To je največja razlika v primerjavi s časovno odvisnim krmiljenjem, koordiniranim prometno odvisnim krmljenem, SCOOT in SCATS. Avtorja sta naredila analizo z dvema različnima prometnima obremenitvama. V primerjavi s časovno odvisnim krmiljenjem so se zamude zmanjšale za 50 %, povprečna zamuda na vozilo pa za 3 sekunde.

2.6 Pregled načinov krmiljenja

V poglavjih 2.3, 2.4 in 2.5 smo predstavili časovno in prometno odvisno krmiljenje oziroma sisteme krmiljenja. Poudarili smo prometno odvisne sisteme krmiljenja, ki so najbolj uveljavljeni in prepoznavni. Časovno odvisno krmiljenje ima fiksne dolžine ciklusov, razporede faz in čase trajanja posameznih faz. Ti sistemi nimajo možnosti prilagajanja na nihanja v prometnem toku. Primerni so za relativno konstantne prometne tokove. S prometno odvisnimi sistemi ta problem deloma rešimo s

pomočjo detektiranega prometa z detektorji. Na podlagi detekcije je možno podaljševanje trenutnih faz, sprememb dolžin ciklusov in zamikov.

Prometno odvisni sistemi krmiljenja se bolje prilagajajo spreminjajočim se prometnim razmeram v realnem času in precej zmanjšajo zamude v križiščih. Te sisteme lahko delimo v dve večji skupini. Sistema SCOOT in SCATS sta tipična sistema prve skupine. Preostale prilagodljive sistema pa lahko štejemo v drugo skupino. Prva skupina ima še vedno fiksne čase ciklusov, čase trajanja faz, zaporedje faz in zamikov. Krmilni sistemi prilagajajo parametre na podlagi prometnih podatkov, ki so pridobljeni v realnem času ali napovedi prometa. V izogib motenja normalnega delovanja se parametri osvežujejo v intervalu nekaj minut. Značilnost druge skupine krmiljenja je, da nima fiksnih ciklusov, fiksnega zaporedja faz in fiksnih zamikov. Vsi parametri so določeni v realnem času na podlagi trenutnih in napovedanih prometnih obremenitev. Vseeno pa imajo taki sistemi določene omejitve.

3 TEORETIČNA IZHODIŠČA

3.1 Umetna inteligenca

Izraz umetna inteligenca je prvič uporabil ameriški znanstvenik in pionir na tem področju John McCarthy. V osnovi lahko zapišemo, da umetna inteligenca (UI) skuša posnemati inteligentne subjekte. Področje UI lahko v današnjem času povežemo z različnimi vedami, kot so filozofija, psihologija, lingvistika, matematika, fizika, nevrologija itd. Od nastanka prvih računalnikov potekajo razprave med znanstveniki in filozofi, ali je možno ustvariti sistem, ki se bo obnašal inteligentno. Že na začetku se pojavi težava definicije UI, saj vsak človek razmišlja in si različno predstavlja UI. Eden izmed glavnih ciljev je najprej razumeti načela inteligence in nato izdelati stroj, ki se bo obnašal kot človek z upoštevanjem čustev in zavesti (Russell in Norvig, 2003). UI se uporablja na različnih področjih, med najpomembnejše vsekakor sodijo vizualna, govorna, manipulativna ter racionalna inteligenca.

Posebno vejo umetne inteligence predstavlja strojno učenje, ki je dandanes v uporabi na različnih področjih. S strojnim učenjem je mogoče pripraviti stroj, ki se sam uči in uporablja pridobljeno znanje. Poznamo več vrst pristopov oziroma metod reševanja strojnega učenja.
Najpomembnejše metode strojnega učenja so:

- nadzorovano učenje,
- nenadzorovano učenje in
- spodbujevano učenje (ang. *reinforcement learning*).

3.1.1 Spodbujevano učenje

SU se deli na tri področja. Prvi dve področji imata dolgo in bogato preteklost. Obe sta se neodvisno razvijali, preden se je razvilo t. i. moderno SU. Prvo področje oziroma metodologija poskusa in napake (ang. *trial and error*) se je začelo z raziskavami psihologije učenja živali. Raziskave na tem področju so med prvimi deli v UI in so ponovni zagon dobile v zgodnjih osemdesetih prejšnjega stoletja. Drugo področje se nanaša na optimalni nadzor, ki za rešitve problemov uporablja vrednostne funkcije in DP. Poleg omenjenih dveh področij se je kasneje razvilo še področje, ki se nanaša na metode učenja po časovnih razlikah (ang. t*emporal-difference*). Področji učenja po časovnih razlikah in optimalne nadzore je Chris Watkins leta 1989 združil in razvil učenje Q (ang. *Q-learning*). Velik

preboj in razpoznavnost sta se zgodila, ko je Gerry Tesauro predstavil reševanje igre »backgammon« s pomočjo SU.

Z metodo SU, ki povzema karakteristike različnih področij strojnega učenja, psihologije, teorije nadzora in nevroznanosti, se je razvilo nekaj uspešnih inženirskih aplikacij (Sutton in Barto, 1998). Uporaba SU se je povečala, ker se v inženirskih panogah uporablja čedalje več reševanja problemov s pristopom UI. Sprva je bila omejena le na logiko, simbole in ne na številke. V desetletjih se je uporaba postopoma spremenila. Sodobni algoritmi na osnovi UI predstavljajo pomembno alternativo pri reševanju raznovrstnih problemov. Prej prezrta področja med UI in konvencionalnim inženirstvom so sedaj bolj v uporabi, vključno s področji, kot so nevronske mreže, inteligentni nadzor in v našem primeru SU. Področje SU se je med drugim razvilo tudi zaradi enostavnega algoritma.

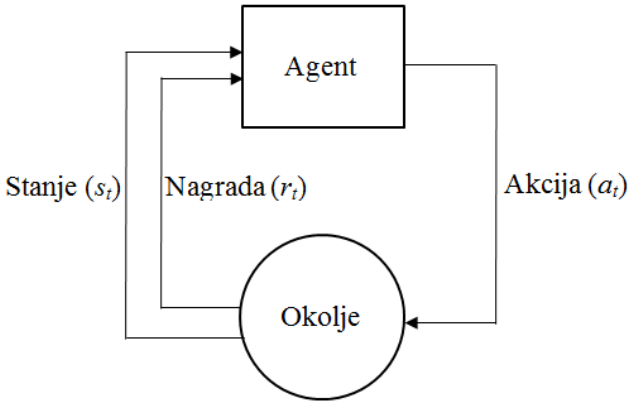
SU je metoda, s katero učenca naučimo izbrati optimalno akcijo v določenem okolju za dosego čim večje nagrade. V SU učenca navadno imenujemo agent. Vse, kar ni agent, pa imenujemo okolje. Na različnih področjih SU je določitev, kaj predstavlja agenta in kaj okolje, različna. Namesto da agentu povemo točno določene primere pričakovanih reakcij, mora agent raziskovati s poskušanjem pridobiti čim večjo nagrado. Nagrada pove agentu, katera akcija je zaželena. Funkcijo nagrade se uporablja za dosego zastavljenih ciljev (npr. čim krajša kolona). Večinoma se akcije ne odražajo samo na trenutno nagrado, temveč tudi na naslednje stanje, s tem pa na vse naslednje nagrade. SU ne potrebuje nobenih informacij o okolju. Od okolja dobi podatek, v kakšnem stanju je. Med procesom učenja agent poskuša z akcijami spremeniti stanje okolja. Pri tem ga spodbujajo nagrade, ki so lahko pozitivne ali negativne. Cilj učenca (agenta) je izbrati optimalno strategijo ali, drugače povedano, kar najbolj povečati skupno vsoto nagrad, ki jih pridobiva z učenjem (Mitchell, 1997).

Slika 3.1 prikazuje proces interakcije med agentom in okoljem. Interakcija se izvaja nepretrgoma in teoretično se agent lahko odloča za akcije kadar koli. Vendar se s praktičnega vidika navadno akcije izvajajo v diskretnih časovnih korakih. V nadaljevanju je predstavljen enostaven primer procedure delovanja v diskretnih časovnih korakih.

1. V časovnem koraku $t = 0$ agent dobi informacijo o stanju okolja $s_t \in S$. S predstavlja vsa možna stanja okolja.
2. Glede na $s_t \in S$ agent izbere akcijo $a_t \in A(s_t)$. $A(s_t)$ predstavlja vse možne akcije stanja s_t.

3. Z izbiro akcije $a_t \in A(s_t)$ v času t in z zaznanim novim stanjem okolja $s_{t+1} \in S$ dobi nagrado r_{t+1} v času $t + 1$.
4. Za učenje agenta uporabi s_t, a_t, s_{t+1} in r_{t+1} in
5. Postavimo $t = t + 1$ in se vrnemo na 2. korak.

V vsakem časovnem koraku sistem prejme informacijo stanja okolja s_t, na podlagi katere izbere akcijo a_t. V naslednjem koraku zazna spremembo okolja s_{t+1} in prejme nagrado r_{t+1}. Prejeta nagrada agentu pove uspešnost izvedene akcije. Povezavo med stanji in akcijami imenujemo strategija (Abdulhai in Kattan, 2003).



Slika 3.1: Osnovni prikaz interakcije v spodbujevanem učenju

Figure 3.1: Interaction between agent and environment

Uporaba SU je primerna, ko ne poznamo okolja ali ko je okolje znano, pa z analitičnim pristopom ne moremo do rešitve. Metode reševanja problemov s SU so se izkazale na področju robotike (Kohl in Stone, 2004), računalniških iger (Tesauro, 1995), ekonomije (Moody in Saffell, 2001), optimizacije delovanja dvigal (Crites in Barto, 1998) in tudi na področju krmiljenja prometa (Abdulhai et al., 2003), (Bazzan, 2005), (Prashanth in Bhatnagar, 2011) in drugi.

3.1.1.1 Elementi spodbujevanega učenja

V SU imamo poleg agenta in okolja še naslednje elemente: funkcijo nagrade v povezavi z vrednostno funkcijo, strategijo in opcijsko lahko še model okolja. Za delovanje sistema na podlagi SU je treba definirati še stanja okolja in akcije agenta.

Ob krmiljenju prometa agenta predstavlja krmilnik svetlobnosignalnih naprav (SSN), okolje pa vse ostalo, kot so npr. prometne obremenitve, trenutno stanje signalov, kolone na posameznih krakih križišč, prometno mrežo itd. Agent je zadolžen za izvajanje akcij na podlagi zaznanih stanj okolja in prejetih spodbud oziroma nagrad. Za agenta ni nujno, da pozna model okolja, na katerega vpliva, mora pa biti sistem tako zasnovan, da agent zazna spremembe v okolju. Poleg tega mora agent imeti cilj ali cilje, ki so povezani s stanji okolja. Stanja okolja se v povezavi z izvedenimi akcijami spreminjajo in opisujejo spremembe okolja. Akcije predstavljajo vse možne izbire agenta v posameznem koraku. V vsakem koraku ima agent na razpolago najmanj dve možni akciji. Z akcijami je določeno, kaj lahko agent v danem koraku stori za spremembo stanja okolja.

Funkcija nagrade definira cilj problema SU. Poenostavljeno, pretvori vsako stanje (ali par stanje-akcija) okolja v število. Cilj agenta je dobiti na dolgi rok čim večjo nagrado. Funkcija nagrade definira, katere akcije so dobre in katere slabe. Služi tudi kot osnova za spremembo strategije. Npr. če na podlagi strategije izbrani akciji sledi nizka nagrada, potem se lahko pričakuje v prihodnje sprememba strategije oziroma izbira druge akcije. Generalno gledano je lahko funkcija nagrade tudi stohastična. Medtem ko funkcija nagrade pove, kaj je dobro v tem trenutku, nam vrednostna funkcija določa, kaj je dobro na dolgi rok. Vrednost stanja je skupna vrednost nagrad, ki jih lahko agent pridobi v prihodnje. Npr. za določeno akcijo lahko dobi nizko nagrado, vendar ima še vedno visoko vrednostno funkcijo, saj lahko nadaljnja stanja prinesejo visoko nagrado in obratno. Nagrade so primarne, medtem ko so vrednostne funkcije sekundarne. Iščemo akcije, ki prinašajo stanja z najvišjimi vrednostmi in ne z največjimi nagradami, zato ker nam take akcije prinašajo na dolgi rok največjo vsoto nagrad. Pri odločanju in načrtovanju je določitev vrednostne funkcije najtežja. Bistveno lažja naloga je določitev nagrad. Nagrade so v osnovi določene neposredno iz okolja, medtem ko se vrednostne funkcije ocenjuje iz zaporedja agentovega opazovanja v vsem času delovanja. Pravzaprav je najpomembnejši sestavni del algoritma SU določitev vrednostne funkcije.

Strategija določa vedenje agenta oziroma katere akcije bo izbral pri zaznavi sprememb okolja. V nekaterih primerih je strategija lahko enostavna funkcija ali preglednica s podatki, medtem ko je v drugih primerih lahko vključeno obsežno računanje, kot je npr. proces iskanja.

3.1.1.2 Raziskovanje in izkoriščanje

Eden od izzivov pri SU, ki ga ni mogoče zaslediti pri drugih vrstah učenja, je iskanje ravnotežja med raziskovanjem (ang. *exploration*) in izkoriščanjem (ang. *exploitation*). Za največjo nagrado mora agent izkoriščati pridobljeno znanje in/ali raziskovati druge izbire, ki morebiti vodijo do boljših

rezultatov. Da bi odkril take akcije, mora poizkusiti akcije, za katere se še ni odločil. Samo kombinacija raziskovanja in izkoriščanja vodi do optimalne rešitve. Agent mora poizkušati različne akcije in postopoma izbirati take, ki dajo boljši rezultat. Npr. samo raziskovalni pristop lahko predpiše kazni (negativne nagrade), ki jo mora agent tudi upoštevati. V primeru stohastičnih procesov se mora vsaka akcija izbrati večkrat, da lahko dobimo zanesljivo oceno o pričakovani nagradi.

Pri SU se za določitev ravnotežja med izbiro raziskovanja in izkoriščanja lahko uporablja požrešno ε-metodo (ang. *ε-greedy*), ki je najpogosteje uporabljen pristop. Pri požrešni ε-metodi se učenec večino časa vede izkoriščevalno, in sicer izbira akcije, ki prinašajo največjo nagrado, in le redkokdaj (z majhno verjetnostjo ε) izbere akcijo naključno. Prednost požrešne ε-metode je, da z naraščanjem števila poskusov narašča tudi število vseh obiskanih stanj in s tem povezanih odločitev. Verjetnost izbire optimalne akcije je večja kot 1 – ε. Slabost požrešne ε-metode je, da izbira enako med vsemi akcijami, ne glede na dodeljeno nagrado za posamezno akcijo. Verjetnost, da izberemo najslabšo varianto, je enaka verjetnosti, da izberemo naslednjo od najboljše. Za zmanjševanje verjetnosti ponovne izbire slabših odločitev se pri simulacijah, ki potekajo v realnem času, daje prednost akcijam, za katere je agent bolje nagrajen (Sutton in Barto, 1998).

3.1.2 Osnovni pristopi reševanja spodbujevanega učenja

Za reševanje problemov SU ter izbire optimalne strategije obstajajo tri osnovne metode, in sicer dinamično programiranje, metoda Monte Carlo in metoda učenja po časovnih razlikah (ang. *temporal difference learning*). DP je metoda planiranja, ki ima model okolja, medtem ko sta drugi dve metodi učni in se učita samo iz izkušenj brez uporabe modela okolja.

Vse tri metode imajo tako pozitivne kot negativne lastnosti. Metode DP so zelo dobro matematično razvite, vendar potrebujejo popoln in točen model okolja. Metode Monte Carlo ne potrebujejo modela okolja in so relativno enostavno zasnovane, vendar niso primerne za izračun po korakih. Zadnja metoda, učenje po časovnih razlikah, prav tako ne potrebuje modela okolja in je primerna za računaje po korakih, vendar je za analizo veliko kompleksnejša. Metode se med seboj razlikujejo tudi glede zmogljivosti in hitrosti konvergence.

3.1.2.1 Metoda učenja po časovnih razlikah

Metode UČR so algoritmi učenja za dolgoročne napovedi dinamičnih procesov (Perez, 1998). So inkrementalne procedure učenja, ki združujejo osnove metode DP in metode Monte Carlo. Tako kot pri metodah Monte Carlo se pri UČR-metodah agent lahko nauči neposredno iz izkušenj, brez poznavanja okolja. Tako učenje lahko uporabimo tudi v realnem svetu. Metode UČR imajo podobnost z DP v sprotnih posodobitvah, brez čakanja na končni izid, medtem ko metode Monte Carlo za določitev prirastka vrednostne funkcije stanja $V(s_t)$ čakajo do konca procesa. Metode UČR v času $t + 1$ takoj določijo cilj in z uporabo nagrade r_{t+1} posodobijo vrednostno funkcijo stanja $V(s_{t+1})$. Osnovno obliko enačbe UČR zapišemo kot:

$$V(s_t) = V(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)], \qquad (3.1)$$

kjer je α faktor stopnje učenja. Metode UČR imajo prednost pred metodami DP, saj so zmožne poiskati optimalno strategijo samo z izkušnjami, brez znanega modela okolja, torej le z nagradami in vrednostmi prihodnjega stanja. V primerjavi z metodami Monte Carlo, ki čakajo na posodobitev do konca epizode, se pri metodah UČR posodobitve izvajajo v vsakem koraku (Sutton in Barto, 1998). Za reševanje problemov z metodo UČR poznamo dva algoritma, in sicer algoritem SARSA in algoritem učenja Q. Algoritem učenja Q je primeren za reševanje problemov, ko imamo veliko število možnih stanj in ne poznamo modela okolja.

3.1.2.2 Učenje Q

Razvoj učenja Q kot algoritma UČR-metode je eden izmed največjih napredkov v SU. Kot smo že zapisali, se pri učenju Q izvaja interakcija med agentom in okoljem. Za vsak prehod med stanji se zapisujejo spremembe oziroma posodobitve vrednostne funkcije akcije $Q(s_t, a_t)$. V matriko se vpisujejo vrednosti $Q(s_t, a_t)$ za vsako dvojico (stanje-akcija). Ob prehodu iz stanja s_t v s_{t+1}, izbiri akcije a_t in dodeljeni nagradi r_{t+1} algoritem opravi naslednjo posodobitev $Q(s_t, a_t)$:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_{a+1} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right], \qquad (3.2)$$

kjer je α faktor stopnje učenja in γ faktor diskontiranja.

Faktor α ($0 < \alpha \leq 1$) določa, koliko bodo novopridobljene informacije vplivale na izkušnje. Vrednost faktorja stopnje učenja ne sme biti nič, saj se v tem primeru agent ne nauči ničesar. Ob izbiri faktorja

$\alpha \approx 0$ se vrednost $Q(s_t, a_t)$ spremeni zelo malo. Z izbiro $\alpha = 1$ bo agent kot svoje izkušnje upošteval le zadnjo pridobljeno informacijo. Če se faktor stopnje učenja manjša na izbrani način z vsakim časovnim korakom in se vsi pari stanj-akcij poskusijo neskončno pogosto, potem učenje Q konvergira k optimalni funkciji Q (Wiering in sod. 2004). V primeru stohastičnega učenja Q pa je lahko faktor stopnje učenja tudi konstanta.

Faktor diskontiranja γ ($0 \leq \gamma < 1$) določa pomembnost (potencialnih) prihodnjih nagrad. Vrednost 0 pomeni, da bo agent upošteval samo trenutno nagrado. Ob izbiri vrednosti blizu 1 bo agent stremel k dolgoročno višji nagradi. Kar pomeni, da bo na novo vrednost $Q(s_t, a_t)$ imela večjo težo vrednost $Q(s_{t+1}, a_{t+1})$.

V nadaljevanju so predstavljeni koraki algoritma učenja Q:
Inicializacija Q (s_t, a_t).
Informacija s_t v $t = 0$.
Za vsako epizodo ponovi:

Izvedi akcijo a_t glede na s_t z uporabo določene strategije (npr. požrešna ε).
Sprememba stanja okolja v s_{t+1} in pridobitev nagrade r_{t+1}.
Sprememba vrednosti Q (s_t, a_t).
$s_t \leftarrow s_{t+1}$.

Dokler s_t ni končno stanje.

Navadno imamo na začetku postopka učenja vrednosti vseh parov stanj-akcij enake 0. Temu pravimo inicializacija matrike Q in pomeni, da agent(i) nimajo nobenega znanja. Če želimo na začetku podati določeno znanje, je lahko matrika Q napolnjena z določenimi vrednostmi, ki so različne od nič. Podrobneje bomo oba primera predstavili v poglavju 5.

Učenje Q pri iskanju optimalne strategije deluje podobno kot iterativna metoda DP (Pendrith, 2000). Učenje Q je algoritem »off-policy« in pridobiva koristne izkušnje tudi med raziskovalnimi akcijami, za katere je možno, da se v nadaljevanju ne izkažejo za najboljše. V primerjavi z algoritmom SARSA so začetni rezultati v fazi učenja pri učenju Q slabši.

4 RAZVOJ ALGORITMA KRMILJENJA

V prejšnjih poglavjih smo predstavili klasične in novejše alternative metode. Iz pregledanih del lahko povzamemo, da so rezultati alternativnih metod boljši od klasičnih časovno in prometno odvisnih sistemov krmiljenja. Razvitih je bilo veliko novih metod, predvsem za samostojna križišča, v novejšem času je opaziti povečano število takih, ki so sposobne krmiliti tudi večje število križišč. Slednje nas je spodbudilo k razvoju stohastičnega algoritma učenja Q za krmiljenje SSN na cestni arteriji. Algoritem je predstavljen v tem poglavju. V naslednjem poglavju so predstavljeni rezultati primerjave s klasičnim prometno odvisnim krmiljenjem.

Uporaba novejših metod na podlagi UI in pristopi z večagentnimi sistemi predstavljajo večji izziv pri krmiljenju cestnih arterij in mrež. Tukaj imamo še posebej v mislih križišča, ki morajo delovati sinhronizirano, kar pomeni usklajevanje med agenti. Vsi agenti morajo svoje akcije tako prilagajati, da dosežejo učinkovitost celotnega območja, ne samo lokalnega. Krmiljenje več križišč hkrati lahko upravljamo samo z enim agentom, vendar je navadno v tem primeru opis stanj preveč kompleksen.

Kot smo zapisali v poglavju 2.2, ni nujno, da imamo sinhronizacijo med SSN na arteriji v obeh smereh. O smeri sinhronizacije in fazi s prednostjo se je mogoče odločati na več načinov.

Prva možnost je neposredna komunikacija med SSN, kar dandanes zaradi komunikacijskih naprav ni ovira (npr. brezžična tehnologija). Z uporabo take tehnologije je mogoča izmenjava krmilnih podatkov prek senzorjev. S senzorji pridobivamo informacije o stanju okolja, ki si jih nato agenti med seboj izmenjujejo s komunikacijo. Vendar sama izmenjava ni zadostna. Prvič, velika količina podatkov nam ne pomaga, če ne moremo vseh podatkov obdelati v realnem času. Drugič, pojavi se vprašanje, zakaj bi agenti žrtvovali lokalno uspešnost v prid globalni. Iz filozofskega stališča se lahko vprašamo, koliko so ti agenti še avtonomni, če morajo slediti vodilnemu agentu, ki uravnava sinhronizacijo (hierarhičnost).

Druga možnost je, da ne uporabimo komunikacije med agenti in preprosto prepustimo agentom, da reagirajo ali se prilagodijo okolju samostojno (de Oliveira in sod., 2004). Taki pristopi ne uporabljajo neposredne komunikacije med agenti in prav tako ne določajo vodilnega agenta. V tako dinamičnem okolju, kot je promet, se nam zato lahko zgodi, da agent ne reagira dovolj hitro. Pri čakanju na spremembe nima dovolj časa za prilagoditev. Za določeno reakcijo se lahko izkaže, da je že prepozna, ker so se med tem časom pogoji spremenili.

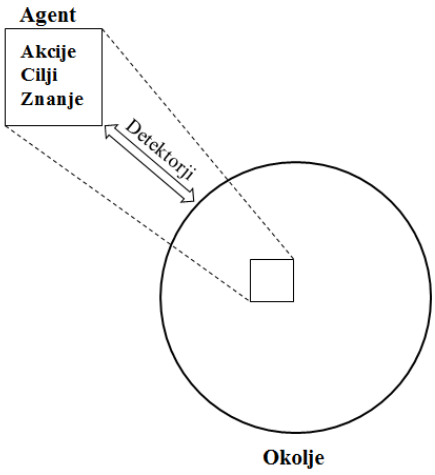
Za potrebe razvoja algoritma so v nadaljevanju opisani vsi osnovni elementi SU, ki smo jih našteli v predhodnem teoretičnem poglavju. Osnovni elementi, agenti, okolje, stanja, akcije, nagrade in način izbiranja akcij so posebej definirani za primer krmiljenja SSN na cestni arteriji.

4.1 Enoagentni in večagentni sistemi

Preden začnemo s predstavitvijo in kategorizacijo VAS, moramo najprej upoštevati najočitnejšo alternativo VAS, in sicer centralizirane enoagentne sisteme (EAS). Centralizirani sistemi imajo agenta, ki se odloča, medtem ko se drugi akterji vedejo podrejeno agentu. EAS lahko imajo več subjektov ali več akterjev. Drugače povedano, če vsak akter pošlje svoje percepcije ter nato dobi iz centralnega dela ukaze, potem je to EAS ali centralni proces (Stone in Veloso, 2000).

4.1.1 Enoagentni sistemi

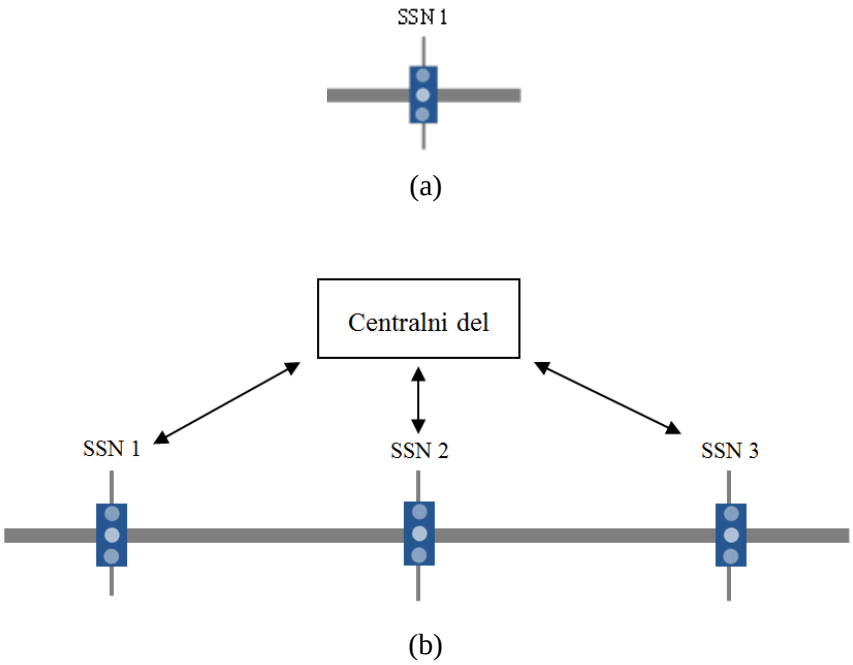
Agent v EAS je samostojen subjekt, ki ima svoje akcije, cilje in znanje. V EAS je agent samo eden. Izvaja akcije na podlagi informacij iz okolja. Agenta predstavlja lahko samo ena naprava (v našem primeru samo ena SSN) ali pa skupek več naprav. Torej tudi v slednjem primeru, ko več naprav pošilja svoje zaznavanje enemu centralnemu delu, ki izbira akcije, govorimo o EAS. Slika 4.1 prikazuje shemo EAS.



Slika 4.1: Zgradba enoganetnega sistema (Stone in Veloso, 2000)

Figure 4.1: Single-agent system framework (Stone and Veloso, 2000)

Slika 4.2 prikazuje shematski prikaz EAS ob krmiljenju prometa. Zgornji del prikaza (a) predstavlja EAS v samostojnem križišču, spodnji del (b) pa več križišč. Cestno omrežje predstavljajo sive črte. Agenti so na sliki predstavljeni kot modre SSN. V primeru samostojnega križišča je en sam agent, ki izvaja akcije, ima svoje cilje in znanje. V več križiščih v sistemu imamo več akterjev, vendar ti ne izvajajo akcij samostojno, temveč ukaze akcij pošilja centralni del. Centralni del krmiljenja je lahko nameščen v bližini krmiljenih naprav ali dislociran v nadzornem centru. V konkretnem primeru krmiljenja so možne akcije centralnega dela (v primeru več križišč), podaljšanje ali zamenjava faze. Podatki iz nameščenih detektorjev prometa se pošiljajo v krmilni del SSN, od tam pa v centralni del. Komunikacija poteka samo med SSN in centralnim delom.



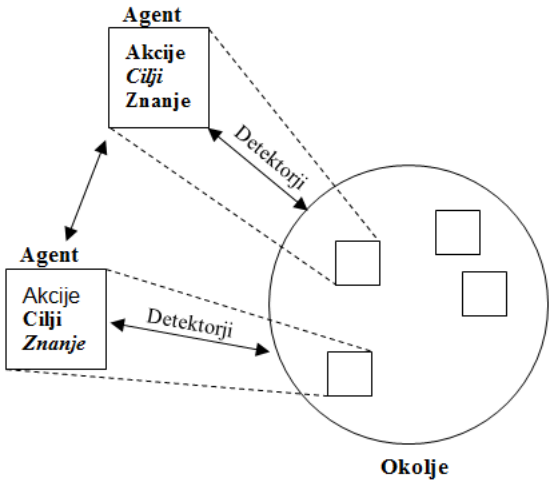
Slika 4.2: Zgradba enoagentnega sistema – krmiljenje prometa v samostojnem križišču (a) ali več križiščih (b)

Figure 4.2: Single-agent system framework – traffic control, in case of single intersection (a) or more intersections (b)

4.1.2 Večagentni sistemi

Pri scenariju z VAS lahko obstajajo neposredne interakcije (komunikacija) med posameznimi agenti. Iz perspektive individualnega agenta se VAS najbolj razlikujejo od EAS v tem, da lahko stanje agentovega okolja določajo drugi agenti. Drugi agenti vplivajo na okolje nepredvidljivo. Slika 4.3 prikazuje, da je vsak agent obenem del okolja in modeliran kot samostojni subjekt. Različni cilji, akcije in znanje agentov so na sliki predstavljeni v različni pisavi. V takih sistemih je lahko poljubno število različnih agentov s sposobnostmi neposredne komunikacije med seboj ali brez njih. Na sliki je

komunikacija prikazana s puščicami. VAS lahko razčlenimo še na homogene in heterogene ter s komunikacijo ali brez nje.

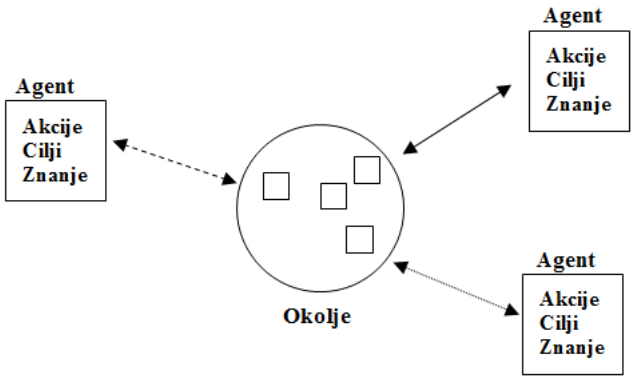


Slika 4.3: Zgradba večagentnega sistema (Stone in Veloso, 2000)

Figure 4.3: Multiagent system framework (Stone and Veloso, 2000)

4.1.2.1 Homogeni VAS brez komunikacije

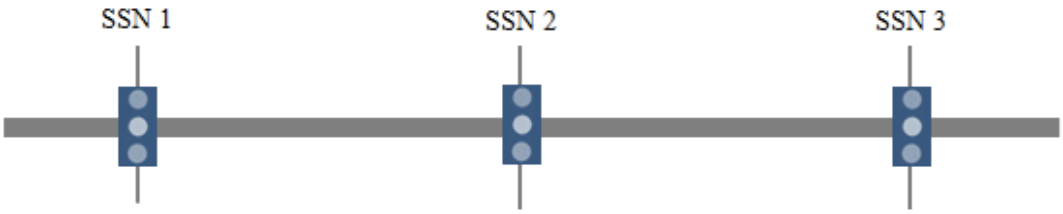
Pri homogenih VAS brez komunikacije imajo vsi agenti enako interno strukturo, vključno s cilji, znanjem in možnimi akcijami. Tudi proceduro oziroma strategijo izvajanja akcij imajo enako. Edina razlika med agenti so njihovi detektorji okolja in aktualne izbire akcije. Razlikujejo se zaradi različnih lokacij v prostoru. Tako ne morejo napovedati akcij sosednjih agentov. Agenti brez komunikacije ne morejo neposredno vplivati drug na drugega. Ker pa so del istega okolja, lahko vplivajo posredno, npr. s spremembo stanja okolja drugega agenta. Struktura homogenih VAS brez komunikacije je prikazana na naslednji sliki 4.4. Različni detektorji okolja so na sliki prikazani kot različne puščice. Enaki cilji, akcije in znanje so prikazani z enako pisavo.



Slika 4.4: Homogeni VAS brez komunikacije (Stone in Veloso, 2000)

Figure 4.4: Non-communicating homogeneous MAS (Stone in Veloso, 2000)

Slika 4.5 prikazuje shematski prikaz homogenega VAS brez komunikacije ob krmiljenju prometa. Sive črte na sliki predstavljajo cestno omrežje. Glavno smer predstavlja debelejša črta, medtem ko so stranske smeri označene s tanjšimi črtami. Enake SSN so prikazane v enaki modri barvi. Za tak tip krmiljenja ni nujno, da si križišča sledijo v sosledju, lahko tvorijo tudi drugo obliko prometne mreže. Agente predstavljajo SSN 1, 2 in 3, ki imajo enako interno strukturo. Nabor akcij je pri vseh agentih enak, je pa lahko njihova izbira akcij v istem trenutku različna. Možne akcije vseh SSN so lahko podaljšanje trenutnega zelenega signala ali zamenjava faze. V takem primeru imajo vse SSN dvofazni krmilni program. Prav tako velja, da morajo biti cilji vseh SSN enaki, npr. favoriziranje glavne smeri v križiščih. Faza glavne smeri je daljša od faze stranske smeri. Vse SSN morajo imeti nameščene detektorje prometa na vseh krakih (polno prometno odvisne naprave). Razlike so v lokacijah v prostoru, različne imajo lahko tudi detektorje prometa (zančni, video, mikrovalovni ...). Med SSN ne poteka nobena komunikacija, s čimer je onemogočen neposreden vpliv. Če so križišča relativno blizu, lahko SSN posredno vplivajo med seboj s spremembo stanja druge SSN (pojav daljših kolon, večji pretok prometa). Taka zgradba sistema je primerna za krmiljenje prometa za križišča, ki so si geometrijsko podobna in imajo med seboj podobne prometne obremenitve.

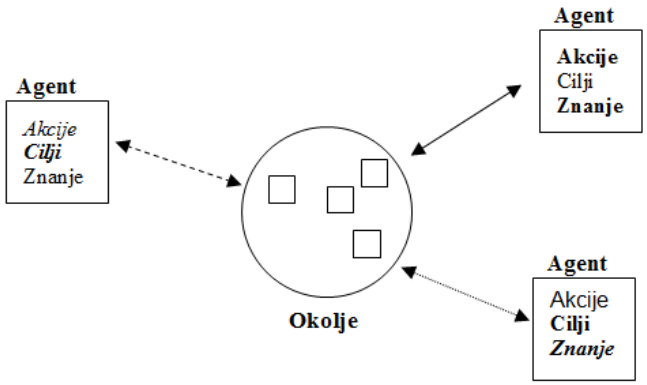


Slika 4.5: Homogeni VAS brez komunikacije – krmiljenje prometa

Figure 4.5: Non-communicating homogeneous MAS – traffic control

4.1.2.2 Heterogeni VAS brez komunikacije

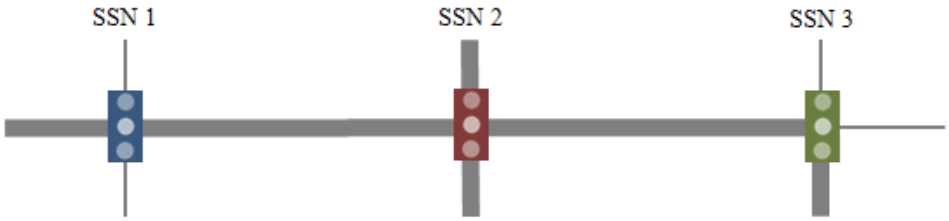
Heterogeni agenti imajo različne cilje, akcije in/ali znanje. Tako kot pri homogenih sistemih so agenti v prostoru različno locirani. To pomeni, da z detektorji zaznavajo različne informacije, kar se odraža v različnih akcijah. Struktura heterogenih VAS brez komunikacije je prikazana na sliki 4.6. Različni cilji, akcije in znanje so na sliki označeni kot različne pisave. Enako velja za detektorje, ki so prikazani z različnimi puščicami.



Slika 4.6: Heterogeni VAS brez komunikacije (Stone in Veloso, 2000)

Figure 4.6: Non-communicating heterogeneous MAS (Stone in Veloso, 2000)

Heterogeni VAS brez komunikacije ob krmiljenju prometa je shematsko prikazan na naslednji sliki 4.7. Cestno omrežje je predstavljeno na sliki s sivimi črtami. Različne SSN so predstavljene v različnih barvah semaforjev. Za križišča ni nujno, da si sledijo v sosledju, lahko tvorijo tudi drugo obliko prometne mreže. Agente predstavljajo SSN, ki imajo različno interno strukturo. Nabor akcij agentov ni nujno enak. SSN imajo lahko različne krmilne programe (dvo- ali večfazne). Nabor možnih akcij (podaljšanje trenutnega zelenega signala, zamenjava faze, preskok faze) se med agenti lahko razlikuje. Nekateri agenti imajo lahko več možnih akcij, drugi manj. Vsaka SSN v omrežju ima lahko drugačne cilje. Npr. SSN 1 lahko favorizira glavno smer zahod–vzhod, SSN 2 sever–jug in SSN 3 zahod–jug. Znanje je pri taki strukturi sistema lahko pri vseh SSN različno. Vsak SSN v sistemu ima nameščeno različno število detektorjev vozil. Nameščeni detektorji so lahko pri vsakem križišču različni. Med SSN ni predvidena komunikacija in s tem je onemogočen neposredni vpliv med SSN. Posredni vpliv med SSN se lahko pojavi, če so križišča relativno blizu. Možna je sprememba stanj drugih SSN ob daljših kolonah, večjem pretoku vozil itd. Tako različne SSN so predvidene za večje geometrijske razlike med križišči ter križišča z različnimi prometnimi obremenitvami. S takim sistemom je težko doseči sinhronizacijo med SSN.

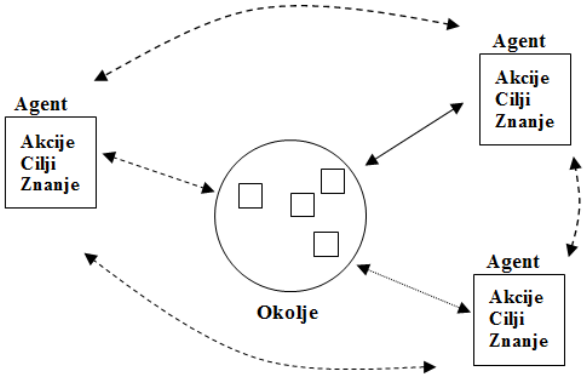


Slika 4.7: Heterogeni VAS brez komunikacije – krmiljenje prometa

Figure 4.7: Non-communicating heterogeneous MAS – traffic control

4.1.2.3 Homogeni VAS s komunikacijo

Tako kot pri homogenih VAS brez komunikacije so tudi pri homogenih VAS s komunikacijo agenti z enako interno strukturo, vključno s cilji, znanjem in možnimi akcijami. Neposredno komunikacijo med agenti predstavljajo puščice (slika 4.8). Enaki cilji, akcije in znanje so na sliki označeni z enako pisavo. Komunikacija med agenti lahko poteka hkrati med vsemi agenti, lahko pa samo med posameznimi pari agentov.

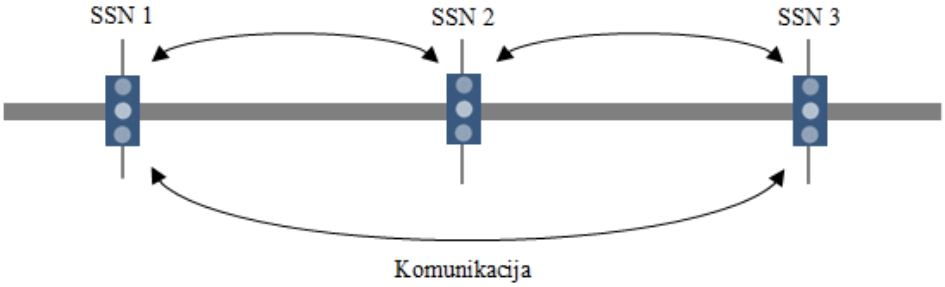


Slika 4.8: Homogeni VAS s komunikacijo (Stone in Veloso, 2000)

Figure 4.8: Communicating homogeneous MAS (Stone and Veloso, 2000)

Slika 4.9 prikazuje homogeni VAS s komunikacijo ob krmiljenju prometa. Sive črte na sliki predstavljajo cestno omrežje. Debelejše črte predstavljajo glavno smer, tanjše pa stranske smeri. Enake SSN so prikazane v enaki modri barvi. Agente predstavljajo SSN z enako interno strukturo. Izbira akcij agentov je v istem trenutku lahko različna, vendar je nabor možnih akcij pri vseh agentih enak. Agenti lahko izbirajo med podaljšanjem zelenega signala ali zamenjavo faze. Pri tem morajo imeti vse SSN dvofazni krmilni program. Cilji vseh SSN se med seboj ne razlikujejo, npr. favoriziranje glavne smeri zahod–vzhod. V homogenem VAS s komunikacijo mora biti znanje vseh

agentov oziroma SSN enako. Število detektorjev prometa mora biti enako na vsek križiščih. Edine razlike med SSN so lokacije v prostoru in tipi detektorjev prometa (zančni, video, mikrovalovni ...). Med SSN je predvidena komunikacija in s tem onemogočen neposredni vpliv med SSN. Informacije med komunikacijo SSN so lahko zelo različne: trenutna faza agenta, trajanje trenutne faze, število detektiranih vozil, kolone vozil na posameznih smereh itd. Komunikacija med agenti je dobrodošla pri ohranjanju sinhronizacije SSN, nudenje prednosti javnemu potniškemu prometu ali vozilom na nujni vožnji. Homogeni VAS s komunikacijo so primerni za križišča, ki so si geometrijsko podobni ter imajo med seboj podobne prometne obremenitve.

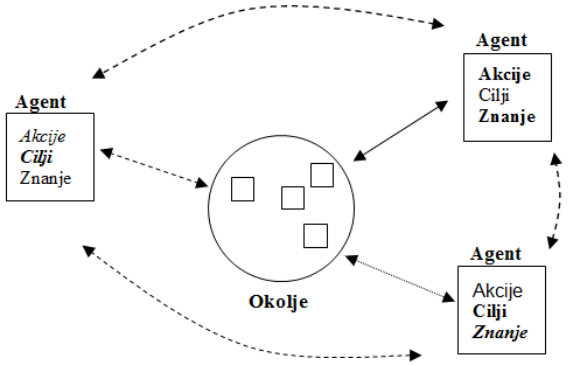


Slika 4.9: Homogeni VAS s komunikacijo – krmiljenje prometa

Figure 4.9: Communicating homogeneous MAS – traffic control

4.1.2.4 Heterogeni VAS s komunikacijo

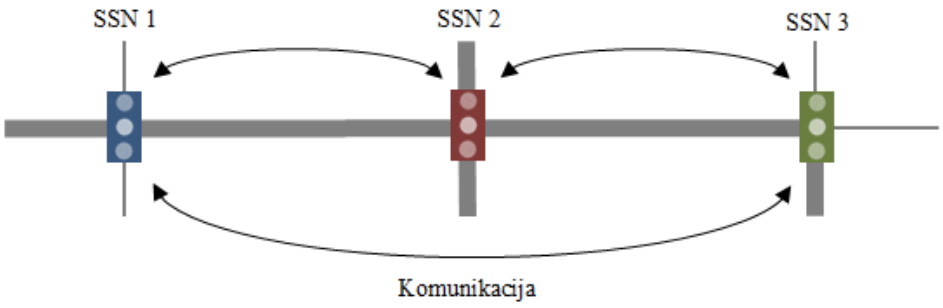
Slika 4.10 prikazuje heterogeni VAS s komunikacijo. Taki sistemi imajo različne cilje, akcije in znanja, ki so na sliki prikazani v različni pisavi. Agenti so lahko zelo kompleksni in učinkoviti. Mnogo večjo učinkovitost pa dosežemo z vključeno komunikacijo med njimi. Tako kot pri homogenih VAS s komunikacijo lahko tudi v tem primeru komunikacija poteka med vsemi agenti hkrati, lahko pa je omejena med posameznimi pari. Z združitvijo komunikacije in heterogenih agentov pridobimo možnost, da se VAS spremenijo v sistem, ki je ekvivalenten EAS.



Slika 4.10: Heterogeni VAS s komunikacijo (Stone in Veloso, 2000)

Figure 4.10: Communicating heterogeneous MAS (Stone and Veloso, 2000)

Slika 4.11 prikazuje shemo heterogenega VAS s komunikacijo ob krmiljenju prometa. Različne sive črte predstavljajo glavne in stranske ceste omrežja. Raznolikost med SSN je predstavljena z različnimi barvami. Agente z različno interno strukturo predstavljajo semaforji. Pri taki strukturi so možne različne akcije. Semaforji se razlikujejo med seboj glede različnih krmilnih programov (dvo- in večfazni). Nabor možnih akcij se med agenti lahko razlikuje. Agenti imajo lahko dve ali več možnih akcij (npr. podaljšanje trenutnega zelenega signala, zamenjava faze, preskok faze). Cilji so lahko različni za vsak semafor. Npr. SSN 1 lahko favorizira glavno smer zahod–vzhod, SSN 2 sever–jug in SSN 3 zahod–jug. Znanje SSN je prav tako različno. Zmožnost prepoznave vozil je pri vsakem SSN različno, prav tako število detektorjev vozil. Detektorji vozil so lahko za vsako križišče različni. Med SSN poteka komunikacija, pretok informacij, ki omogoča neposredni vpliv med SSN. Podatki med SSN so lahko: trenutna faza agenta, trajanje trenutne faze, število detektiranih vozil, kolone vozil na posameznih smereh itd. Sistem s tako strukturo je treba upoštevati na cestah, kjer imamo križišča z večjimi geometrijskimi razlikami ter z različnimi prometnimi obremenitvami.



Slika 4.11: Heterogeni VAS s komunikacijo – krmiljenje prometa

Figure 4.11: Communicating heterogeneous MAS – traffic control

Razvoj in preizkušanje algoritma smo izvedli na cestni arteriji s tremi semaforiziranimi križišči. Vsaka SSN predstavlja enega agenta. Predpostavili smo, da ima vsak agent enako interno strukturo, vključno s cilji, znanjem in možnimi akcijami. Zaradi tega je naš VAS homogen. V fazi preizkušanja algoritma nismo omogočili nikakršnih informacij med agenti. V tem primeru smo imeli homogeni VAS brez komunikacij. Kasneje smo preizkusili vedenje algoritma ob informacijah med agenti, torej s komunikacijo. Informacije smo podajali o stanju sosednjega, t. i. vodilnega agenta. Kot vodilnega agenta smo za vsako od SSN izbrali sosednjo SSN. Najprej smo podali informacijo o fazi vodilnega agenta, ki smo jo označili s f_{va}. Nato smo informaciji o fazi vodilnega agenta dodali še informacijo o trajanju zelena signala vodilnega agenta, ki smo jo označili s t_{va}. Informaciji o fazi in trajanju zelenega signala vodilnega agenta predstavljata dodatna stanja agenta. Več o možnih stanjih agentov je opisano v poglavju 4.3 Stanja.

4.2 Okolje

Okolje zajema vse prometne in geometrijske elemente prometne arterije. Prometna infrastruktura zajema v našem testnem primeru tri križišča, ki si sledijo v sosledju tako, da skupaj tvorijo cestno arterijo. Vsa križišča so štirikraka, na katerih so dodatni pasovi za leve zavijalce. Le severni kraki so brez dodatnega pasu za leve zavijalce. Na vseh križiščih oziroma vseh krakih so dovoljeni vsi manevri (levo, naravnost, desno). Razdalja med križišči je cca. 450 m. Vsa tri križišča so opremljena s SSN z dvofaznimi krmilnimi programi. Glavna faza predstavlja smer vožnje po arteriji, in sicer vzhod–zahod. Druga faza omogoča vožnjo vozilom iz severa in juga. Fazi se izmenjujeta tako, da je zagotovljena varnost. Med prvo in drugo fazo smo za izpraznitev križišča upoštevali rdeči signal v vseh smereh in sicer tri sekunde. V raziskavi smo upoštevali samo zeleni in rdeči signal. Okolje smo v raziskavi nadomestili s prometnim simulatorjem. Uporabljeni prometni simulator upošteva vse vozne karakteristike vozil, ki ponazarjajo realno stanje v prometnem toku, vključno s hitrostmi, pospeški, pojemki, varnostno razdaljo itd. V simulaciji so upoštevani različni tipi vozil oziroma voznikov, in sicer od zmernih do agresivnejših. S tem je zagotovljen boljši približek realnemu stanju v prometu. Podrobneje je delovanje izbranega prometnega simulatorja opisano v naslednjem poglavju 5.

4.3 Stanja

Krmiljenje SSN na cestni arteriji s pomočjo učenja Q, modeliranega kot EAS, ima dva osnovna problema. Prvi izhaja iz večanja števila križišč. Če gledamo vsa stanja skupaj, nam število stanj eksponentno narašča.

Da bi zmanjšali množice možnih stanj, smo se odločili za VAS in upoštevali vsako križišče posebej. Za potrebe raziskave smo v nadaljevanju analizirali primere brez komunikacije med agenti in z njo. Problem večanja števila stanj se pojavi, če so za posameznega agenta pomembna vsa stanja v sistemu, če pa opazujemo stanja z vidika enega agenta, ki se uči samo na osnovi svojega križišča, tega problema nimamo. Zato smo stanja opisali tako, da se je njihovo osnovno število (enega križišča) razlikovalo samo zaradi komunikacije med sosednjimi agenti in ni bilo pogojeno z večanjem števila križišč. S tem smo postavili osnovo algoritmu, ki je sposoben krmiliti večje število križišč.

4.3.1 Stanja brez komunikacije

Za potrebe algoritma smo stanja določili kot štiridimenzionalni vektor.
Dimenzije vektorja predstavljajo:

- dolžina kolone v glavni smeri,
- dolžina kolone v prečni smeri,
- trenutna faza semaforja in
- trajanje zelenega signala.

Z upoštevanjem dolžine kolon na meter natančno in trajanje zelenega signala v sekundah, se število možnih stanj hitro povečuje. Pri tako velikem porastu možnih stanj, kot to lahko zasledimo v delu Abdulhai in sod. (2003), je učenje agenta neučinkovito. Jin in sod. (2009) so raziskovali velikost razmikov pri speljevanju v semaforiziranem križišču. V raziskavi so ugotovili, da daljša kot je kolona vozil, krajši je razmak med vozili pri speljevanju. Povprečne vrednosti razmakov med vozili pri speljevanju se vrednosti do petega vozila (cca. 30 m) zmanjšujejo, nato pa so vrednosti podobne. Avtorji so v raziskavi upoštevali devet vozil (cca. 60 m) v koloni. Avtorji Li in Prevedouros (2002) ter Prashanth in Bhatnagar (2011) v svojih raziskavah obravnavajo kratke, srednje in dolge kolone. Tudi v naši raziskavi smo število stanj omejili z definiranjem treh možnih dolžin kolon. Tako smo upoštevali kratko, srednjo in dolgo kolono. Kratka meri 30 m ali manj, srednje dolga več kot 30 m in do 60 m ter dolga kolona več kot 60 m. Za dodatno zmanjšanje števila stanj smo razdelili trajanje

zelenega signala na razrede. Trajanje zelenega signala agenta smo razdelili na 12 razredov in pri tem upoštevali, da je en razred dolg 8 sekund. S tem smo dobili število vseh možnih stanj brez komunikacije med agenti za eno križišče 216 (3 × 3 × 2 × 12). Dolžino razreda trajanja zelenega signala smo določili poskusno. Dolžino razredov smo spreminjali med 4 in 24 sekundami. Na podlagi analize rezultatov, pridobljenih z različnimi dolžinami razredov se je izkazala dolžina razreda 8 sekund kot najbolj primerna. Z omejitvami na tri dolžine kolon in razdelitev trajanja zelenega signala na razrede smo se izognili neprimerno večjemu številu možnih stanj za posamezno križišče.

4.3.2 Stanja s komunikacijo

V primeru VAS s komunikacijo med agenti imamo možnost podajanja dodatnih informacij o stanjih sosednjih agentov. Vendar se z vključeno komunikacijo med agenti število možnih stanj, ki smo jih predhodno opisali, dodatno poveča. Ob komunikaciji smo pri opisu stanj analizirali še vedenje algoritma z dvema dodatnima informacijama o stanju vodilnega agenta. Kot vodilnega agenta smo za vsakega od semaforjev določili sosednji semafor na zahodni strani. Tako smo agentom podali še informacijo o fazi vodilnega agenta (f_{va} = {0,1}). Informacija je bila zeleni ali rdeči signal. Druga informacija, ki smo jo dodali in analizirali, je bila trajanje zelenega signala vodilnega agenta (t_{va}). Z upoštevanimi dodatnimi informacijami o vodilnem agentu smo posamezno stanje opisali s šestdimenzionalnim vektorjem.

Dimenzije vektorja predstavljajo:

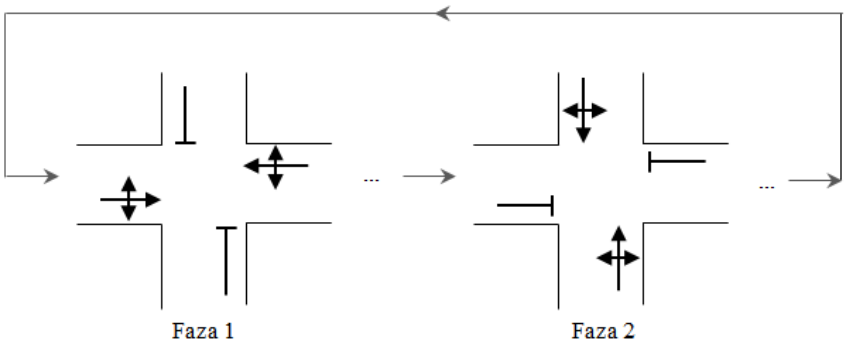
- dolžina kolone v glavni smeri,
- dolžina kolone v prečni smeri,
- trenutna faza semaforja,
- trajanje zelenega signala,
- faza vodilnega agenta in
- trajanje faze vodilnega agenta.

Da bi se izognili velikemu številu možnih stanj in da bi se agenti bolje in lažje učili, smo tudi pri komunikaciji upoštevali tri možne dolžine kolon. Za trajanje zelenega signala agenta smo upoštevali 12 razredov. Dodatno smo trajanje zelenega signala vodilnega agenta razdelili na 4 razrede. Trajanje posameznega razreda zelenega signala vodilnega agenta smo določili na 24 sekund (t_{va} = {1 do 4}). Dolžino razreda trajanja zelenega signala smo določili poskusno. Na podlagi analize rezultatov, pridobljenih z različnimi dolžinami razredov se je izkazala dolžina razreda zelenega signala vodilnega agenta, 24 sekund kot najbolj primerna. Tako je z upoštevanimi omejitvami in dodatnimi

informacijami o vodilnem agentu število vseh možnih stanj za eno križišče znašalo 216 × | f_{va}| × |t_{va}| oziroma 216 × 2 × 4 = 1728.

4.4 Akcije

Možni akciji agenta(-ov) sta podaljšanje trenutne faze ali njena zamenjava (preklop zelenega signala v drugi smeri). Agenti se odločajo o podaljšanju ali zamenjavi faze v diskretnem definicijskem območju časa, in sicer na 4 sekunde, a ne pred potekom minimalnega trajanja zelenega signala (10 sekund). Omejitev trajanja minimalnega zelenega signala smo podali zaradi zagotavljanja prometne varnosti, ki je odvisna od geometrije križišča in udeležencev v prometnem toku. Upoštevali smo dvofazni krmilni program s spremenljivim semaforskim ciklusom, kot je prikazan na sliki 4.12.



Slika 4.12: Upoštevan dvofazni krmilni program algoritma

Figure 4.12: 2-phase signal sequence considered by the algorithm

4.5 Nagrade

V našem primeru smo za funkcijo nagrade izbrali dolžine kolon na krakih križišč. Za kolone in ne za drugo mero uspešnosti smo se odločili, ker se kolone lahko enostavno merijo na terenu in ker si ne želimo imeti takih zajezitvenih dolžin, da segajo od križišča do sosednjega križišča. Take situacije lahko z lahkoto kaznujemo. Z uporabo kolon za nagrade lahko enostavno preprečimo situacije, ko vozila stojijo na celotni dolžini med križišči. Vrednost funkcije nagrade smo označili z negativnim predznakom. To pomeni, da daljša kot je kolona, manjša je nagrada. Nagrajevanje agentov smo preverili z dvema tipoma nagrad, in sicer z globalno in lokalno. Analizirali smo, kateri tip nagrade povzroči boljše in hitrejše učenje agentov. Boljše učenje agentov pomeni krajše kolone ter manjše zamude v križišču.

Ob globalni nagradi predstavlja nagrado vsota dolžin vseh kolon na vseh križiščih. Pri takem tipu nagrade so agenti v bistvu nagrajeni kot skupina. V tem primeru se nam lahko zgodi, da agent pri določeni izbiri akcije na račun drugih agentov ne dobi primerne nagrade. Poleg tega agent pri tem ne ve, kaj je naredil narobe, in se v naslednjem poizkusu odloči za drugačno odločitev. To nam daje slutiti, da je za dosego optimalnega krmiljenja ob uporabi globalnega nagrajevanja treba vključiti komunikacijo med agenti, ki smo jo opisali v predhodnih poglavjih.

Ko uporabimo lokalno nagrajevanje, je nagrada samo vsota dolžin kolon na krakih križišča, ki ga krmili agent. Z lokalnimi nagradami so agenti nagrajeni kot posamezniki in ne več kot skupina.

Funkcijo nagrade zapišemo:

$$R_{t+1} = \sum_i q_{i\,(t+1)} w_i \ , \tag{4.1}$$

kjer je q vrednost dolžine kolon, w utež posameznega pasu.

V poglavju 4.3 Stanja je zapisano, da smo upoštevali tri tipe oziroma dolžine kolon. Za vse tri definirane dolžine kolon (kratka, srednje dolga in dolga) smo definirali vrednost zajezitvene dolžine q. Izbrane vrednosti q za posamezen prometni pas so:

$$q_i = \begin{cases} 0, & l_i \leq 30 \\ 0{,}2, & 30 < l_i \leq 60 \\ 0{,}4, & 60 < l_i \end{cases}, \tag{4.2}$$

kjer je l_i dolžina kolone na prometnem pasu i.

Cilj učinkovitega krmiljenja cestne arterije, so čim manjše zamude na glavni prometni smeri, saj so navadno na arterijah oziroma vpadnicah bistveno večje prometne obremenitve na glavnih smereh. Favoriziranje glavnega prometnega toka oziroma smeri smo dosegli z uvedbo uteži, ki smo jo definirali kot $w_i = \begin{cases} 0{,}6, i \in L_m \\ 0{,}4, i \in L_s \end{cases}$, L_m množica prometnih pasov na glavni smeri in L_s množica prometnih pasov na prečno-stranski smeri. Ker smo v nadaljnjih preizkusih želeli preveriti vedenje algoritma tudi ob bolj uravnoteženih prometnih obremenitvah, tj. ob večjih spremembah v prometnem toku ali ob krmiljenju prometnih mrež, smo uteži na glavnih in prečnih smereh enačili. Tako smo upoštevali $w_i = \begin{cases} 0{,}5, i \in L_m \\ 0{,}5, i \in L_s \end{cases}$. Vrednosti uteži w_i smo povzeli po raziskavi Prashanth in Bhatnagar (2011).

4.6 Način izbiranja akcij

Pri učenju Q je treba zagotoviti ustrezno stopnjo agentovega raziskovanja okolja. Z izbiranjem različnih akcij v danih stanjih se agent uči in stremi k optimalni strategiji. Z ustrezno stopnjo raziskovanja dosežemo, da rezultati konvergirajo k optimalnim vrednostim oziroma obiščemo vsa potencialna stanja. Če agent ne raziskuje ali preneha raziskovanja prezgodaj, se lahko zgodi, da ne dosežemo globalnega optimuma. Po prenehanju raziskovanja agent začne samo z izkoriščanjem svojega znanja, ki ga je pridobil predhodno. Izkoriščanje pomeni, da agent izbere akcijo, ki prinaša največje vrednosti funkcije Q. Za določitev optimalnega razmerja med raziskovanjem in izkoriščanjem obstajajo različne metode. V naši raziskavi smo uporabili požrešno ε-metodo ki je med drugim tudi najbolj razširjena (Abdulhai in sod., 2003).

V parametrični analizi smo parameter ε variirali med 10 in 40, kar pove, v koliko odstotkih se bo agent odločil za raziskovanje in ne za izkoriščanje svojega znanja. Poleg parametra ε smo analizirali še parameter n_ε, ki pove, po koliko obiskih posameznega stanja se agent preneha učiti in začne uporabljati samo svoje znanje. Parameter n_ε smo analizirali med vrednostmi 200 in 800 s korakom 200, pri čemer smo želeli poiskati najhitrejše konvergiranje rezultatov k optimalnim vrednostim. Poleg hitrosti konvergence smo analizirali tudi čim manjše nihanje rezultatov, predvsem v sklepnih iteracijah, ko agent preneha preizkušanje in začne izkoriščati samo znanje.

4.7 Formalizacija algoritma

Da bi se še bolj približali realnemu dogajanju na prometni mreži, smo osnovno enačbo učenja Q nekoliko modificirali in uvedli stohastični algoritem učenja Q. Stohastični algoritem učenja Q smo uvedli, ker iz nekega stanja pri isti odločitvi ni nujen vedno povratek v isto stanje. Konkretno v našem primeru izbira določene odločitve lahko povzroči kolono na določenem kraku ali pa te kolone ni. Zaradi tega se mora v stohastičnih procesih vsaka akcija izbrati večkrat, saj lahko samo s tem dobimo zanesljivo oceno o pričakovani nagradi. Vrednost kriterijske funkcije v prihodnjem koraku zapišemo:

$$Q\,(s_t, a_t) \leftarrow\ Q\,(s_t, a_t) + \ \alpha\left[r_{t+1} + \ \gamma\, Q_p\,(s_{t+1}, a_{t+1}) - Q\,(s_t, a_t)\right] \qquad (4.3)$$

z upoštevanjem

$$Q_p\ (s_{t+1}, a_{t+1}) = \frac{\sum_j \left[n\left(s_t, s_{t+1,j}, a_t\right) \times \max_{a_{t+1,i}} Q\ (s_{t+1,j}, a_{t+1,i}) \right]}{n\ (s_t, a_t)}, \qquad (4.4)$$

kjer je $n\left(s_t, s_{t+1,j}, a_t\right)$ število prehodov agenta iz stanja s_t v stanje $s_{t+1,j}$ pri akciji a_t in $n\ (s_t, a_t)$ je število vseh obiskov oziroma vseh že izvedenih akcij v stanju s_t.

Pričakovana vrednost novega stanja je verjetnost, da prideš v to stanje pomnoženo z maksimalno vrednostjo tega stanja.

Predlagan stohastični algoritem učenja Q za krmiljenje cestne arterije smo predstavili v nadaljevanju v obliki psevdokode.

```
Vhodni parametri:

        - I_Mreža
        - I_Inicializacija (da/ne)
        - I_ŠtPonovitev (št. iteracij)
        - I_Sekund (trajanje ene iteracije)
        - I_Seme (seme za naključno generiranje prometa)
        - I_TipUčenja (Q_learn, Stohastic Q_learn, SARSA)
        - I_TipIzbora (e-greedy, Bolzman)
        - I_TipNagrade (kolona, volumen prometa)
        - I_Epsilon (funkcija, parametri funkcije ...)
        - I_Alfa (funkcija, parametri funkcije ...)
        - I_Gama (funkcija, parametri funkcije ...)
        - I_InfoOLeaderju (nič, faza, faza in trajanje faze)
        - I_Lokalni_reward (da/ne)
        - I_KorakSimulacije (v večini simulacij 4 sekunde)

Algoritem:
procedura zagon
begin

        if I_Inicializacija Then
                Inicializiraj Q ... (tj. brez znanja)
        else
                Uporabi predhodno pridobljeno znanje
        end if

        Za vsako Iteracijo od 1 do I_ŠtPonovitev
        begin

                ' ena simulacija prometa v dolžini I_Sekund
                        inicializacija prometa (I_Seme)
```

```
                        inicializacija semaforjev
                        preberi stanja agentov
                        čas = 0
                        {resnično se izvede še 3 sekunde simulacije na začetku}
                        dokler je Čas < I_Sekund
                        begin
                                izbor akcij agentov (agent. stanje, I_Epsilon)  ' akcija 1 =
                                zamenjava faze

                                za vsakega agenta: if agent.akcija = 1 then
                                agent.PrižgiVseRdeče

                                simuliraj 3 sekunde '3 sekunde so semaforji na križišču/agentu
                                rdeči)
                                čas = čas + 3

                                za vsakega agenta: if agent.akcija = 1 then agent.ZamenjajFazo

                                preberi nova stanja agentov

                                za vsakega agenta: dodeli LokalnoNagrado

                                GlobalnaNagrada = seštevek LokalnihNagrad vseh agentov

                                za vsakega agenta : Posodobi Q(agent.staroStanje, agent.akcija,
                                agent.novoStanje, I_Alfa, I_Gama, LokalnaNagrada ali
                                GlobalnaNagrada)

                                simuliraj (I_KorakSimulacije - 3) sekund
                                čas = čas + I_KorakSimulacije - 3
                        end

                ' KONEC ENE SIMULACIJE PROMETA
                zapiši rezultate (rezultati so - zamude, št. vozil, hitrost ...)

        end
                shrani pridobljeno znanje

end

opombe:
- LokalnaNagrada je utežena vsota kolon; utež je 0,6 glavna in 0,4 stranska ali 0,5 za vse
- vrednost posamezne kolone (kazen/nagrada) je 0 za [0-30] m, 0,2 za (30-60], 0,4 za (60-
...)
```

5 NUMERIČNI EKSPERIMENTI IN VREDNOTENJE REZULTATOV ALGORITMA S POMOČJO MIKROSIMULACIJSKEGA MODELA

V tem poglavju je predstavljeno vrednotenje rezultatov predlaganega algoritma s pomočjo mikrosimulacijskega modela. Primerjali smo rezultate mikroskopskih simulacij krmiljenja prometa s predlaganim algoritmom in rezultate klasičnega prometno odvisnega krmiljenja. Analizo smo izvedli na cestni arteriji treh križišč v enakih prometnih razmerah za oba primera krmiljenja. Pred samo primerjavo zmogljivosti in učinkovitosti predlaganega algoritma smo izvedli še analize, kako posamezni parametri vplivajo na učenje ter hitrost konvergence.

Poglavje začenjamo s splošnim opisom mikrosimulacijskega orodja, ki je predstavljal naš simulator, ter s ponujenimi prednostmi. V nadaljevanju bomo predstavili metodologijo simuliranja z algoritmom učenja Q in metodologijo klasičnega prometno odvisnega krmiljenja. Sledi prikaz analize parametrov in konvergence algoritma ter nato še primerjava krmiljenja z učenjem Q in klasičnim prometno odvisnim krmiljenjem. Na koncu je predstavljen še povzetek rezultatov primerjave.

5.1 Mikrosimulacijski model

Kot nadomestilo realnega sveta oziroma okolja smo uporabili mikroskopski prometni simulator. Mikroskopska računalniška simulacija prometa je eno od pomembnih orodij, ki omogoča analizo in preverjanje novih predlaganih prometnih rešitev. Med drugim je mogoče preveriti prihodnje strategije krmiljenja SSN. Z mikrosimulacijskimi orodji lahko preverimo nove prometne alternative še pred implementacijo in se pri tem izognemo morebitnim negativnim učinkom na terenu. Mikroskopske simulacije so kot metode veliko uporabnejše kot katere koli empirične in/ali analitične metode za izračune in/ali prometno opazovanje na terenu, še posebno zaradi naslednjega:

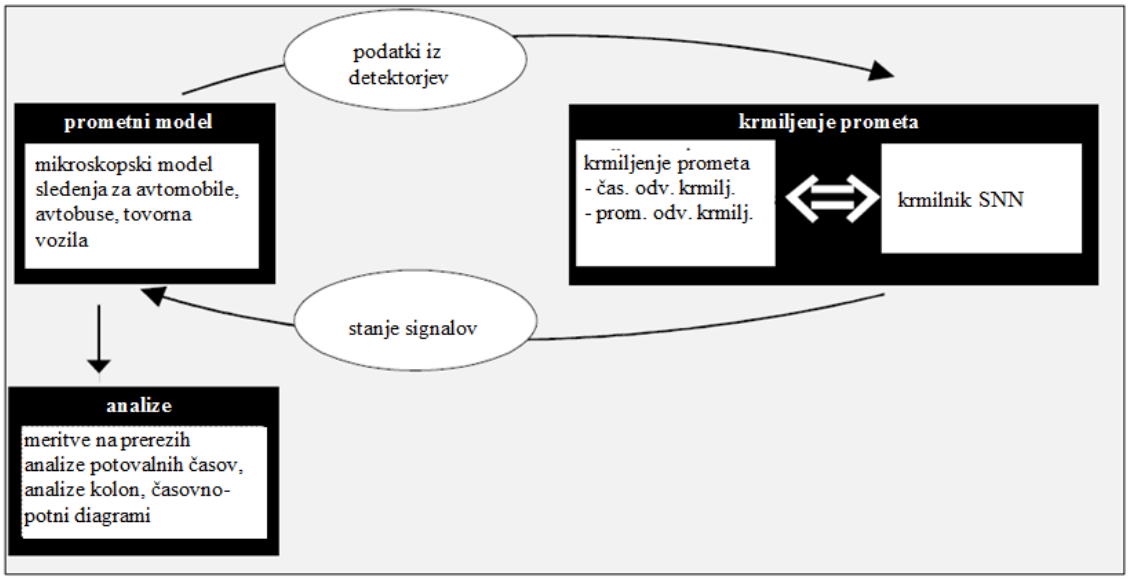
- Nižji stroški. Testiranje krmiljenja prometa z mikroskopskim orodjem je bistveno lažje kot testiranje na terenu v realnem svetu. Pri simulacijah odpadejo vsa gradbena dela, kot je inštalacija strojno-komunikacijske opreme, postavitve detektorjev itd.
- Hitrost. Rezultati so na razpolago v neprimerno krajšem času, saj lahko mikrosimulacijski model mreže izdelamo v bistveno krajšem času. V virtualnem okolju lahko hitrost simuliranega obdobja poljubno pohitrimo. Prihranek je odvisen samo od zmogljivosti strojne opreme.

- Varnost. S simulacijo lahko varno testiramo krmilne sisteme, še preden bi se morebitni negativni učinki ali celo nesreče zgodile na terenu. Oviranje prometa, ki se mu med terenskimi meritvami ali ob spremembi prometnega režima ni mogoče izogniti, v virtualnem okolju prometa povsem odpade.
- Prilagodljivost. Prometnim simulacijam se lahko sprotno spreminja parametre v odvisnosti od želenih namenov ali zastavljenih ciljev. Možno je postopno izključevanje slabih variant do izbire najustreznejše. S simulacijo je mogoče z zadovoljivo natančnostjo napovedati, kakšno bo dogajanje v prometu ob novih ukrepih in prometnih režimih. Na terenu je to veliko težje, če ne že nemogoče.
- Nadzor. Z uporabo enakega naključnega semena lahko testiramo prometne analize različnih krmilnih strategij v enakih prometnih razmerah. Medtem ko je v realnem svetu skoraj nemogoče ponoviti enako prometno stanje. Med različnimi časovnimi obdobji ni mogoče pričakovati enakih prometnih razmer. Sklepi na podlagi rezultatov so pogostokrat vprašljivi ravno zaradi razlike med prometnimi razmerami (Gartner in sod., 2001).

Za simulacijo prometa na mikro ravni obstaja veliko uveljavljenih in dobro preizkušenih orodij. Med bolj razširjenimi in uporabljanimi so VISSIM, SimTraffic, Paramics, AIMSUN in CORSIM. Narejenih je bilo veliko študij in primerjav med posameznimi simulacijskimi orodji, vendar v nobeni ni enotnega mnenja, kateri program najbolje ponazarja realno stanje v prometu (Birst in sod., 2007). Za analiziranje, vrednotenje in testiranje delovanja algoritma smo se odločili za simulacijsko orodje VISSIM predvsem zaradi naslednjih razlogov:

- S programskim orodjem VISSIM je možno enostavno primerjati zdajšnje metode krmiljenja z novopredlaganimi. VISSIM je eden izmed najbolj uveljavljenih in najbolj uporabljanih mikroskopskosimulacijskih programov na svetu.
- Paket VISSIM vsebuje urejevalnik VisVAP, s katerim je možno programirati prometno odvisne signalne naprave.
- Dostop do podatkovnega in simulacijskega modela je možen prek vmesnika COM, ki omogoča, da upravljamo s programom VISSIM iz drugih aplikacij (VISSIM COM, 2011).

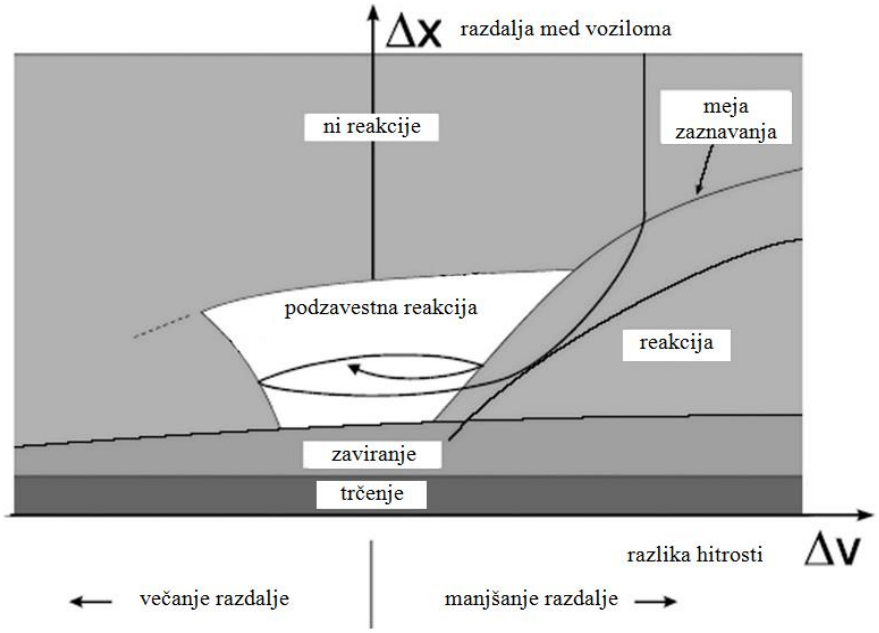
Simulacijski paket VISSIM je sestavljen iz dveh delov, prometnega modela in generatorja signalnih stanj. Ta dva dela si izmenjujeta podatke o stanju detektorjev v eni smeri in ukaze za spremembo signalov v drugi smeri, kot je prikazano na sliki 5.1. Generator signalnih stanj pridobiva informacije iz detektorjev v diskretnih časovnih korakih. Nato določi stanje signalov za naslednji časovni korak in vrne informacijo prometnemu simulatorju (VISSIM, 2011).



Slika 5.1: Komunikacija med generatorjem signalnih stanj in prometnim simulatorjem (VISSIM User Manual, 2011)

Figure 5.1: Communication between signal state generator and traffic simulator (VISSIM User Manual, 2011)

Kakovost prometnega modela je bistvenega pomena za izdelavo kakovostne simulacije. V nasprotju s poenostavljenimi modeli, ki uporabljajo konstantne hitrosti in deterministično logiko sledenja vozil, VISSIM uporablja psiho-fizični model zaznavanja, ki ga je razvil Wiedemann 74. Osnovni koncept modela je, da voznik hitrejšega vozila začne zavirati na kritični oddaljenosti od počasnejšega, pred sabo vozečega vozila. Ker voznik ni zmožen določiti točne hitrosti počasnejšega, pri zaviranju pade njegova hitrost pod hitrost spredaj vozečega vozila. Nato voznik pospeši, dokler ponovno ne doseže kritične razdalje med voziloma. To povzroča iterativni proces med zaviranjem in pospeševanjem oziroma osciliranjem razmaka med vozili. Vožnja se razlikuje od voznika do voznika. To je v VISSIMU upoštevano z različnimi distribucijami hitrosti in upoštevanjem varnostne razdalje med vozili. Grafičen prikaz Wiedemannovega modela sledenja vozil je na sliki 5.2.

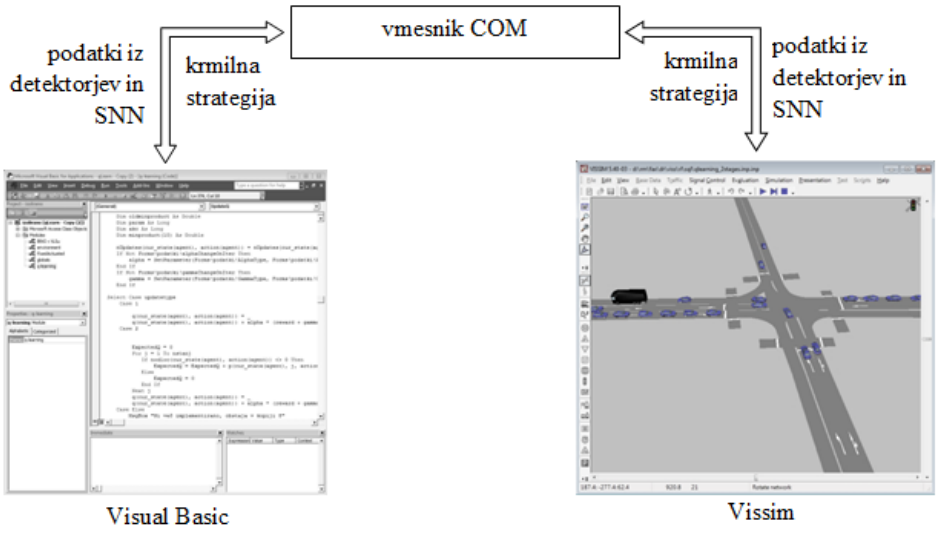


Slika 5.2: Wiedemannov model sledenja vozil (VISSIM User Manual, 2011)

Figure 5.2: Car following model by Wiedemann 1974 (VISSIM User Manual, 2011)

5.1.1 Implementacija algoritma učenja Q

Logiko algoritma učenja Q smo napisali v programskem jeziku Visual Basic in prek vmesnika COM dostopali do podatkov modela ter krmilili simulacije v mikrosimulacijskem programu. Slika 5.3 prikazuje celoten proces in logiko postopka krmiljenja na cestni arteriji. S slike je razvidno, da vmesnik COM služi za povezavo oziroma komunikacijo med krmilnikom in simulacijo prometa v okolju VISSIM. Vmesnik pridobiva detektirane podatke iz mikroskopskega simulatorja in jih pošilja v krmilnik. Povratno krmilnik pošlje nazaj krmilne ukaze simulatorju za krmiljenje simuliranega prometa. Vmesnik COM služi kot prevajalec med obema deloma: pretvarja in pošilja ukaze krmilne logike algoritma in s tem spreminja krmilne programe v okolju prometne simulacije.



Slika 5.3: Povezava med programskim orodjem VISSIM, vmesnikom COM in Visualom Basicom (stohastičnim algoritmom učenja Q)

Figure 5.3: Relationship between VISSIM software, COM interface and Visual Basic (stochastic Q-learning algorithm)

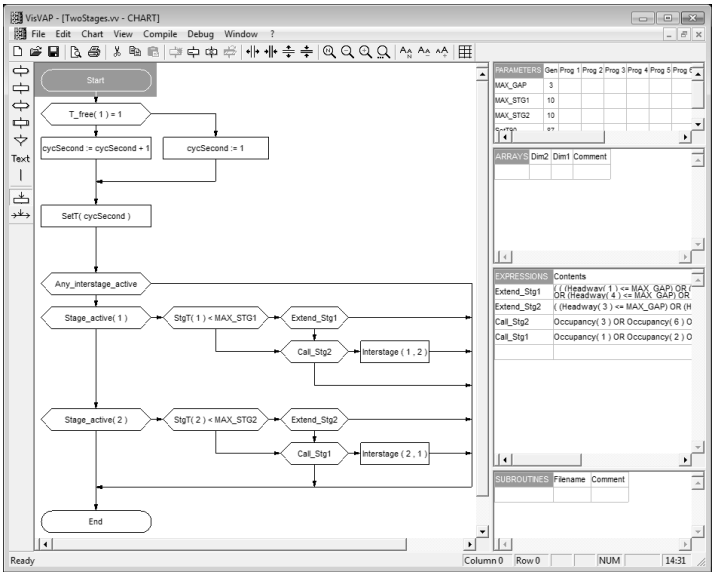
5.1.2 Klasično prometno odvisno krmiljenje z mikrosimulacijskim orodjem

Programsko orodje VISSIM omogoča implementacijo časovno in prometno odvisnih SSN. Klasično prometno odvisno krmiljenje je mogoče modelirati z dodatnim modulom VAP (*Vehicle Actuated Programming*) programskega orodja VISSIM. Modul VAP omogoča programskemu orodju VISSIM simuliranje oziroma upoštevanje želenih prometno odvisnih SSN. Krmilna logika je zapisana v enostavnem programskem jeziku v tekstovni datoteki. Modul VAP pretvarja krmilno logiko v krmilne ukaze za prometno mrežo v simulaciji VISSIM. Sočasno se iz simulacije prek detektorjev prometa prenašajo spremenljivi prometni podatki, ki se obdelujejo v zapisani logiki. Detektorji prometa so postavljeni na vhodnih pasovih križišča.

Logiko krmiljenja SSN v modulu VAP je možno definirati tudi v obliki diagramov z uporabo enostavnega orodja VisVAP (krajše za »Visual VAP«). VisVAP omogoča tudi, da uporabnik najde in odpravi napako še pred simulacijo. Uporabnik lahko preveri zapisano logiko korak po korak (VAP, 2011).

S pomočjo modula VAP smo določili logiko klasičnega prometno odvisnega krmiljenja SSN. V simulacijah s klasičnim prometno odvisnim krmiljenjem smo uporabili enako prometno mrežo kot pri simulacijah s predlaganim algoritmom učenja Q, vključno z vhodnimi prometnimi podatki. Tako kot pri krmiljenju s predlaganim algoritmom učenja Q smo tudi z modulom VAP omejili minimalni

zeleni čas in omogočili spremenljive semaforske cikluse. Krmilne signale smo omejili samo na zeleno in rdečo luč. Na vseh treh križiščih smo upoštevali dvofazne krmilne programe. Možni izbiri SSN sta bili podaljšanje trenutne faze ali zamenjava le-te. Po poteku minimalne zelene faze se krmilnik prvič odloči za podaljšanje ali zamenjavo prve faze. Dolžino minimalnega zelenega signala smo določili na 10 sekund. Za podaljšanje ali zamenjavo faz se krmilnik odloča na podlagi časovne vrzeli med vozili posamezne smeri. Časovne vrzeli se merijo s pomočjo detektorjev, ki smo jih namestili na vseh krakih križišč. Na podlagi izkušenj, preizkusov in rezultatov smo določili maksimalno časovno vrzel, ki je znašala 3 sekunde. Če je torej čas med prvim in drugim vozilom na enem izmed detektorjev na glavni smeri večji kot 3 sekunde, krmilnik zamenja glavno fazo 1 na fazo 2. Če pa je čas krajši od maksimalne časovne vrzeli, krmilnik podaljšuje fazo 1. Maksimalno podaljšanje dolžine faze 1 je odvisno od maksimalnega ciklusa. Maksimalna dolžina faze 1 je razlika med maksimalnim ciklusom in minimalno fazo 2. Maksimalen ciklus smo upoštevali 120 sekund. Rdeči signal v vseh smereh smo upoštevali 3 sekunde. Logika podaljševanja ali zamenjave faze 1 v 2 velja tudi obratno – za fazo 2 v 1. Razlika je le pri merjenju časovnih vrzeli, in sicer namesto na glavni smeri se vrzeli merijo na detektorjih, postavljenih na stranskih krakih križišča. Z opisano metodologijo klasičnega prometno odvisnega krmiljenja smo za vse scenarije naredili po 50 iteracij. Trajanje ene iteracije je bilo 3600 sekund. Za vsako iteracijo je bilo izbrano različno seme generatorja naključnih števil. S tem so bile zajete stohastična narava prometa in realnejše simulacije vozil. **Napaka! Vira sklicevanja ni bilo mogoče najti.** prikazuje videz modula VAP oziroma VisVAP.

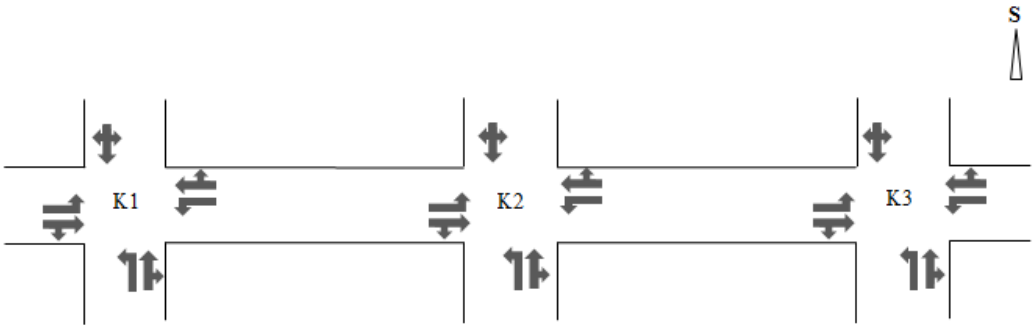


Slika 5.4: Logika krmiljenja SSN v modulu VisVAP

Figure 5.4: Logic to control traffic lights in VisVAP module

5.2 Vhodni podatki in izhodišča simulacij

Zmogljivosti in učinkovitost algoritma na podlagi učenja Q smo analizirali na cestni arteriji treh križišč, kar je geometrijsko gledano povsem tipična cestna arterija brez nikakršnih poenostavitev. Primer cestne arterije smo uporabili za preverjanje metode v stohastičnem okolju ter določevanju parametrov, ki vplivajo na algoritem. Mreža zajema tri štirikraka križišča z levimi zavijalnimi pasovi na vseh krakih, razen na severnih. Na vseh treh križiščih so dvofazni semaforji. V prvi fazi je zeleni signal v smeri zahod–vzhod, v drugi fazi sever–jug. Na vseh križiščih oziroma krakih so možni vsi manevri (desno, naravnost, levo). Shema obravnavane prometne mreže s shematsko prikazanimi kraki in manevri je predstavljena na sliki 5.5.



Slika 5.5: Shema obravnavane prometne mreže

Figure 5.5: Road network scheme

5.2.1 Prometne obremenitve

Prometno mrežo smo obremenjevali z različnimi prometnimi obremenitvami. Uspešnost in učinkovitost predlaganega algoritma smo analizirali v velikih prometnih obremenitvah. Simulacije smo izvedli v nenasičenih in nasičenih prometnih razmerah (v/c < 0,90 in v/c = 1). V nenasičenih prometnih razmerah so bila razmerja med volumnom in kapaciteto na vseh treh križiščih med 0,85 in 0,9, torej na meji z nasičenimi razmerami. S tem smo želeli simulirati samo konične ure, saj navadno zunaj konic dosegamo zadovoljiv nivo uslug tudi s klasičnim prometno odvisnim krmiljenjem. Upoštevane prometne obremenitve so prikazane v naslednjih preglednicah v vozilih na uro (preglednica 5.1, preglednica 5.2 in preglednica 5.3). Strukturo prometa v simulacijah smo določili 95 % osebnih in 5 % tovornih vozil. Za boljšo ponazoritev prometnih obremenitev vseh treh preverjenih setov volumnov je na sliki 5.6 predstavljeno skupno število vstopnih vozil za posamezno simulacijsko uro. Najmanjše obremenitve so pri P1, največje pa pri P3. V slednjem primeru je poleg največjih obremenitev bistveno spremenjeno razmerje prometa med glavno in prečno smerjo.

Preglednica 5.1: Prometne obremenitve P1

Table 5.1: Traffic volume P1

Krak/Križišče	Križišče 1 (K1)				Križišče 2 (K2)				Križišče 3 (K3)			
	L	N	D	Vsi	L	N	D	Vsi	L	N	D	Vsi
Sever (S)	40	20	40	100	40	20	40	100	40	20	40	100
Vzhod (V)	114	685	114	913	132	793	132	1057	156	938	156	1250
Jug (J)	80	40	80	200	80	40	80	200	80	40	80	200
Zahod (Z)	156	938	156	1250	132	793	132	1057	114	685	114	913

Preglednica 5.2: Prometne obremenitve P2

Table 5.2: Traffic volume P2

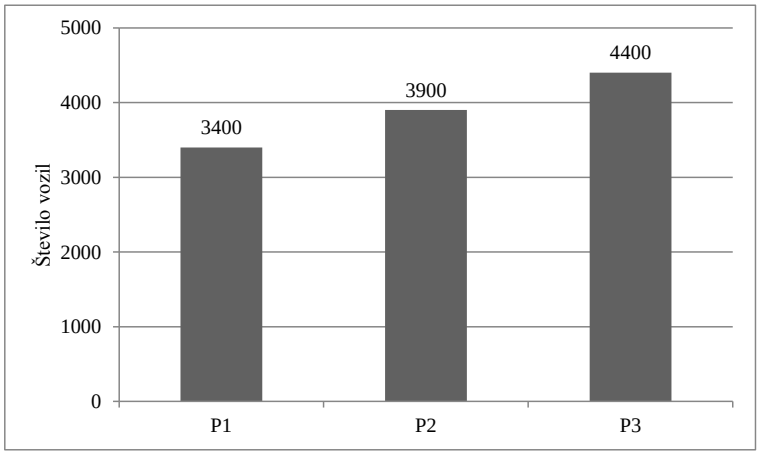
Krak/Križišče	Križišče 1 (K1)				Križišče 2 (K2)				Križišče 3 (K3)			
	L	N	D	Vsi	L	N	D	Vsi	L	N	D	Vsi
Sever (S)	40	20	40	100	40	20	40	100	40	20	40	100
Vzhod (V)	132	790	132	1054	156	934	156	1246	188	1125	188	1501
Jug (J)	80	40	80	200	80	40	80	200	80	40	80	200
Zahod (Z)	188	1125	188	1501	156	934	156	1246	132	790	132	1054

Preglednica 5.3: Prometne obremenitve P3

Table 5.3: Traffic volume P3

Krak/Križišče	Križišče 1 (K1)				Križišče 2 (K2)				Križišče 3 (K3)			
	L	N	D	Vsi	L	N	D	Vsi	L	N	D	Vsi
Sever (S)	40	20	40	100	40	20	40	100	40	20	40	100
Vzhod (V)	137	821	137	1095	143	855	143	1141	150	900	150	1200
Jug (J)	200	100	200	500	200	300	200	700	200	100	200	500
Zahod (Z)	150	900	150	1200	143	855	143	1141	137	821	137	1095

Razmerje vhodnega prometa iz glavnih in prečnih smereh je pri obremenitvi P1 74 : 26, pri P2 je 77 : 23 in pri P3 je 55 : 45. Prvi dve prometni obremenitvi imata bistveno drugačno razmerje med vhodnim prometom iz glavnih proti stranskim smerem. Predstavljata tipične obremenitve arterije. Tretja obremenitev P3 je bolj uravnotežena. Z njo smo želeli dokazati, da je algoritem učinkovit in odziven ob večjih spremembah v prometnem toku, npr. preusmeritev prometa zaradi izrednega dogodka, saj se iz južnih krakov prometni volumen bistveno poveča.

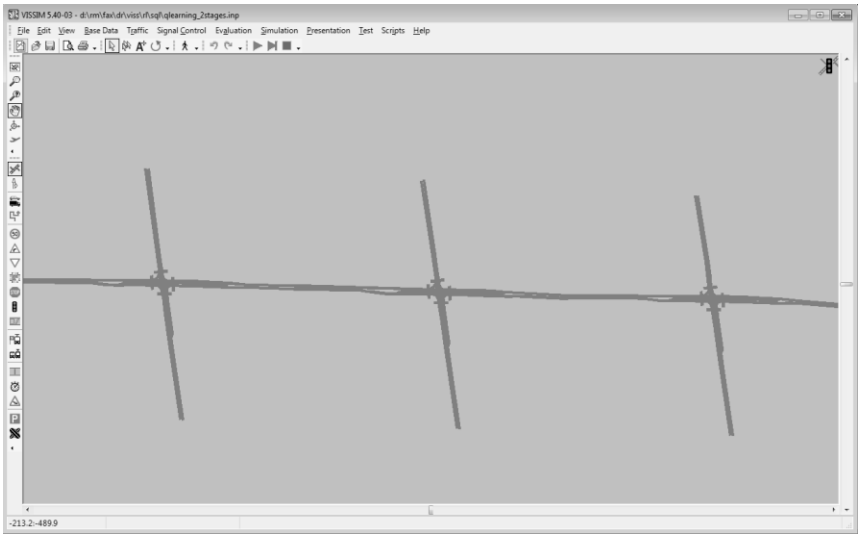


Slika 5.6: Prometne obremenitve (skupno število vozil na vhodu) za tri scenarije

Figure 5.6: Traffic volumes (total vehicle entrance)

5.2.2 Geometrija arterije

Slika 5.7 prikazuje cestno arterijo treh križišč, ki smo jo izdelali in simulirali s programskim orodjem VISSIM. Isto mrežo smo uporabil za simulacije s predlaganim algoritmom Q kot tudi s klasičnim prometno odvisnim krmiljenjem.



Slika 5.7: Prikaz cestne arterije v programskem orodju VISSIM

Figure 5.7: A screenshot of arterial road in VISSIM

5.2.3 Izhodišča simulacij

Ustreznosti posameznih vhodnih parametrov smo preverili z več iteracijami, kjer ena iteracija predstavlja eno uro (3600 s). Za vsak sklop oziroma kombinacijo izbranih parametrov smo izvedli po 50 iteracij, kar znaša med 1500 in 2000 semaforskimi ciklusi.

Na začetku simulacije oziroma v prvi iteraciji navadno agenti nimajo nikakršnega predzadnja pri izbiri akcij, to pomeni, da so vse vrednosti $Q\ (s_t, a_t)$ nič. Posledično se agenti na začetku odločajo naključno. Za vse tri prometne obremenitve smo v prvi iteraciji posamezne simulacije začeli s prazno matriko Q. Začetek simuliranja s prazno matriko oziroma brez kakršnega koli agentovega predznanja smo imenovali inicializacija.

Seveda pa obstaja še druga možnost, in sicer da že na začetku podamo agentu določeno predznanje. Vrednosti $Q\ (s_t, a_t)$ niso enake nič, temveč so vrednosti iz predhodno narejenih simulacij. Test smo v našem primeru izvedli kot preveritev, ali so agenti sposobni samostojne prilagoditve novim, še neznanim prometnim razmeram. Tako smo izvedli še dodatne preizkuse brez inicializacije. Po končanih 50 simulacijah izbranih parametrov s prometnimi obremenitvami P2 smo mrežo obremenili s prometnimi volumni P3 brez inicializacije. Agenti niso začeli s prazno matriko Q ali drugače, začeli so z določenimi izkušnjami iz predhodnih scenarijev. Izkoriščali so predznanje iz že doseženih parov stanj in akcij ter se učili naprej za nove prometne situacije. V naslednjem poglavju so omenjeni scenariji in rezultati podrobneje prikazani in opisani.

5.3 Rezultati raziskave in ocena uspešnosti algoritma

V predhodnem poglavju so bile opisane prometne obremenitve, s katerimi smo obremenjevali prometno mrežo. Za vsako izmed treh prometnih obremenitev smo izvedli več analiz, kako različni parametri vplivajo na učinkovitost predlaganega algoritma stohastičnega učenja Q. Parametri, ki smo jih variirali v analizah, so bili:

- vpliv lokalne oziroma globalne nagrade,
- faktor stopnje učenja α,
- faktor diskontiranja γ,
- parameter raziskovanja ε,
- parameter izkoriščanja znanja $n\varepsilon$.

Poleg naštetega smo preverili tudi vpliv komunikacije med agenti ter vpliv pomena uteži pri nagrajevanju agentov.

Programsko orodje VISSIM ponuja velik nabor izhodnih rezultatov. Najznačilnejši so povprečna zamuda na vozilo, število prepeljanih vozil skozi mrežo, povprečna hitrost, število ustavljanj na vozilo, prevožena pot, potovalni časi, zajezitvene dolžine itd. Za določitev optimalnih parametrov smo primerjali povprečno zamudo na vozilo v posamezni simulaciji.

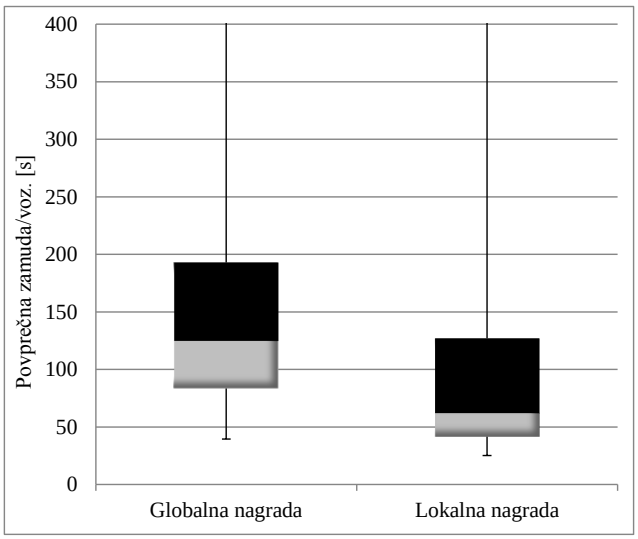
5.3.1 Analiza parametrov in konvergence algoritma

Za določitev najboljših vhodnih parametrov smo upoštevali vse preverjene simulacije oziroma iteracije na prometni arteriji na sliki 5.7, v različnih razmerah in z različnimi parametri, tako kot je napisano na začetku tega podpoglavja. Rezultate analize izbranih parametrov smo prikazali s škatlo z brki (ang. *box plot*). Škatla z brki je v opisni statistiki zelo uporabljana in prikladna za grafično ponazoritev številčnih podatkov s 5 števili: minimalna vrednost vzorca, prvi kvartil (q1), mediana (q2), tretji kvartil (q3) in maksimalna vrednost vzorca. Pri analizi optimalnih parametrov smo upoštevali vse iteracije z različnimi prometnimi obremenitvami. Za primere, kjer smo izvedli dodatno analizo s testom *T*, smo primerjali rezultate zadnjih 20 od 50 iteracij posamezne kombinacije. Zadnje simulacije predstavljajo stanja, za katera predpostavljamo, da so agenti že naučeni in ne raziskujejo več. Za določene ugodne kombinacije parametrov so vrednosti v zadnjih iteracijah blizu optimuma z zelo majhnimi nihanji. Na slikah za določitev vhodnih parametrov so na ordinatah prikazane povprečne zamude na vozilo v sekundah, na abscisah pa posamezni analizirani parametri.

5.3.1.1 Raziskava vpliva tipa nagrade

Slika 5.8 prikazuje dva vzorca rezultatov simulacij, in sicer z uporabo globalne ali lokalne nagrade. Želeli smo določiti, katero nagrajevanje je uspešnejše pri zmanjševanju povprečnih zamud. Levi del slike predstavlja rezultate z globalnim nagrajevanjem, medtem ko desni z lokalnim nagrajevanjem. Konec spodnjega dela brkov predstavlja minimalno vrednost, spodnja meja sivega pravokotnika predstavlja prvi kvaril (25. centil), meja med sivo in črno barvo predstavlja mediano, zgornji del črnega pravokotnika tretji kvaril (75. centil) in zgornji del brkov maksimalno vrednost. Pri analiziranju vseh upoštevanih iteracij je pri uporabi lokalne nagrade večji raztros rezultatov v primerjavi z globalnimi nagradami. Namenoma je skala ordinate omejena na vrednost 400, saj so bile maksimalne vrednosti tudi več kot 900 s. Pri uporabi lokalne nagrade je dosežena minimalna povprečna zamuda na vozilo 25 s, medtem ko je pri globalni nagradi 40 s. Z uporabo lokalne nagrade je bila ena četrtina vseh povprečnih zamud na vozilo nižja od 41 s, pri globalni nagradi pa nižja od 83 s. Z uporabo lokalne nagrade je 50 % vseh povprečnih zamud manjših ali enakih 62 s, na drugi strani

je vrednost 125 s z uporabo globalne nagrade. 75 % povprečnih zamud na vozilo je nižjih od 127 s z lokalno nagrado in 193 s v primeru globalnih nagrad. Pri analizi določitve globalne ali lokalne nagrade je razvidno, da je uporaba lokalne nagrade bistveno boljša kot uporaba globalne, saj je bolj navzdol utežen desni diagram. Menimo, da je razlog v tem, da se pri upoštevanju globalne nagrade zamegli uspešnost posameznega agenta, kar pomeni, da dobi agent manj zanesljive izkušnje, zato bodo slabše prihodnje odločitve.



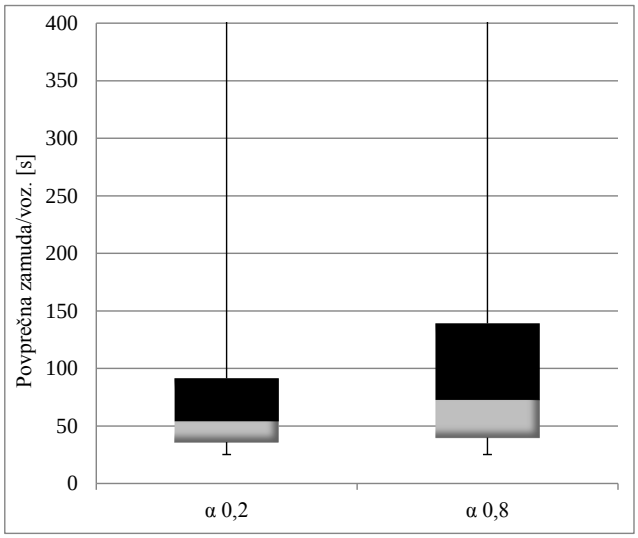
Slika 5.8: Primerjava vseh iteracij z uporabo globalne in lokalne nagrade

Figure 5.8: Comparison of all iterations using global and local reward

5.3.1.2 Raziskava vpliva faktorja stopnje učenja

Konvergenca stohastičnih problemov je zagotovljena, če faktor stopnje učenja zmanjšujemo proti nič. Pogosto se uporablja konstantna vrednost blizu vrednosti nič (Sutton in Barto, 1998). V naši raziskavi smo preverili uspešnost algoritma za vrednosti faktorja stopnje učenja za vrednosti blizu 0 in 1. Slika 5.9 predstavlja rezultate za izbiro vrednosti faktorja učenja. Za boljši prikaz je skala povprečnih zamud omejena navzgor, saj so bile maksimalne vrednosti z določenimi kombinacijami parametrov večje kot 900 s. Za potrebe določitve faktorja α smo upoštevali samo iteracije z uporabljeno lokalno nagrado. Primerjavo smo naredili za vrednosti 0,2 in 0,8. Iz grafičnega prikaza (uteženosti diagrama) je razvidno, da dobimo nekoliko boljše rezultate z uporabo nižje vrednosti α. V obeh primerih je razpršenost vrednosti povprečnih zamud na vozilo zelo podobna, prav tako je bila dosežena v obeh primerih minimalna povprečna zamuda na vozilo cca. 25 s. Z uporabljenim faktorjem 0,2 in upoštevanimi lokalnimi nagradami je 25 % vseh rezultatov nižjih od 36 s, z uporabljenim faktorjem 0,8 pa nižjih od 40 s. Polovico vseh povprečnih zamud je nižjih od 54 s pri α

= 0,2 in 73 s v primeru faktorja α = 0,8. Tri četrtine vseh povprečnih zamud na vozilo je nižjih od 92 s faktorjem učenja 0,2 in 139 s z uporabo vrednosti 0,8. Rezultati dokazujejo, da z upoštevanjem konstantne vrednosti faktorja stopnje učenja blizu 0, dosežemo večjo uspešnost in učinkovitost algoritma, v nadaljnjem delu bi bilo treba preveriti še vrednosti α < 0,2.



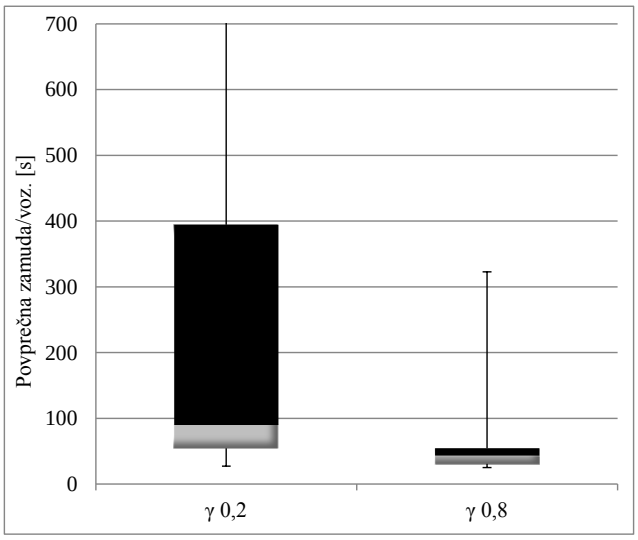
Slika 5.9: Povprečne zamude na vozilo po α z lokalno nagrado

Figure 5.9: Average delay time per vehicle by α with local reward

5.3.1.3 Raziskava vpliva faktorja diskontiranja

Faktor diskontiranja definira pomembnost naslednje nagrade. Iz raziskav Prashantha in Bhatnagara (2011) ter Wieringa in sod. (2004) zasledimo, da so rezultati z vrednostmi faktorja diskontiranja bliže 1 boljši. Predlagan algoritem spodbujevanega učenja smo preverili za vrednosti faktorja diskontiranja 0,2 in 0,8. Na sliki 5.10 smo z upoštevanjem lokalnih nagrad in vrednostjo faktorja učenja 0,2 določili večjo učinkovitost algoritma za različna faktorja diskontiranja γ (0,2 in 0,8). S slike je razvidno, da z vrednostjo γ = 0,8 dobimo bistveno boljše rezultate. Diagram je neprimerno bolj utežen navzdol ob višji vrednosti. To pomeni, da si agent prizadeva dolgoročno višje nagrade. Diskontiran faktor igra ključno vlogo, saj nižje vrednosti služijo za večjo diskontiranost prihodnjih nagrad, s čimer damo manjši poudarek nagradam. Obratno dosežemo z višjimi vrednostmi. Analizirani rezultati z upoštevanim lokalnim nagrajevanjem in faktorjem učenja 0,2 pokažejo, da je ob γ = 0,8 bistveno manjši raztros rezultatov kot pri γ = 0,2. Minimalna dosežena vrednost povprečnih zamud na vozilo z izbiro faktorja diskontiranja 0,8 je 25 s, z izbiro faktorja 0,2 pa 29 s. Četrtina vseh rezultatov je nižjih od 31 z γ = 0,8, z uporabljeno γ = 0,2 pa 55 s. 50 % vseh vrednosti povprečnih zamud na vozilo je nižjih od 43 s pri uporabi vrednosti faktorja diskontiranja 0,8 in 90 s z

uporabo vrednosti γ = 0,2. 75 % vseh povprečnih zamud na vozilo je nižjih od 54 s faktorjem γ = 0,8 in 394 s z uporabo vrednosti 0,2. Rezultati dokazujejo, da z upoštevanjem konstantne vrednosti faktorja diskontiranja blizu 1, dosežemo večjo uspešnost in učinkovitost algoritma, v nadaljnjem delu bi bilo treba preveriti še vrednosti γ > 0,8.

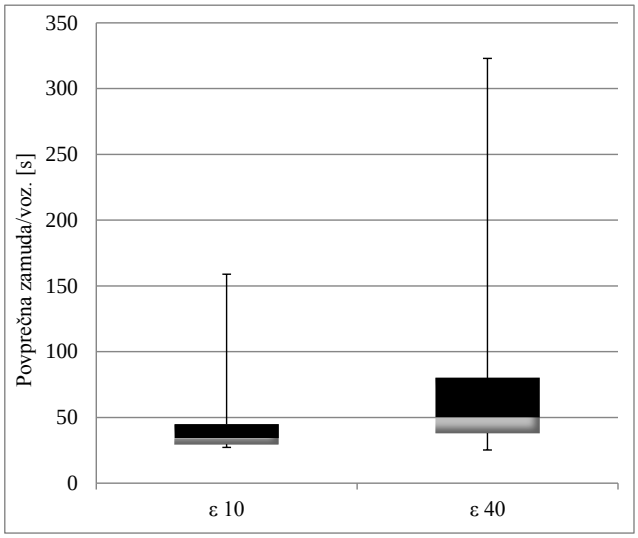


Slika 5.10: Povprečne zamude na vozilo po γ z α = 0,2 in z lokalno nagrado

Figure 5.10: Average delay time per vehicle by γ with α = 0,2 and local reward

5.3.1.4 Raziskava vpliva parametra raziskovanja

Za določitev optimalne vrednosti parametra raziskovanja ε smo analizirali izhodne rezultate za vrednosti 10 in 40. To pomeni, da so agenti raziskovali samo v 10 oziroma 40 %. Pri tem smo upoštevali samo iteracije z lokalno nagrado α = 0,2 in γ = 0,8. Na sliki 5.11 je očitno, da nižji odstotek oziroma vrednost ε da bistveno boljše rezultate. Razlika med minimalnimi vrednostmi povprečnih zamud je sicer majhna, 25 s v primeru ε = 10 in 27 s z ε = 40, vendar je raztros rezultatov bistveno manjši pri ε = 10. V prvem kvartilu je vseh povprečnih zamud na vozilo nižjih od 30 s, pri ε = 40 je ta vrednost 38 s. Z uporabo parametra ε = 10 je polovico vseh povprečnih zamud manjših ali enakih 34 s, na drugi strani je vrednost 50 s z uporabo parametra ε = 40. Tri četrtine povprečnih zamud na vozilo s parametrom ε = 10 je manjših od 45 s in 80 s v primeru izbire vrednosti 40.

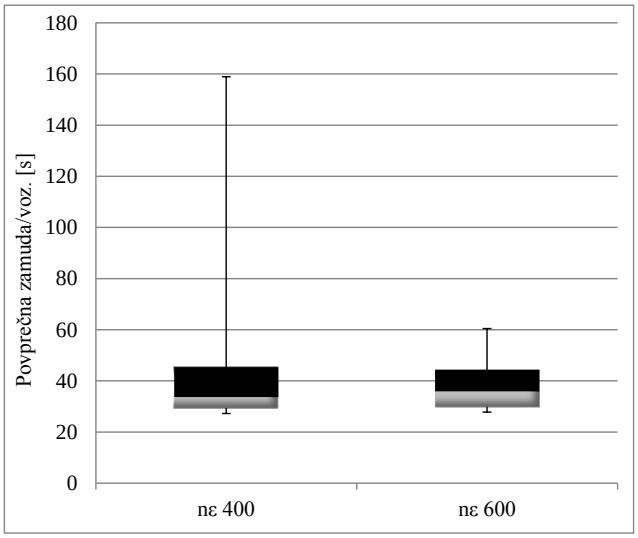


Slika 5.11: Povprečne zamude na vozilo po ε z $\gamma = 0{,}8$, $\alpha = 0{,}2$ in z lokalno nagrado

Figure 5.11: Average delay time per vehicle by ε with $\gamma = 0{,}8$, $\alpha = 0{,}2$ and local reward

5.3.1.5 Raziskava vpliva parametra izkoriščanja znanja

Kot smo zapisali v poglavju 4.6, smo variirali še parameter izkoriščanja znanja n_ε, tj. število obiskov posameznega stanja, po katerem agent začne izkoriščati samo znanje in ne raziskuje več. Slika 5.12 prikazuje rezultate vrednosti parametra n_ε = 400 in 600. Pri tem je treba dodati, da smo analizirali vrednosti, tudi nižje od 400 in višje od 600. Analizirali smo samo ti dve vrednosti, ker so bili drugi rezultati bistveno slabši v primerjavi s 400 in 600. V iskanju optimalne vrednosti smo se tako omejili samo na ta dva seta. Iz rezultatov vidimo, da je raztros vrednosti povprečnih zamud na vozilo manjši pri faktorju 600. Z vrednostjo n_ε = 400 je minimalna vrednost zamud 27 s, s parametrom n_ε = 600 pa 28 s. Četrtina vseh vrednosti je pri parametru 400 29 s, pri parametru 600 pa 30 s. Z uporabo parametra 400 je mediana 34 s in 36 s z upoštevanim parametrom 600. 75 % rezultatov je nižjih od 45 s (n_ε = 400) in prav tako nižjih od 45 s ob parametru 600. S slike ni povsem razvidno, katera izbrana vrednost parametra je boljša. Statistično značilnost razlike med vzorcema smo preverili s pomočjo testa *T*, ki je pokazal, da razlika ni statistično značilna. Iz tega lahko izhajamo, da izbira vrednosti n_ε = 400 ali 600 ne vpliva bistveno na rezultate. Zaradi majhnih razlik oziroma statistično neznačilnih razlik med 400 in 600 so v naslednjih poglavjih predstavljene dodatne analize za različne prometne obremenitve. Z dodatnimi analizami smo želeli preveriti, ali veljajo enake ugotovitve pri vseh analiziranih prometnih volumnih.

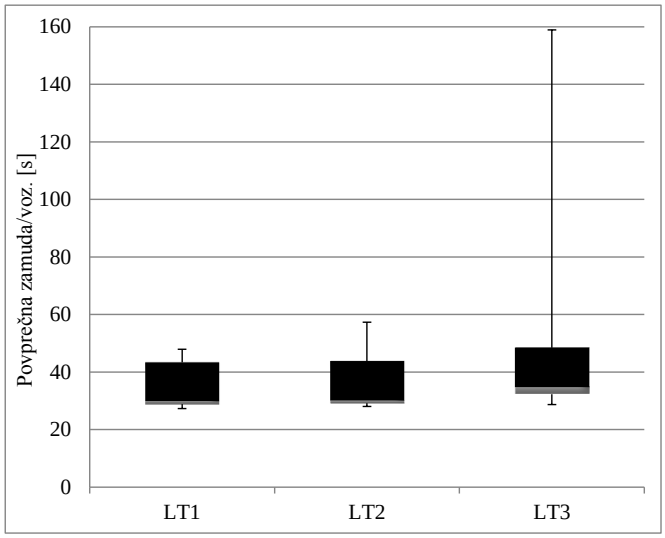


Slika 5.12: Povprečne zamude na vozilo po n_ε z ε = 10, γ = 0,8, α = 0,2 in z lokalno nagrado

Figure 5.12: Average delay time per vehicle by n_ε with ε = 10, γ = 0,8, α = 0,2 and local reward

5.3.1.6 Raziskava vpliva komunikacije med agenti

Pri analizi komunikacije med agenti smo preverili tri scenarije komuniciranja med njimi. V prvem primeru ni bilo nobene komunikacije med agenti (LT1), v drugem smo podali informacijo o fazi vodilnega, levega agenta (LT2) in v tretjem primeru smo poleg faze vodilnega agenta dodali še informacijo o trajanju zelenega signala vodilnega agenta (LT3). S slike 5.13 je razvidno, da so rezultati zelo podobni pri LT1 in LT2, medtem ko so pri LT3 povprečne zamude višje. Najmanjša razpršenost rezultatov je pri LT1, nato LT2 in največja pri LT3. Najmanjša povprečna zamuda je bila pri LT1, in sicer 27 s, pri LT2 28 s in 29 s pri LT3. Pri LT1 je v prvem kvartilu vseh povprečnih zamud na vozilo nižjih od 29 s, z upoštevano informacijo o fazi vodilnega agenta so vrednosti nižje od 30 s in pri LT3 32 s. Pri LT1 je polovico vseh rezultatov nižjih od 30 s, pri LT2 prav tako 30 s in pri LT3 je vrednost 35 s. Tri četrtine vseh povprečnih zamud na vozilo je nižjih od 43 s brez komunikacije, 44 s pri LT2 in 48 s pri LT3. Rezultati kažejo, da več informacij med agenti ne zagotavlja nižjih zamud. Dodatna informacija o trajanju zelenega signala vodilnega agenta otežuje učenje agentov. Za dodatno analizo določitve najboljšega scenarija smo se poslužili testa *T*. Primerjavo smo naredili med LT1 in LT2. Tudi na podlagi testa *T* se je izkazalo, da razlika med rezultati brez komunikacije med agenti in informacije faze glavnega agenta ni statistično značilna (vrednost p = 0,38). Zaradi majhnih razlik oziroma statistične neznačilnosti med LT1 in LT2 so prav tako kot pri parametru izkoriščanja znanja n_ε v naslednjih poglavjih predstavljene dodatne analize za različne prometne obremenitve. Z dodatnimi analizami smo želeli preveriti, ali veljajo enake ugotovitve pri vseh analiziranih prometnih obremenitvah.



Slika 5.13: Povprečne zamude na vozilo po tipu komunikacije med agenti z n_ε = 400, ε = 10, γ = 0,8, α = 0,2 in z lokalno nagrado

Figure 5.13: Average delay time per vehicle by type of communication between agents with n_ε = 400, ε = 10, γ = 0,8, α = 0,2 and local reward

Po izvedenih analizah začetnih parametrov lahko povzamemo, da dobimo najnižje povprečne zamude na vozilo z izbiro lokalnega nagrajevanja, faktorjem učenja 0,2, diskontiranim faktorjem 0,8 in faktorjem ε = 10. Izbrani parametri so neodvisni od prometnega toka oziroma prometnih obremenitev. Na podlagi vseh simulacij pa ne moremo trditi, katera vrednost n_ε je boljša, zato so v nadaljevanju predstavljene dodatne analize za različne prometne obremenitve posebej. Enako velja za komunikacijo med agenti, tj. brez komunikacije med agenti (LT1) ali z informacijo o fazi vodilnega agenta (LT2). V nadaljevanju smo navedene parametre uporabili za nadaljnjo analizo uspešnosti algoritma in primerjavo s klasičnim prometno odvisnim krmiljenjem.

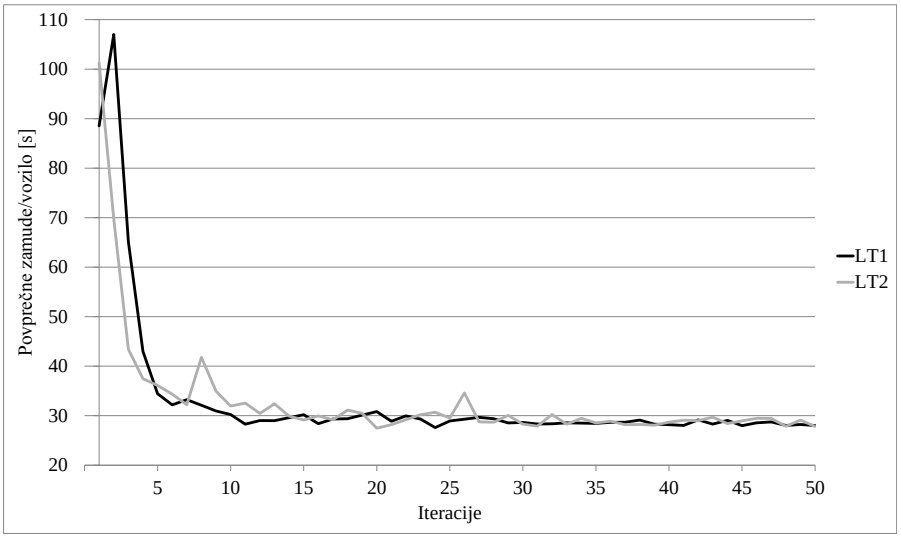
5.3.1.7 Analiza konvergence v nenasičenih prometnih razmerah P1

S predlaganim algoritmom krmiljenja smo s predhodno analiziranimi in določenimi parametri najprej vrednotili rezultate na arteriji s prometnimi obremenitvami P1. Prometne obremenitve P1 odražajo na arteriji nenasičeni prometni tok, vendar na meji z nasičenim prometnim stanjem, saj je bilo razmerje v/c na treh križiščih med 0,85 do 0,9. Izbrani parametri dajejo najmanjše zamude, rezultati najhitreje konvergirajo k optimalnim vrednostim in dajejo najmanjšo razpršenost rezultatov. S predhodno analizo s škatlastimi brki glede informacij med agenti smo izločili tretjo možnost LT3, saj je očitno slabša v primerjavi s prvima dvema. Za posamezno kombinacijo parametrov smo izvedli po 50 iteracij z inicializacijo. Na sliki 5.14 je predstavljena konvergenca algoritma za najboljši dve seriji rezultatov. Na sliki so na ordinati prikazane povprečne zamude na vozilo v sekundah. Na abscisi pa

iteracije. Črna krivulja predstavlja učenje agentov brez vseh informacij med tremi agenti (LT1). Siva krivulja pa predstavlja učenje agentov s komunikacijo oziroma informacijo o trenutni fazi vodilnega agenta (LT2). Na sliki se dobro vidi, kako v začetnih iteracijah agenti nimajo še nobenega znanja, saj so povprečne zamude na vozilo v obeh primerih zelo visoke. Po prvih 10 iteracijah se obe krivulji spustita na neprimerno nižje vrednosti v primerjavi z začetnimi. S slike je razvidno, da algoritem v obeh primerih hitro konvergira k optimalnim zamudam. Algoritmi z izbranimi parametri se že po 20 iteracijah približajo optimalnim vrednostim. Med 10 in 30 iteracijo se rezultati gibajo blizu optimalnim vrednostim, vendar krivulje nihajo. Nekoliko večje nihanje je opaziti v drugem primeru z vključeno komunikacijo. To pomeni, da so dodatne informacije med agenti upočasnile učenje agentov v primerjavi s primerom LT1. Krivulji se v zadnjih 20 iteracijah zelo umirita in vrednosti se gibajo blizu optimalnim. Krivulji ne odstopata veliko druga od druge. Iz teh rezultatov lahko sklepamo, da so agenti v zadnjih 20 iteracijah naučeni in izkoriščajo samo še svoje znanje.

Maksimalna vrednost 107 s je v seriji LT1, vendar je ta dosežena v začetnih iteracijah, še v fazi učenja algoritma. Pri LT2 je bila maksimalna vrednost 101 s. Povprečne vrednosti zamud na vozilo zadnjih 20 iteracij se gibajo pod 30 s brez večjih nihanj. Razlike v povprečnih vrednostih zadnjih 20 iteracij so velikostnega reda manj kot 0,5 s v korist primera brez informacij med agenti. Zaradi tako majhnih odstopanj med LT1 in LT2 smo izvedli statistično metodo test *T*, da bi ugotovili, ali obstaja statistično značilna (pomembna) razlika med dvema vrstama informiranja med agenti. Na podlagi rezultatov testa *T*, kjer smo analizirali zadnjih 20 iteracij, lahko zapišemo, da razlika med LT1 in LT2 ni statistično značilna, saj vrednost *p* znaša 0,085.

V nenasičenih razmerah, ko še ne dosežemo kapacitete cestne arterije (med križišči ni velikih zajezitvenih dolžin), smo pričakovali, da bo v pomoč dodatna informacija o fazi agenta (LT2). Z dodatno informacijo naj bi se lažje ustvarila linijska koordinacija med SSN na cestni arteriji. Z dodatno informacijo naj bi bili časovni zamiki med SSN ustrezni, kar bi ustvarilo skupino skupaj vozečih vozil. Vendar iz pridobljenih rezultatov lahko sklepamo, da so križišča ravno dovolj oddaljena drugo od drugega, da informacije o dolžinah kolon podajo vse tisto, kar mora posamezen agent oziroma SSN vedeti o svojem sosednjem oziroma vodilnem agentu. Čeprav ni bila potrjena statistična značilnost (zadnjih 20 iteracij), da so rezultati brez komunikacije boljši, lahko vseeno sklepamo, da se algoritem, gledano na celoten potek učenja, vseeno hitreje nauči. Hitrost konvergence je hitrejša brez dodatnih informacij.

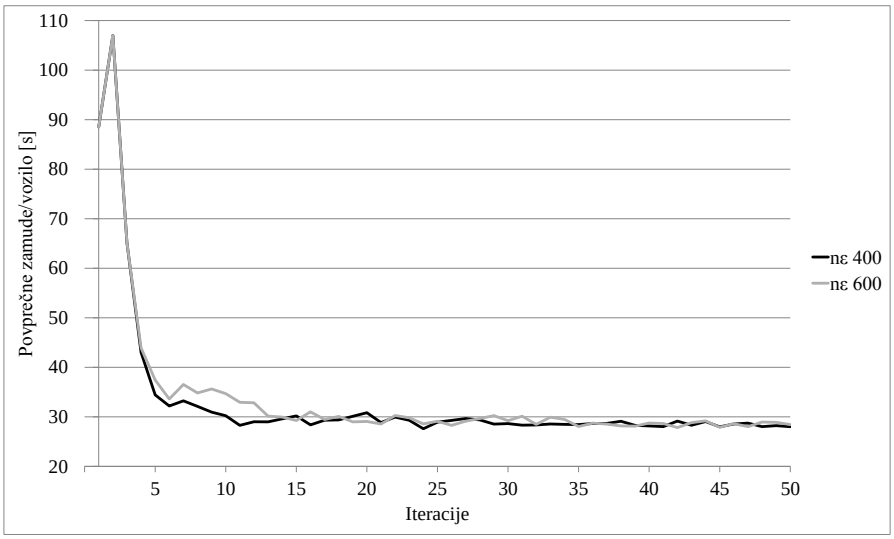


Slika 5.14: Analiza konvergence algoritma v odvisnosti brez komunikacije med agenti in z njo (nenasičeni prometni tok P1)

Figure 5.14: Proposed algorithm behaviour according to no-communication and communication between agents (saturated traffic flow P1)

Slika 5.15 prikazuje rezultate dveh serij z izbranimi parametri α = 0,2, γ = 0,8, ε = 10, brez komunikacije med agenti LT1 in ob lokalni nagradi. Analizirali smo vpliv faktorja n_ε, ki definira, kdaj agent prekine raziskovanje. Primerjava je narejena na podlagi povprečnih zamud na vozilo za vseh 50 iteracij ob prometnih obremenitvah P1. Črna krivulja predstavlja faktor n_ε = 400, kar pomeni, da agent po 400 obiskih posameznega stanja začne izkoriščati svoje znanje. Druga, siva krivulja pa predstavlja n_ε = 600. S slike lahko razberemo, da sta krivulji prvih 5 iteracij skoraj prekriti, vrednosti zamud pa krepko nad minimalnimi. Krivulji še najbolj odstopata druga od druge med 5. in 15. iteracijo. Nihanje krivulj pri obeh serijah, najmanjše v zadnjih 20 iteracijah. Na podlagi grafičnega prikaza lahko zapišemo, da je hitrost konvergence rezultatov nekoliko hitrejša, ko izberemo vrednost n_ε = 400, vendar je razlika minimalna.

Povprečje zadnjih 20 iteracij je manjše v seriji n_ε = 600, vendar je velikostni razred razlike manj kot 0,5 s. S testom *T* smo preverili, ali je razlika med vzorcema zadnjih 20 iteracij statistično značilna, in ugotovili, da ni. Vrednost *p* je znašala 0,161. Prav tako kot v predhodni analizi faktorja n_ε (poglavje 5.3.1.5) lahko ob prometnih obremenitvah P1 sklepamo, da ni bistvenih razlik med izbiro vrednosti 400 in 600.



Slika 5.15: Analiza konvergence algoritma v odvisnosti od parametra n_ε (nenasičeni prometni tok P1)

Figure 5.15: Proposed algorithm behaviour for different n_ε (saturated traffic flow P1)

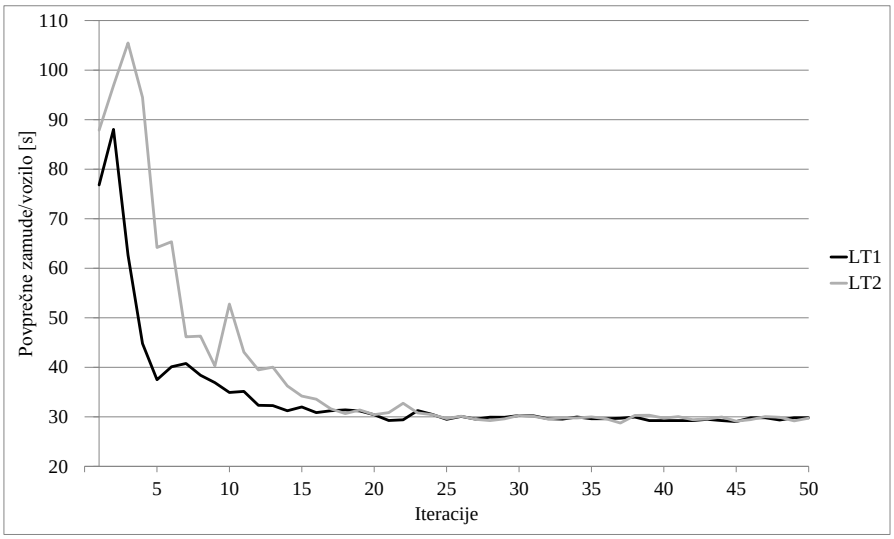
5.3.1.8 Analiza konvergence v nasičenih prometnih razmerah P2

S predhodno določenimi parametri smo model obremenili s prometnimi obremenitvami P2 ter analizirali konvergenco predlaganega algoritma. Prometne obremenitve P2 odražajo na obravnavani arteriji nasičene prometne razmere. To pomeni, da s takim prometnim volumnom dosežemo razmerje volumna proti kapaciteti 1 ali več. Iz analize lahko povzamemo, da izbrani parametri v simulacijah s prometom P2 dajejo najmanjše zamude in najhitreje konvergirajo k optimalnim vrednostim. Tudi v primeru nasičenih prometnih razmer smo dodatno analizirali vpliv komunikacije med agenti. Analizirali smo LT1 in LT2, medtem ko smo informacijo o trajanju zelenega signala vodilnega agenta iz dodatne analize izvzeli. LT3 je bila že v prvotni analizi bistveno slabša kot prvi dve možnosti. Za oba primera smo naredili po 50 iteracij z inicializacijo. Na sliki 5.16 je primerjava med simulacijo brez komunikacije in z njo. Na vertikalni osi grafa so prikazane povprečne zamude na vozilo, na horizontalni pa iteracije. Črna linija (LT1) prikazuje simulacijo brez komunikacije, medtem ko siva linija z informacijo o vodilnem agentu (LT2). Ker smo v obeh primerih začeli simulaciji z inicializacijo, so povprečne zamude na vozilo v začetnih iteracijah visoke. Agenti nimajo nobenega predznanja. Največji skok oziroma največje zmanjšanje povprečnih zamud na vozilo je opaziti med 5. in 10. iteracijo. Obe krivulji se spustita blizu optimalnim vrednostim nekje na 25. iteraciji. Večja hitrost učenja je vidna pri LT1, saj se gibajo zamude blizu minimalnim že pri 15. iteraciji. Poleg počasnejšega učenja pri LT2je opaziti tudi večje zamude v prvih iteracijah. Po 30. iteraciji se nihanje linij zelo umiri. Rezultati zadnjih 20. iteracij so zelo primerljivi med seboj. Tako kot pri prejšnjem primeru s prometnimi obremenitvami P1 se agenti po 30. iteraciji ne učijo več in

samo še izkoriščajo znanje. Obenem pa lahko opazimo, da se s povečanim prometom agenti učijo dlje časa v primerjavi z nenasičenimi razmerami.

Pri obeh serijah so maksimalne vrednosti zamud več kot trikrat večje od optimalnih, vendar so te vrednosti dosežene v začetnih iteracijah. Maksimalna vrednost brez komunikacije je 88 s, pri informaciji o fazi vodilnega agenta pa 105 s. Tako pri LT1 kot pri LT2 je opaziti v prvih 25. iteracijah občuten trend zmanjševanja zamud. Pri LT2 je opaziti okoli 10. simulacije nekoliko večje nihanje, vendar se v nadaljevanju rezultati ponovno postopoma izboljšajo. Kljub nekoliko slabšim rezultatom prvih 25 iteracij ob komunikaciji so razlike povprečnih vrednosti zadnjih 20 iteracij velikostnega reda manj kot 0,5 s. Povprečje zadnjih 20 iteracij je nižje brez informacij med agenti. Rezultati testa *T* kažejo, da je razlika med serijama statistično neznačilna, saj je vrednost $p = 0{,}08$. Na podlagi tega ne moremo trditi, da je boljša izbira brez komunikacije, vendar če gledamo z vidika hitrosti učenja, je taka odločitev boljša.

Povzamemo lahko, da v nasičenih razmerah informacije o vodilnem agentu, ki bi lahko vplivale na časovne zamike med posameznimi semaforji, ne zmanjšajo zamud. Tako kot ob nenasičenih prometnih razmerah smo pričakovali, da bodo dodatne informacije o vodilnem agentu koristnejše v smislu ustvarjanja časovnih zamikov med semaforji. Tudi v tem primeru lahko zapišemo, da so križišča ravno dovolj oddaljena drugo od drugega, da informacije o dolžinah kolon podajajo vse tisto, kar mora posamezni agent vedeti o svojem sosednjem, vodilnem agentu. Dodatna informacija nam sicer ne poslabša končnih rezultatov, vendar učenja agenta upočasni. Hitrost konvergence je ob nasičenih razmerah hitrejša brez komunikacije.

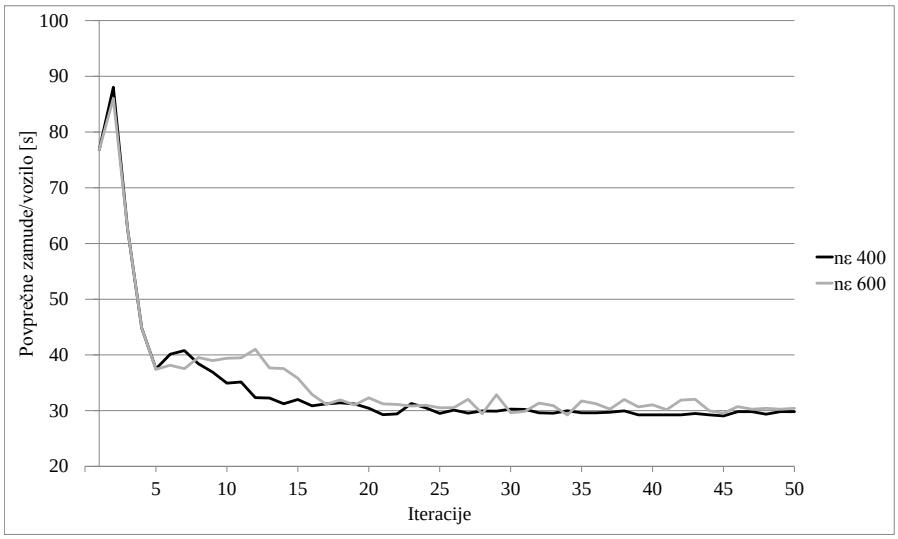


Slika 5.16: Analiza konvergence algoritma v odvisnosti brez komunikacije in z njo med agenti (nasičeni prometni tok P2)

Figure 5.16: Proposed algorithm behaviour according to no-communication and communication between agents (over-saturated traffic flow P2)

Zaradi statistično neznačilnih razlik n_ε = 400 in n_ε = 600, ki smo jih dokazali v raziskavi vpliva parametra izkoriščanja znanja (poglavje 5.3.1.5) v množici vseh izvedenih iteracij, smo posebej analizirali vpliv stopnje učenja samo ob prometnih obremenitvah P2. Slika 5.17 prikazuje rezultate dveh serij, in sicer n_ε = 400 črna linija in n_ε = 600 siva linija. V prvih 5 iteracijah sta liniji skoraj povsem prekriti. Maksimalni vrednosti, tako kot je v začetnih iteracijah brez znanja pričakovano, sta neprimerno višji od končnih. Največja razlika med serijama je med 5. in 15. iteracijo. V teh iteracijah so zamude z višjo vrednostjo parametra n_ε višje v primerjavi s 400. Tudi v tem primeru lahko opazimo, da se rezultati zadnjih 20 iteracij prvega seta gibajo blizu optimalnim vrednostim z zelo malo nihanji. Kljub temu da z izbiro n_ε = 600 dosežemo zelo primerljive končne rezultate, je nihanje rezultatov večje. Iz prikazane grafike razberemo, da je nekoliko boljša izbira vrednost 400. Tudi test *T* to potrjuje, saj je razlika med vzorcema zadnjih 20 iteracij statistično značilna. Vrednost *p* je znašala 6,93 E-06.

Če ob prometnih obremenitvah P1 nismo dokazali, katera vrednost prametra stopnje učenja je boljša, smo v nasičenih razmerah P2 to s testom *T* dokazali. Prav tako je že s slike razvidno, da je linija z vrednostjo 400 stabilnejša. Po našem mnenju je manj obiskanih stanj za začetek čistega izkoriščanja znanja zato, ker je v nasičenih razmerah manj raznolikih prometnih razmer kot v nenasičenih.



Slika 5.17: Analiza konvergence algoritma v odvisnosti od parametra n_ε (nasičeni prometni tok P2)

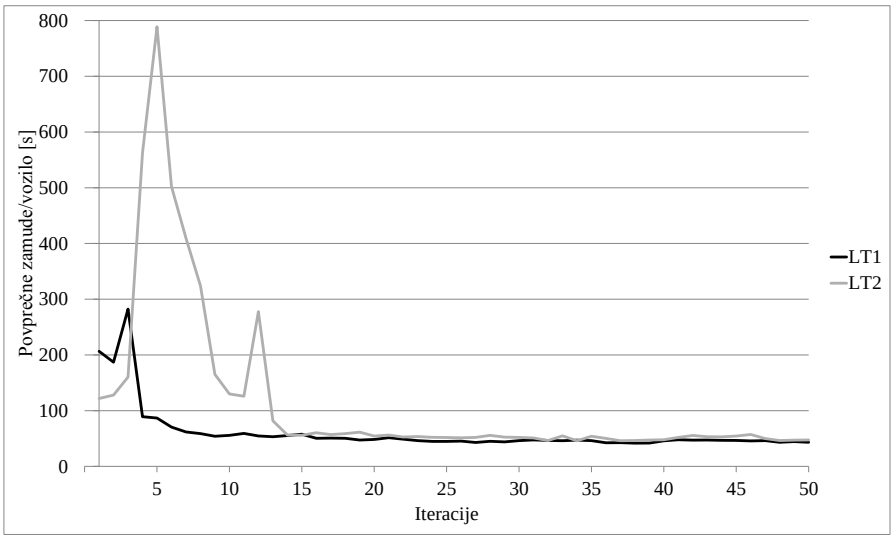
Figure 5.17: Proposed algorithm behaviour for different n_ε (over-saturated traffic flow P2)

5.3.1.9 Analiza konvergence v nasičenih prometnih razmerah P3

V nadaljevanju so predstavljeni rezultati s prometnimi obremenitvami P3. Poleg večjega števila vozil v primerjavi z obremenitvami P1 in P2 je bistveno spremenjeno razmerje med glavno in prečno smerjo. S tem smo želeli preveriti in dokazati, da se je algoritem učenja Q sposoben naučiti in uspešno krmiliti SSN tudi ob večjih spremembah. Prometne obremenitve P3 odražajo nasičene prometne razmere. Število vozil na vstopih v mrežo v primerjavi s P2 je še večje, predvsem pa je bistveno spremenjeno razmerje med glavno in prečno smerjo. V prometnih razmerah P3 smo dosegli konvergenco rezultatov. Tako kot v predhodnih analizah smo tudi v tem primeru izvedli po 50 iteracij na scenarij. Poleg preizkusov z inicializacijo smo izvedli preizkuse brez nje. Na sliki 5.18 je analiza in primerjava konvergence za primer brez komunikacije LT1 in z informacijo o fazi vodilnega agenta. Prva serija je prikazana s črno krivuljo, druga pa s sivo. Analiza je prikazana za povprečne zamude na vozilo v sekundah. S slike je razvidno, da je uspešnost učenja algoritma v prvih iteracijah boljša v primeru LT1. Pri slednjem primeru se agenti hitreje učijo, saj je velik skok pri zmanjšanju zamud viden že pri 5. iteraciji. V drugem primeru (LT2) so velika nihanja in zamude do 15. iteracije. V tem primeru izstopajo zelo visoke vrednosti zamud, kar kaže na povsem neučinkovito krmiljenje. Pri vizualni analizi teh iteracij smo opazil zastoje na vseh krakih križišč. Kljub temu pa je v nadaljnjih iteracijah viden napredek učenja. V obeh primerih je v zadnjih 20 iteracijah s prometnimi obremenitvami P3 nihanje minimalno in zamude se gibajo blizu optimuma. Za oba primera brez komunikacije in z njo velja, da se agenti naučijo ustrezne strategije krmiljenja SSN.

Odraz zgrešenega krmiljenja v prvi fazi učenja v primeru LT2 so tudi neprimerno večje maksimalne zamude (790 s) v primerjavi z maksimalnimi zamudami pri LT1 (280 s). Povprečna vrednost zamud zadnjih dvajsetih iteracij je 45,34 sekunde brez komunikacije. Z upoštevanimi informacijami o fazi glavnega agenta je povprečje 50,34 s. Kljub temu da je razlika očitna, smo preverili statistično značilnost s testom *T*. Vrednost *p* znaša 1,17 E-05, kar potrjuje, da je izbira brez komunikacije boljša.

Tako kot v prejšnjih dveh primeri z manjšimi prometnimi obremenitvami smo tudi tukaj pričakovali, da bodo dodatne informacije med agenti pripomogle k boljšim rezultatom. Od vseh treh analiz z različnimi prometnimi obremenitvami je najočitnejša razlika ravno pri P3, kar nakazuje, da večje kot imamo prometne obremenitve, manj informacij potrebujemo. Dolžine kolon so zadostna informacija, da agent ustrezno in učinkovito krmili promet.



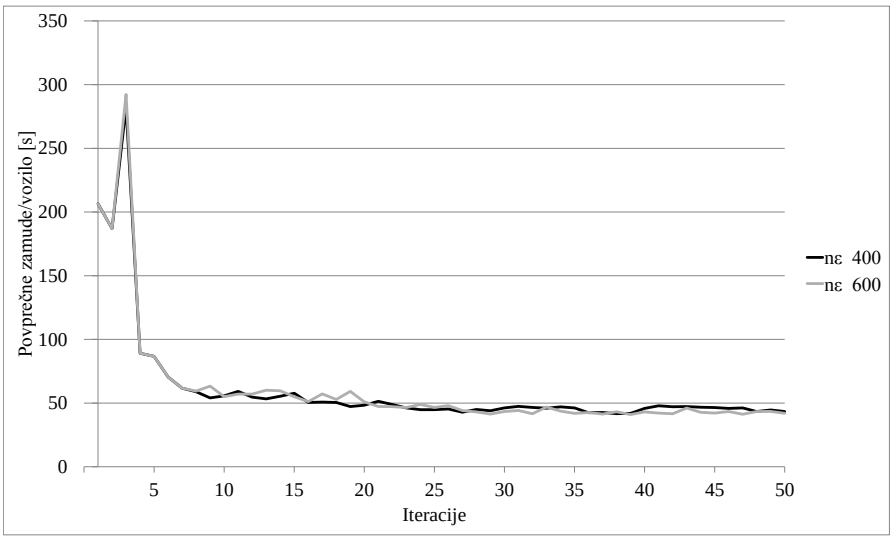
Slika 5.18: Analiza konvergence algoritma brez komunikacije in z njo med agenti (nenasičeni prometni tok P3)

Figure 5.18: Proposed algorithm behaviour according to no-communication and communication between agents (over-saturated traffic flow P3)

Slika 5.19 prikazuje primerjavo med rezultati povprečnih zamud z izbiro parametra n_ε= 400 in 600. Za oba primerjana seta so bili uporabljeni parametri α = 0,2, γ = 0,8, ε = 10, brez komunikacije med agenti LT1 in lokalna nagrada. Na sliki so primerjani rezultati povprečnih zamud na vozilo za vseh 50 iteracij z inicializacijo in prometnimi obremenitvami P3. Iteracije z uporabljenim faktorjem n_ε= 400 so prikazane s črno krivuljo, z nižjo vrednostjo faktorja n_ε pa s sivo barvo. Rezultati kažejo, da se ne glede na izbiro vrednosti parametra algoritem relativno hitro uči. Razlike so med serijami s

prometnimi intenzitetami P3 majhne, še posebno v prvih iteracijah. Nihanje krivulj je zelo podobno. Malenkost večje nihanje sive krivulje je opaziti med 10. in 20. iteracijo.

Povprečje zadnjih 20 iteracij je nižje z izbiro vrednosti 600. V nasprotju s primerom P1 in P2, kjer so bile dosežene nižje zamude z vrednostjo 400. Iz tega lahko sklepamo, da je smiselno pri večjih in bolj uravnoteženih prometnih obremenitvah podaljšati proces raziskovanja. S tem dosežemo, da agent več preizkuša in raziskuje ter izboljša rezultate. Ker je v prometnih obremenitvah P3 večji pretok vozil iz stranskih smeri, je posledično več raznolikih prometnih razmer v primerjavi s prometnimi obremenitvami P1 in P2. S testom *T* smo preverili še statistično značilnost razlike med serijama. Vrednost *p* znaša 1,21 E-04, kar pomeni, da je za prometne obremenitve P3 smiselna uporaba višje vrednosti parametra n_ε.



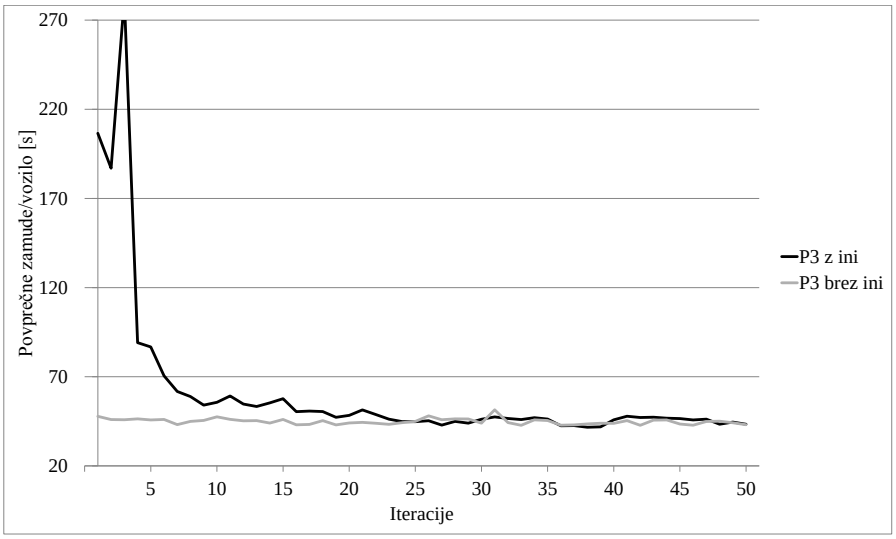
Slika 5.19: Analiza učinkovitosti algoritma v odvisnosti od parametra n_ε (nasičeni prometni tok P3)

Figure 5.19: Proposed algorithm behaviour for different n_ε (over-saturated traffic flow P3)

Z nasičenimi prometnimi razmerami P3 smo želeli dokazati, da je predlagan algoritem učenja Q sposoben učinkovito krmiliti promet tudi, ko prometne obremenitve niso tipične za cestne arterije. Poleg tega smo želeli dokazati, da se je algoritem sposoben naučiti in prilagoditi novim prometnim razmeram samostojno. Še več, želeli smo dokazati, da se je sposoben hitro naučiti tudi na podlagi drugih, predhodnih prometnih razmer. To smo izvedli tako, da smo simulirali P3 brez inicializacije. Z najboljšimi izbranimi parametri, s prometnimi intenzitetami P3 smo začeli simulirati z določenim predznanjem. Najprej smo simulirali 50 iteracij z inicializacijo s prometnimi obremenitvami P2. Z napolnjeno matriko Q smo nato simulirali 50 iteracij s prometnimi obremenitvami P3. Ko nismo

začeli s prazno matriko Q, smo agentom pustili znanje iz predhodnih prometnih iteracij. Algoritem je sposoben s predhodnim znanjem že v prvi iteraciji uspešno krmiliti SSN, kar je razvidno s slike 5.20.

Vidimo, da so dosežene vrednosti povprečnih zamud na vozilo zadnjih 20 iteracij zelo primerljive, kar potrjuje tezo, da se je algoritem sposoben naučiti tudi s predhodnim znanjem, pridobljenim z različnim prometom. Z začetkom simuliranja brez predhodnega znanja ali z njim smo dosegli skoraj enake rezultate. To dokazuje, da se bo algoritem prilagodil in učinkovito krmilil tudi v še neznanih prometnih razmerah, za katere ne moremo predvideti, ali se bodo zgodile.



Slika 5.20: Povprečne zamude na vozilo v primeru prometnih obremenitev P3 z inicializacijo in brez nje
Figure 5.20: Average delay time per vehicle in case of traffic volume P3 with and without initialization

Iz analize konvergence lahko povzamemo, da je algoritem sposoben učinkovito krmiliti SSN v vseh treh preverjenih prometnih obremenitvah. V vseh primerih rezultati konvergirajo k optimalnim vrednostim. Dodatne analize parametra izkoriščanja znanja so pokazale, da pri prometnih obremenitvah P1 ni bistvene razlike med izbiro vrednosti 400 ali 600, pri prometnih obremenitvah P2 je boljša izbira vrednosti 400, medtem ko je pri prometnih obremenitvah P3 boljša izbira vrednosti 600. Rezultati kažejo, da je, če imamo prometne obremenitve na arteriji bistveno večje kot na stranskih smereh, boljša izbira nižje vrednosti parametra izkoriščanja znanja. Obratno pa velja, ko imamo netipične obremenitve za arterijo, tj. bolj uravnotežene.

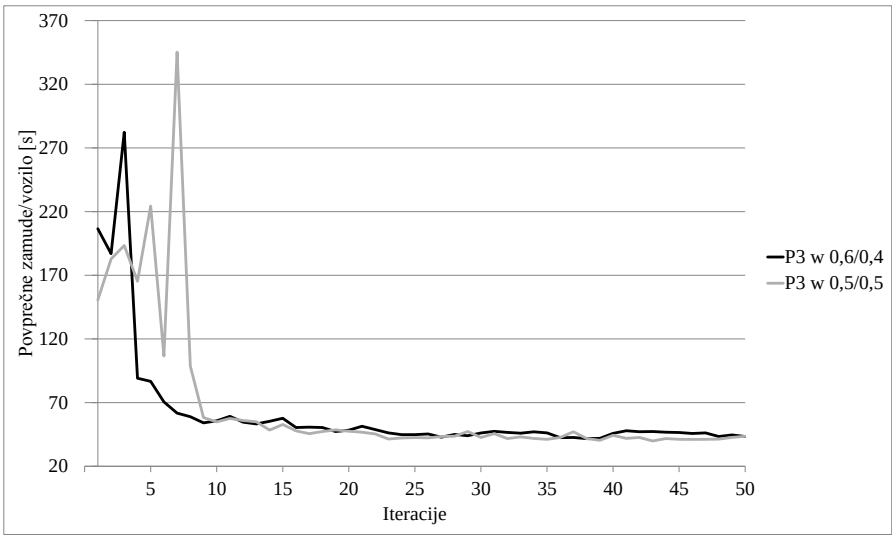
Pri dodatnih analizah komunikacije med agenti se je izkazalo, da je v vseh primerih bolje, če nimamo vključene dodatne komunikacije med agenti. Najmanjša razlika brez komunikacije in z njo je ob prometnih obremenitvah P1. Tudi ob prometnih obremenitvah P2 je razlika statistično neznačilna,

vendar je razvidno, da se agenti učijo hitreje brez komunikacije. Da dodatne komunikacije med agenti ne potrebujemo, pa najbolj potrjuje primer s prometnimi obremenitvami P3, kjer je razlika največja.

5.3.1.10 Analiza pomena uteži

V poglavju 4.5 Nagrade smo zapisali, da smo pasove na glavni in stranski smeri različno utežili. Za utež na glavnih smereh smo izbrali vrednost 0,6, medtem ko smo na stranski smeri izbrali vrednost 0,4. S tem smo dali prednost vozilom na glavni smeri oziroma smo favorizirali prvo fazo. Tako razmerje uteži smo sprva upoštevali za vse zgoraj navedene scenarije. Ker pa je razmerje med prometnimi obremenitvami glavnih in prečnih smeri pri prometnih obremenitvah P3 bistveno drugačno kot pri P1 in P2, smo analizirali rezultat z drugačnimi utežmi. Prometne obremenitve P3 so veliko bolj uravnotežene v smeri zahod–vzhod proti sever–jug. Da bi dodatno izboljšali rezultate pri najprometneje obremenjenem scenariju, smo enačili vrednosti uteži. Torej v enačbi 4.1 smo vrednosti 0,6 in 0,4 zamenjali z 0,5 in s tem izničili favoriziranje glavne smeri.

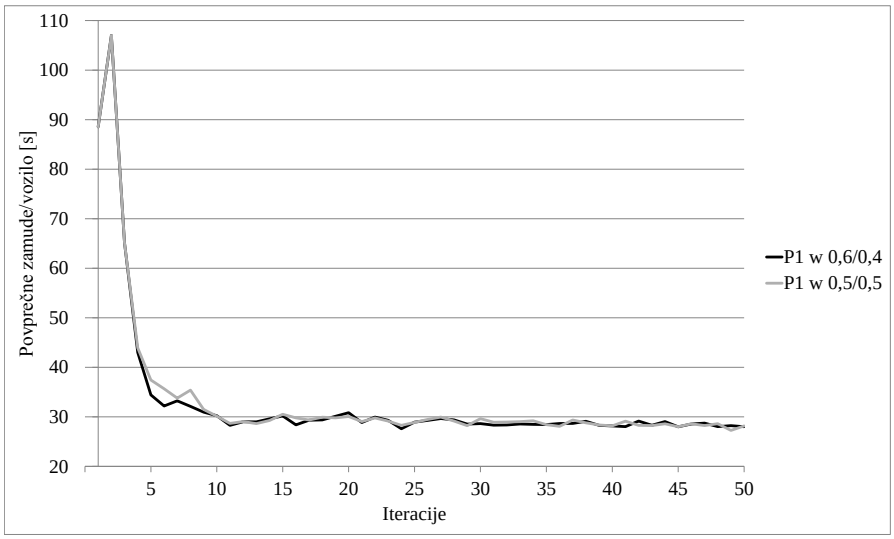
Na sliki 5.21 je primerjava konvergence algoritma za prometne obremenitve P3 z različnimi (P3 w 0,6/0,4) in enakimi utežmi (P3 w 0,5/0,5). Prva serija, prikazana s črno barvo, je enaka kot na sliki 5.20 (P3 z ini). Pri uteži 0,5 (siva linija) je razvidno, da je učenje v primerjavi s prvim setom upočasnjeno. Prav tako je v prvih 10 iteracijah večje nihanje. Od 10. do 30. iteracije je v obeh primerih vidno počasno izboljševanje rezultatov. Zadnjih 20 iteracij sta krivulji umirjeni in se gibata blizu optimalnim vrednostim. Povprečne vrednosti zadnjih 20 iteracij kažejo v korist drugemu primeru. V drugem primeru imamo za več kot 3 sekunde nižje povprečne zamude na vozilo. Da je boljša izbira z uravnoteženimi utežmi, nam potrjuje tudi test *T*. Razlika v korist drugemu primeru je statistično značilna (vrednost *p* je 1,23 E-05). Kljub počasnejšemu učenju je pri prometnih obremenitvah P3 smiselno, da uporabljamo uravnotežene uteži. Končne iteracije oziroma faza, ko agenti izkoriščajo le znanje, so pomembnejše od hitrosti učenja. Hitrost učenja oziroma simuliranja nekaj 10 iteracij take prometne mreže pri današnji strojni opremi ne predstavlja velike izgube časa. Za tako simuliranje uporablja sistem zelo majhno količino procesorskega časa.



Slika 5.21: Povprečne zamude na vozilo pri prometnih obremenitvah P3 z različnimi (0,6/0,4) in enakimi (0,5/0,5) utežmi

Figure 5.21: Average delay time per vehicle based on traffic volume P3 with different (0,6/0,4) and equal (0,5/0,5) weights

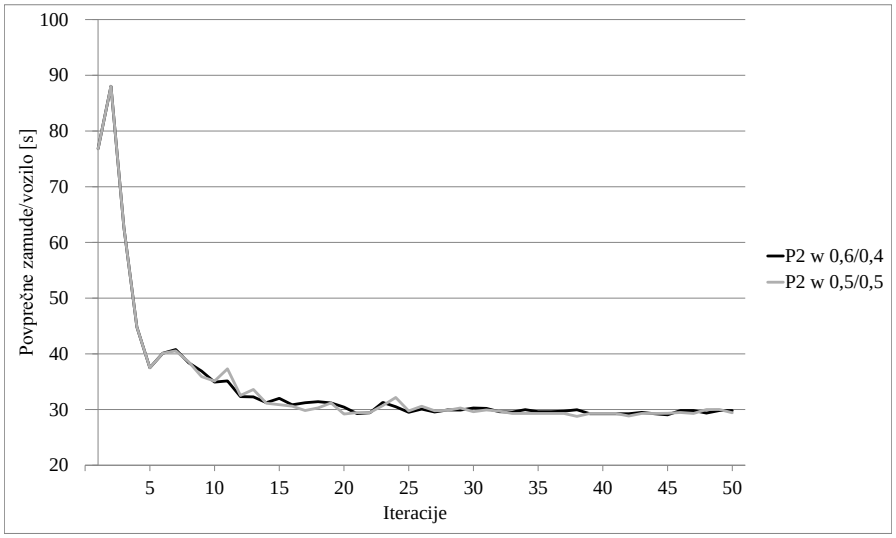
Na podlagi zadnje analize o pomenu uteži pri prometnih obremenitvah P3 smo izvedli še analizo za primer nenasičenih prometnih razmer P1. Na sliki 5.22 so rezultati z različnimi in enakimi utežmi. Razlika ni tako vidna kot pri predhodni analizi. Trend konvergence je pri obeh krivuljah zelo podoben. Krivulji se skoraj prekrivata celo v začetnih iteracijah, kar pomeni, da uteži nimajo bistvenega vpliva na hitrost učenja in učinkovitega krmiljenja. Slednje velja predvsem za zadnjih 20 iteracij, saj je povprečje obeh setov skoraj enako. Potrditev dobimo s testom *T*, ki pokaže, da razlika ni statistično značilna (vrednost *p* 0,62). Kot kaže, so agenti brez dodatnega faktorja pri nagrajevanju sposobni učinkovito krmiliti promet.



Slika 5.22: Povprečne zamude na vozilo ob prometnih obremenitvah P1 z različnimi (0,6/0,4) in enakimi (0,5/0,5) utežmi

Figure 5.22: Average delay time per vehicle based on traffic volume P1 with different (0,6/0,4) and equal (0,5/0,5) weights

Zelo primerljive rezultate kot pri prometnih obremenitvah P1 smo dobili za nasičene prometne razmere P2. Na sliki 5.23 je prav tako prikazana konvergenca z vrednostmi uteži 0,6 in 0,4 (črna linija) ter uravnoteženimi smermi (siva linija). Krivulji se skoraj prekrivata, še posebno v prvih 10 iteracijah. V nadaljnjih iteracijah je nihanje in odstopanje prav tako majhno. Razliko zadnjih 20 iteracij smo preverili s testom *T*, ki je potrdil, da razlika ni statistično značilna. Vrednost *p* je 0,06, kar pomeni, da je skoraj vseeno, ali nagrade posameznih smeri množimo z dodatnimi faktorji uteži.



Slika 5.23: Povprečne zamude na vozilo ob prometnih obremenitvah P2 z različnimi (0,6/0,4) in enakimi (0,5/0,5) utežmi

Figure 5.23: Average delay time per vehicle based on traffic volume P2 with different (0,6/0,4) and equal (0,5/0,5) weights

Iz analize pomena uteži ugotavljamo, da dodatna pomoč oziroma favoriziranje ne prinaša boljših končnih rezultatov. Pri prometnih obremenitvah P3 je celo slabše. V tem primeru se agenti nekoliko počasneje učijo, vendar so povprečne zamude v končnih iteracijah nižje kot z različnimi utežmi. Pri scenarijih P1 in P2 smo dokazali, da je razlika statistično neznačilna z različnimi ali enakimi utežmi. S tem povzemamo, da uporabo uteži lahko opustimo. S praktičnega vidika je to dobro, saj nam ni treba skrbeti za vhodne parametre glede na tip prometne mreže. Algoritem krmiljenja je brez uporabe uteži univerzalnejši in neodvisnejši.

5.3.1.11 Nastavitve algoritma

Preverili in testirali smo vedenje algoritma za različne kombinacije parametrov in prometne obremenitve. Na podlagi vseh opravljenih simulacij smo izbrali naslednje nastavitve algoritma:

- uporaba lokalne nagrade,
- faktor stopnje učenja $\alpha = 0{,}2$,
- faktor diskontiranja $\gamma = 0{,}8$,
- parameter raziskovanja $\varepsilon = 10$,
- parameter izkoriščanja znanja $n_\varepsilon = 400$ (P1 in P2) ali $n_\varepsilon = 600$ (P3),
- brez komunikacije med agenti LT1 in
- brez uporabe uteži.

Izbrane parametre smo uporabili za nadaljnjo primerjavo med predlaganim algoritmom z učenjem Q in klasičnim prometno odvisnim krmiljenjem.

5.3.2 Primerjava krmiljenja z učenjem Q in klasičnim prometno odvisnim krmiljenjem

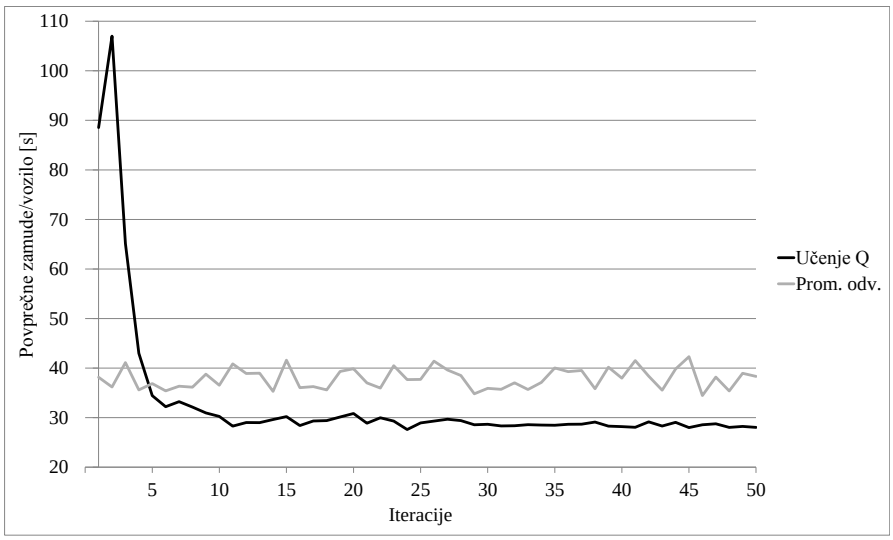
Po določitvi optimalnih parametrov oziroma nastavitev algoritma smo naredili primerjavo rezultatov med algoritmom učenja Q in klasičnim prometno odvisnim krmiljenjem. Primerjavo med obema vrstama krmiljenja smo prikazali podobno kot pri predhodni analizi konvergence, s povprečnimi zamudami na vozilo ter dodatno še s številom prepeljanih vozil skozi mrežo. Število prepeljanih vozil skozi mrežo v simulacijskem obdobju nam dodatno kaže, kolikšna je uspešnost krmiljenja. Nadalje smo za primerjavo vključili še dodatne prometne kazalce, in sicer povprečno hitrost, število ustavljanj na vozilo, skupne zamude zaradi mirovanja in potovalne čase na cestni arteriji.

Primerjali smo povprečne vrednosti in standardni odklon vseh izhodnih rezultatov za zadnjih 20 iteracij. Zadnje iteracije predstavljajo rezultate, ko je algoritem že naučen. Statistično značilno

razliko rezultatov z algoritmom Q in klasičnim prometnim krmiljenjem smo preverili s testom *T*. Stopnjo značilnosti (vrednost *p*) smo uporabili 0,05.

5.3.2.1 Primerjava v nenasičenih prometnih razmerah P1

Primerjavo med algoritmom učenja Q in klasičnim prometno odvisnim krmiljenjem smo najprej naredili za nenasičene prometne razmere P1. Za primerjavo smo tudi za klasično prometno odvisno krmiljenje izvedli 50 iteracij. Na sliki 5.24 so povprečne zamude na vozilo za oba tipa krmiljenja. Krmiljenje s pomočjo učenja Q je prikazano s črno linijo (Učenje Q), klasično prometno odvisno krmiljenje pa s sivo linijo (Prom. odv.). S slike je dobro razvidno, da so začetne zamude v prvih iteracijah boljše pri klasičnem prometno odvisnem krmiljenju, vendar se razmere že po 5. iteraciji obrnejo v prid novemu krmiljenju. Pri klasičnem krmiljenju so zamude med 35 in 42 sekundami. Klasično krmiljenje dosega zelo primerljive rezultate skoraj v vseh iteracijah. Je pa v primerjavi z učenjem Q nihanje krivulje večje v zadnjih 20 iteracijah. Zadnjih 20 iteracij je nihanje črne krivulje zelo blago, kar dokazuje, da se kljub stohastični naravi prometa izjemno dobro prilagaja spreminjajočim razmeram. Povprečne zamude na vozilo se pri učenju Q, ko je algoritem naučen, gibajo okoli 28 sekund. Tako lahko zapišemo, da je krmiljenje z učenjem Q v primerjavi s klasičnim prometno odvisnim krmiljenjem učinkovitejše v nenasičenih prometnih razmerah.

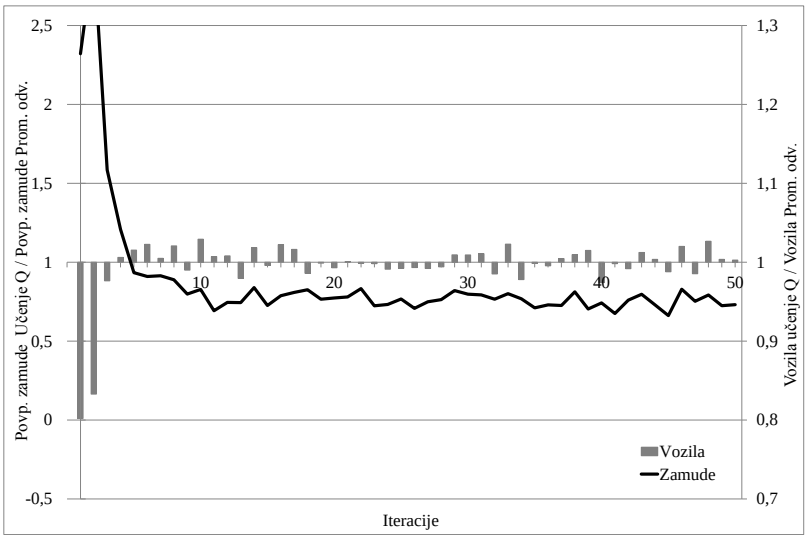


Slika 5.24: Primerjava povprečnih zamud na vozilo med krmiljenjem z učenjem Q in klasičnim prometno odvisnim krmiljenjem (nenasičeni prometni tok P1)

Figure 5.24: Average delay time per vehicle comparison between Q learning algorithm and actuated traffic light optimization (saturated traffic flow P1)

Na sliki 5.25 je razmerje povprečnih zamud na vozilo ter število prepeljanih vozil skozi mrežo med učenjem Q in klasično prometno odvisnem krmiljenjem. Na sliki predstavlja črna linija razmerje povprečnih zamud na vozilo. Če je linija nad številom 1 (leve navpične skale), pomeni, da so zamude pri učenju Q večje v primerjavi s klasičnim krmiljenjem. Če je linija pod 1, pa obratno, kar pomeni, da je krmiljenje z učenjem Q boljše. Sivi stolpci predstavljajo razmerje števila prepeljanih vozil skozi mrežo. Pri vozilih velja, da stolpci nad številom 1 (desne navpične skale) pomenijo več prepeljanih vozil, stolpci pod številom 1 pa manj prepeljanih vozil v primerjavi z novim krmiljenjem.

Slika 5.25 nam nazorno kaže, da je v prvih 5 iteracijah poleg večjih zamud še enkrat manj prepeljanih vozil skozi mrežo ob novem krmiljenju. To pomeni, da je v prvih iteracijah popolnoma zgrešeno krmiljenje, saj so skoraj na celotni mreži zastoji, ki onemogočajo prihod novim vozilom v omrežje. S slike lahko razberemo, da je razlika v številu prepeljanih vozil v končnih iteracijah minimalna ± nekaj odstotkov. V večini zadnjih 20 iteracij je več prepeljanih vozil pri krmiljenju z algoritmom Q. Z dodatnim prikazom razmerja števila prepeljanih vozil skozi mrežo smo želeli dokazati, da je kljub manjšim zamudam število prepeljanih vozil skozi mrežo enako ali večje v primerjavi s klasičnim krmiljenjem. Na podlagi prikazanega števila prepeljanih vozil lahko še dodatno potrdimo, da je mreža z novim krmiljenjem prepustnejša.



Slika 5.25: Primerjava med krmiljenjem z učenjem Q in klasičnim prometno odvisnim krmiljenjem na podlagi povprečnih zamud na vozilo ter s številom prepeljanih vozil skozi mrežo (nenasičeni prometni tok P1)

Figure 5.25: Comparison between Q learning algorithm and actuated traffic light optimization based on average delay time per vehicle and number of vehicles that have left the network (saturated traffic flow P1)

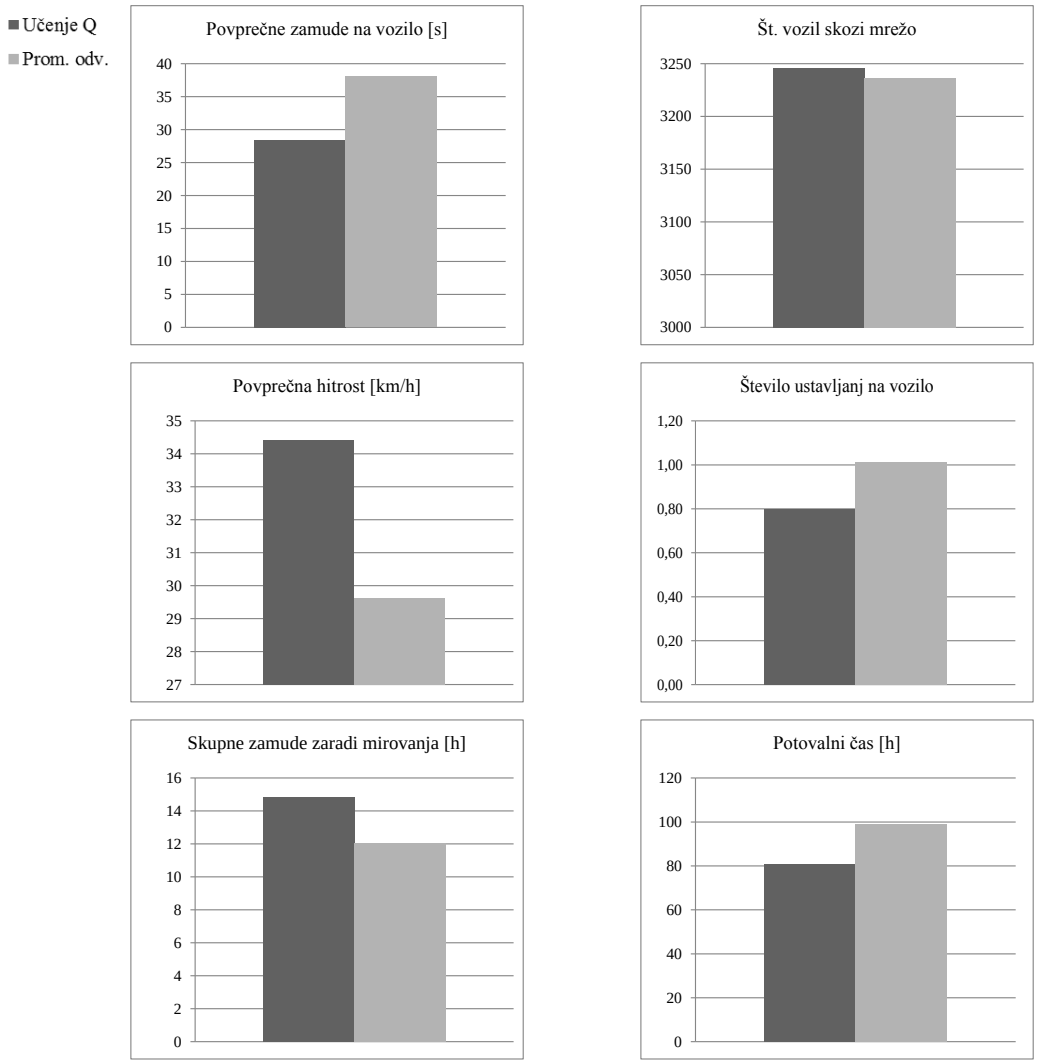
Poleg glavnih dveh kazalcev, povprečne zamude na vozilo in števila vozil, ki so prepeljala skozi mrežo, smo primerjali še povprečno hitrost na mreži, število ustavljanj na vozilo, skupne zamude

zaradi mirovanja in potovalni čas vseh vozil na mreži oziroma katera so zapustila mrežo. Primerjava v naslednji preglednici 5.4 in sliki 5.26 je narejena za zadnjih 20 iteracij s prometnimi obremenitvami P1. Predstavljeni izhodni rezultati za nenasičene prometne razmere kažejo, da smo s predlagano alternativno metodo krmiljenja SSN uspešno zmanjšali povprečne zamude na vozilo za 25,2 %, povečali skupno število vozil, ki so zapustila mrežo, za 0,3 %, zvišali povprečno hitrost vozil za 16,2 %, zmanjšali število ustavljanj za 21,1 %, in skrajšali potovalni čas za 18,1 %. Krmiljenje z algoritmom Q se izkaže za slabše pri skupnih zamudah zaradi mirovanja, saj zamude narastejo za 23,5 %. Vendar moramo biti pri tem poslabšanju pozorni na število ustavljanj, saj se nam zmanjša za več kot 20 odstotkov. To pomeni, da agenti zadržijo na glavni smeri večje število vozil in jih nato kot skupino spustijo po arteriji. S tem algoritem ustvari boljše razmere na arteriji in učinkovito podaljša zeleni čas na glavni smeri. Lahko rečemo, da s takim krmiljenjem agent ustvari dobro linijsko koordinacijo.

Preglednica 5.4: Rezultati simulacij za prometne obremenitve P1

Table 5.4: Simulation results based on traffic volume P1

Način krmiljenja		Povprečne zamude/voz [s]	Št. vozil skozi mrežo	Povprečna hitrost [km/h]	Število ustavljanj na vozilo	Skupne zamude zaradi mirovanja [h]	Potovalni čas [h]
Učenje Q	Povprečje	28,46	3246	34,42	0,80	14,85	80,92
	Standardni odklon	0,36	3	0,19	0,02	0,30	0,46
Prometno odvisno	Povprečje	38,06	3236	29,62	1,01	12,03	98,83
	Standardni odklon	2,18	43	0,73	0,04	0,59	3,32
Učenje Q/Prom.odv.	Razlika	-9,60	10	4,80	-0,21	2,83	-17,91
	Razlika [%]	-25,2%	0,3%	16,2%	-21,1%	23,5%	-18,1%



Slika 5.26: Rezultati simulacij za prometne obremenitve P1

Figure 5.26: Simulation results based on traffic volume P1

Za razlike v rezultatih med krmiljenjema smo preverili statistično značilnost s testom *T*. Statistična značilnost razlik med učenjem Q in klasičnim krmiljenjem je v preglednici 5.5. Statistična značilnost je bila potrjena za vse izhodne rezultate, razen pri številu prepeljanih vozil skozi mrežo. Razlika v številu prepeljanih vozil skozi mrežo sicer ni statistično značilna, vendar je število prepeljanih vozil skozi mrežo večje pri učenju Q. V preglednici 5.5 je pri skupnih zamudah zaradi mirovanja označena primerjava z zvezdico (*), kar se nanaša na prej omenjeno poslabšanje, vendar skupaj z zmanjšanim številom ustavljanj dosežemo boljšo linijsko koordinacijo na arteriji.

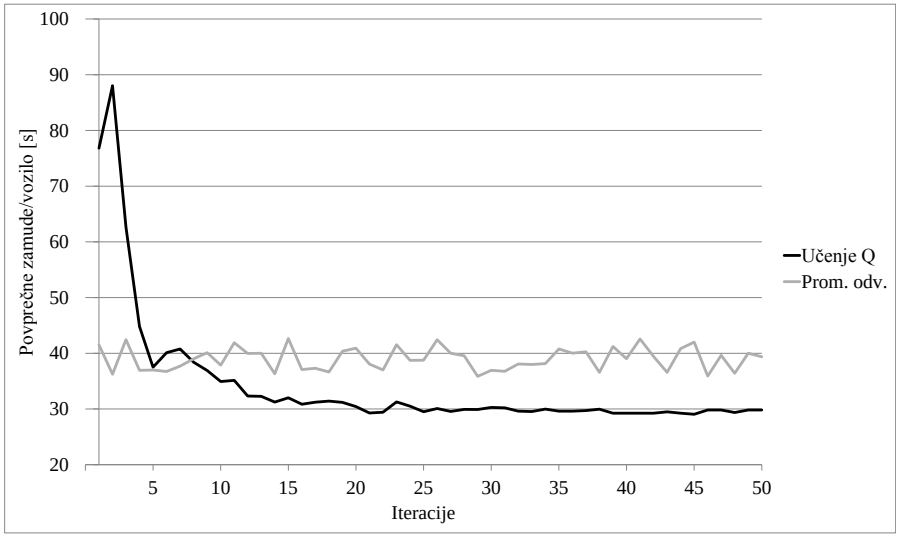
Preglednica 5.5: Rezultati testa *T* za prometne obremenitve P1

Table 5.5: *T*-test results based on traffic volume P1

Način krmiljenja	Stat.	Povprečne zamude/voz [s]	Št. vozil skozi mrežo	Povprečna hitrost [km/h]	Število ustavljanj na vozilo	Skupne zamude zaradi mirovanja [h]	Potovalni čas [h]
Učenje Q/Prom.odv.	Vrednost *p*	1,93E-14	7,34E-01	7,61E-19	1,08E-16	1,32E-17	3,53E-16
	Primerjava	Boljše	Boljše	Boljše	Boljše	Slabše*	Boljše

5.3.2.2 Primerjava v nasičenih prometnih razmerah P2

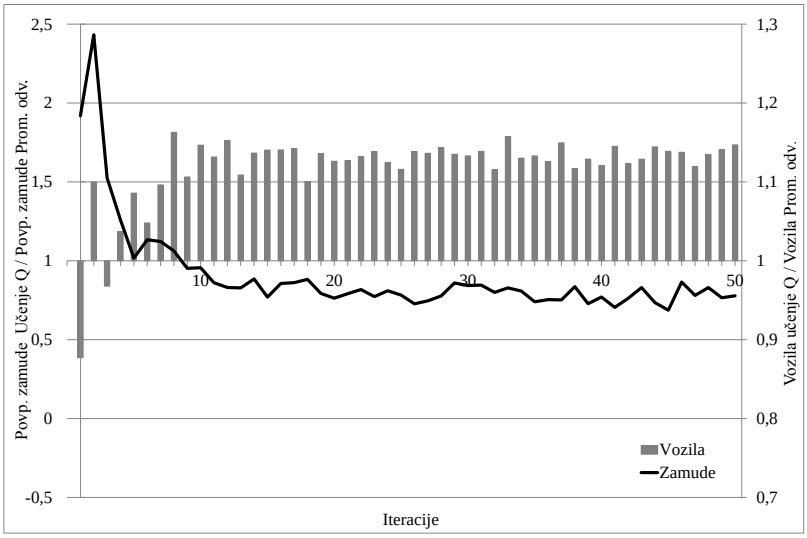
Prav tako kot za nenasičene prometne razmere smo primerjavo naredili med algoritmom učenja Q in klasičnim prometno odvisnim krmiljenjem za nasičene prometne razmere P2. Na sliki 5.27 so prikazane povprečne zamude na vozilo za prometne obremenitve. Zamude z učenjem Q so prikazane s črno, medtem ko so s sivo barvo prikazane zamude pri klasičnem prometnem krmiljenju. Podobno kot pri P1 so tudi pri P2 začetne zamude slabše pri krmiljenju s pomočjo učenja Q. Razmere se v primerjavi z nenasičenimi razmerami obrnejo v prid novemu krmiljenju nekoliko kasneje, pri 10. iteraciji. Zamude se vseh 50 iteracij pri klasičnem krmiljenju gibajo med 36 in 44 sekundami. Kljub temu da so zamude s klasičnim krmiljenjem zelo primerljive v vseh iteracijah so pri učenju Q razlike med iteracijami v zadnjih 20 iteracijah bistveno manjše. S slike vidimo, da je nihanje krivulje zamud zelo majhno. Povprečne vrednosti zamud so v zadnjih iteracijah nekje pri 30 sekundah. Iz prikazanega je razvidno, da so rezultati povprečnih zamud boljši z učenjem Q kot s klasičnim prometno odvisnim krmiljenjem v nasičenih prometnih razmerah.



Slika 5.27: Primerjava povprečnih zamud na vozilo med krmiljenjem z učenjem Q in klasičnim prometno odvisnim krmiljenjem (nasičeni prometni tok P2)

Figure 5.27: Average delay time per vehicle comparison between Q learning algorithm and actuated traffic light optimization (over-saturated traffic flow P2)

Na sliki 5.28 je razmerje povprečnih zamud na vozilo ter število prepeljanih vozil skozi mrežo med učenjem Q in klasično prometno odvisnim krmiljenjem v nasičenih prometnih razmerah P2. Črna linija predstavlja razmerje povprečnih zamud na vozilo, sivi stolpci pa razmerje med številom prepeljanih vozil skozi mrežo. Linija pod vrednostjo 1 pomeni, da so povprečne zamude na vozilo manjše z novim krmiljenjem. Stolpci nad številom 1 pomenijo, da je število prepeljanih vozil skozi mrežo večje s predlaganim algoritmom učenja Q. S slike je razvidno, da so zamude v prvih 10 iteracijah večje z novim krmiljenjem, vendar je samo v nekaj iteracijah število prepeljanih vozil skozi mrežo manjše. Najmanj primerno krmiljenje z učenjem Q lahko vidimo v prvi in tretji iteraciji. Poleg večjih zamud imamo tudi manj prepeljanih vozil skozi mrežo. V 5. iteraciji se zamude v obeh primerih izenačijo, pri učenju Q imamo celo več prepeljanih vozil, vendar se nato v naslednjih nekaj iteracijah razmere ponovno poslabšajo. Po 10. iteraciji so zamude konstantno manjše v primerjavi s klasičnim algoritmom. Prav tako imamo več prepeljanih vozil skozi mrežo za več kot 10 %. V zadnjih 20 iteracijah, ko je algoritem krmiljenja naučen, vidimo, da so zamude manjše med 20 in 35 % in število prepeljanih vozil večje za 11 do 15 %. V primerjavi z nenasičenimi razmerami (slika 5.25), potrebuje algoritem v prometnih razmerah P2 več iteracij za učenje, vendar je na koncu razlika večja in očitnejša. Sklepamo lahko, da je kljub večjemu prometnemu volumnu na arteriji z učenjem Q zmogljivost prometne mreže večja kot s klasičnim krmiljenjem.



Slika 5.28: Primerjava med krmiljenjem z učenjem Q in klasičnim prometno odvisnim krmiljenjem na podlagi povprečnih zamud na vozilo in številom prepeljanih vozil skozi mrežo (nasičeni prometni tok P2)

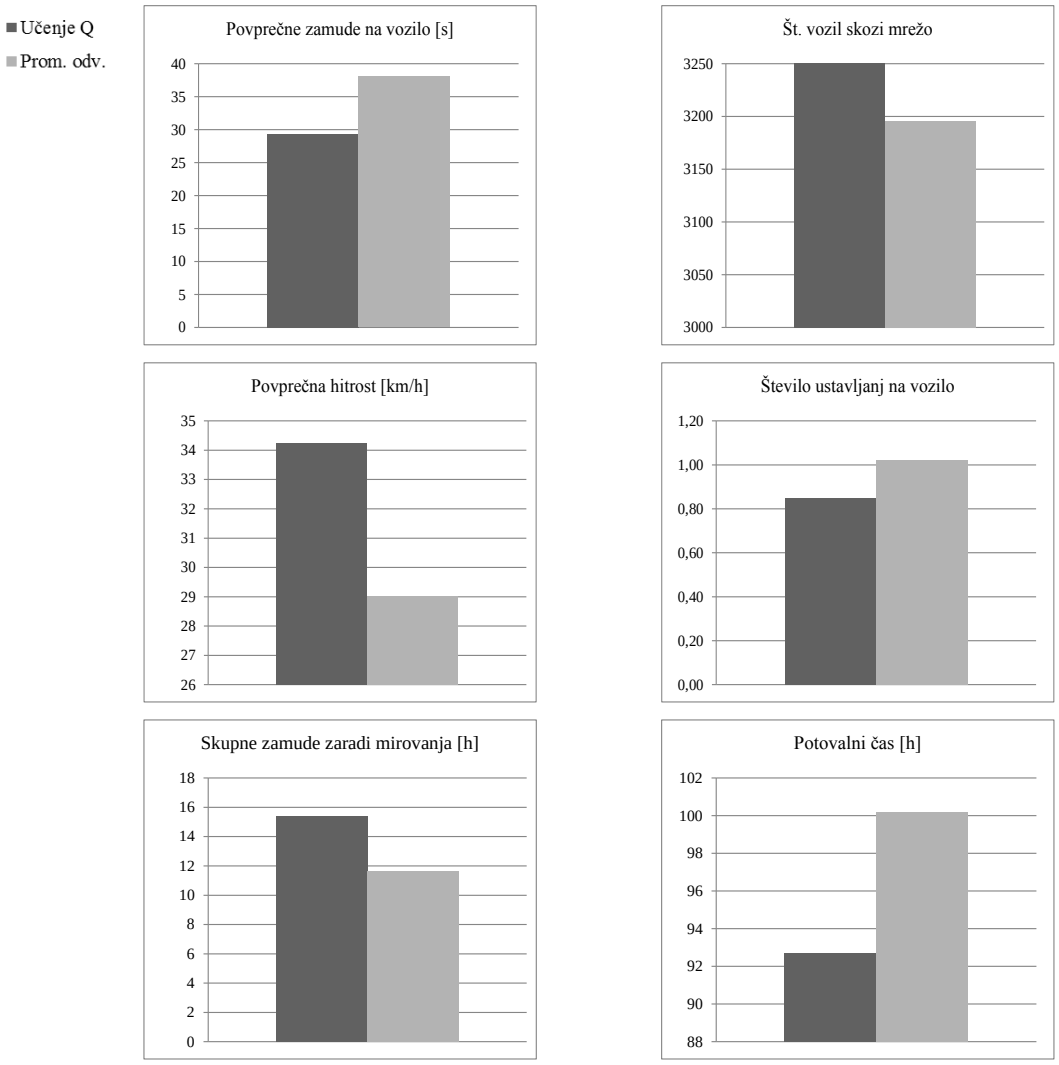
Figure 5.28: Comparison between Q learning algorithm and actuated traffic light optimization based on average delay time per vehicle and number of vehicles that have left the network (over-saturated traffic flow P2)

Vsi izhodni rezultati simulacij v nasičenih prometnih razmerah P2 so v preglednici 5.6 in na sliki 5.29. Prikazane vrednosti veljajo za zadnjih 20 iteracij. Prikazani rezultati so primerljivi s predhodnimi rezultati simulacij z obremenitvami P1. Z algoritmom učenja Q smo v primerjavi s klasičnim krmiljenjem zmanjšali povprečne zamude na vozilo za 22,8 %, povečali skupno število vozil, ki so zapustila mrežo za 13,9 %, zvišali povprečno hitrost vozil za 18,0 %, zmanjšali število ustavljanj za 17,0 %, in skrajšali potovalni čas za 7,5 %. Z učenjem Q se nam poslabšajo rezultati glede skupnih zamud zaradi mirovanja. Zamude se nam zvečajo za 32,3 %. Vendar tako kot pri P1 imamo občutno zmanjšanje števila ustavljanj. Iz obeh kazalcev skupaj lahko sklepamo, da agenti zadržijo na glavni smeri večje število vozil ter jih nato v konvoju spustijo po arteriji. Kot kaže, imamo z novim krmiljenjem manjše število preklopov med fazama. S tem so dosežene boljše razmere linijske koordinacije v primerjavi s klasičnim krmiljenjem.

Preglednica 5.6: Rezultati simulacij za prometne obremenitve P2

Table 5.6: Simulation results based on traffic volume P2

Način krmiljenja		Povprečne zamude/voz [s]	Št. vozil skozi mrežo	Povprečna hitrost [km/h]	Število ustavljanj na vozilo	Skupne zamude zaradi mirovanja [h]	Potovalni čas [h]
Učenje Q	Povprečje	29,39	3640	34,25	0,85	15,37	92,72
	Standardni odklon	0,32	4	0,13	0,01	0,08	0,32
Prometno odvisno	Povprečje	38,06	3196	29,02	1,02	11,62	100,21
	Standardni odklon	2,18	25	0,20	0,02	0,35	1,25
Učenje Q/Prom.odv.	Razlika	-8,67	444	5,22	-0,17	3,75	-7,49
	Razlika [%]	-22,8%	13,9%	18,0%	-17,0%	32,3%	-7,5%



Slika 5.29: Rezultati simulacij za prometne obremenitve P1

Figure 5.29: Simulation results based on traffic volume P1

Statistično značilnost med razlikami izhodnih rezultatov med obema načinom krmiljenja smo preverili s testom *T* (preglednica 5.7). V preglednici vidimo, da je bila vrednost *p* za vse kazalce manj kot 0,05, kar pomeni potrditev statistične značilnosti. Enako kot pri prejšnji analizi s testom *T* smo tudi v naslednji preglednici označili primerjavo za skupne zamude zaradi mirovanja z zvezdico (*). Oznaka se nanaša na pogojno poslabšanje rezultatov, saj v povezavi s številom ustavljanj ne velja, da je zaradi tega krmiljenje slabše.

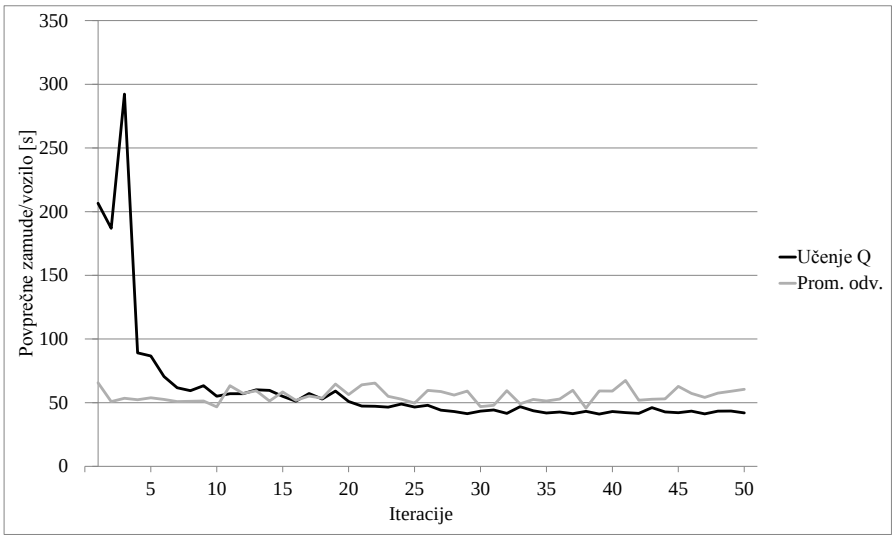
Preglednica 5.7: Rezultati testa *T* za prometne obremenitve P2

Table 5.7: *T*-test results based on traffic volume P2

Način krmiljenja	Stat.	Povprečne zamude/voz [s]	Št. vozil skozi mrežo	Povprečna hitrost [km/h]	Število vstavljanj na vozilo	Skupne zamude zaradi mirovanja [h]	Potovalni čas [h]
Učenje Q/Prom.odv.	Vrednost *p*	1,27E-13	1,50E-41	1,50E-41	1,77E-20	1,37E-22	1,97E-17
	Primerjava	Boljše	Boljše	Boljše	Boljše	Slabše*	Boljše

5.3.2.3 Primerjava v nasičenih prometnih razmerah P3

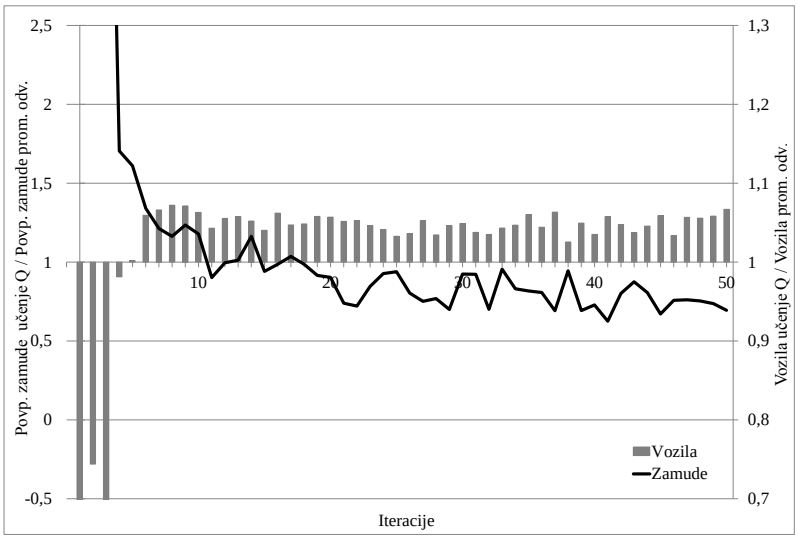
Primerjava zamud v nasičenih prometnih razmerah P3 med učenjem Q in klasičnim prometno odvisnim krmiljenjem je predstavljena na sliki 5.30. Kot v prejšnjih dveh primerih s prometnimi obremenitvami P1 in P2 potrebuje algoritem učenja Q (črna linija) določeno število iteracij, da se nauči pravilne strategije krmiljenja. Ob prometnih obremenitvah P3 potrebuje v primerjavi z ostalima dvema največ časa (iteracij). V prvih iteracijah so tudi zamude v primerjavi s klasičnim krmiljenjem bistveno večje. Zamude so tudi več kot 200 sekund/vozilo, kar kaže na popolnoma zgrešeno krmiljenje. Med 10. in 19. iteracijo se v nekaj primerih krivulje prekrižajo, vendar se šele po 19. iteraciji situacija obrne v prid učenju Q, tako da ostajajo zamude do konca nižje. Med 19. in 30. iteracijo je viden napredek pri zmanjševanju zamud. Po 30. iteraciji je krivulja učenja Q v primerjavi s klasičnim krmiljenjem veliko stabilnejša kot pri klasičnem krmiljenju. Pri slednjem so vrednosti med 46 in 65 sekundami. Povprečna vrednost zadnjih 20 iteracij pri učenju Q je slabih 43 sekund. Na podlagi vseh treh prometnih obremenitev lahko sklepamo, da je potreben čas učenja pogojen s količino prometa. Več prometa imamo in bolj je uravnotežen med obema smerema, več raznolikih situacij imamo. To pa posledično vpliva na čas učenja. Kljub temu je krmiljenje s predlaganim algoritmom boljše.



Slika 5.30: Primerjava povprečnih zamud na vozilo med krmiljenjem z učenjem Q in klasičnim prometno odvisnim krmiljenjem (nasičeni prometni tok P3)

Figure 5.30: Average delay time per vehicle comparison between Q learning algorithm and actuated traffic light optimization (over-saturated traffic flow P3)

Za največje prometne obremenitve P3 smo prav tako prikazali razmerje povprečnih zamud na vozilo ter število prepeljanih vozil skozi mrežo med novim krmiljenjem in klasičnim prometno odvisnim krmiljenjem. Na sliki 5.31 je primerjava zamude prikazana s krivuljo, primerjava števila vozil pa s stolpci. V iteracijah, kjer je krivulja nižja od ena, pomeni manjše zamude pri učenju Q. S predlaganim algoritmom učenja Q je število prepeljanih vozil skozi mrežo večje v iteracijah, ko so stolpci nad vrednostjo ena. V iteraciji 11 se prvič zgodi, da algoritem zmanjša zamude v primerjavi s klasičnim krmiljenjem, vendar se že v naslednji iteraciji rezultati ponovno poslabšajo. Od 19. iteracije dalje je razmerje povprečnih zamud na vozilo med učenjem Q in klasičnim algoritmom konstantno manj kot 1. Največja razlika v prid učenju Q je vidna v zadnjih iteracijah. Število prepeljanih vozil skozi mrežo v začetnih iteracijah je bistveno manjše z učenjem Q. V začetnih iteracijah je tudi krivulja zamud veliko nad 2, kar pomeni, da so zastoji na mreži. Taki zastoji oziroma kolone onemogočajo vstop novim vozilom v mrežo. Od 5. iteracije naprej pa je razmerje števila vozil skozi mrežo vedno nad številom 1. Nad 1 je celo v primerih med 5. in 19. iteracijo, ko so zamude večje pri učenju Q. Tudi ob večjih in bolj uravnoteženih prometnih obremenitvah algoritem učenja Q krmili SSN učinkoviteje kot klasično prometno odvisno krmiljenje.



Slika 5.31: Primerjava med krmiljenjem z učenjem Q in klasičnim prometno odvisnim krmiljenjem na podlagi povprečnih zamud na vozilo ter števila prepeljanih vozil skozi mrežo (nasičeni prometni tok P3)

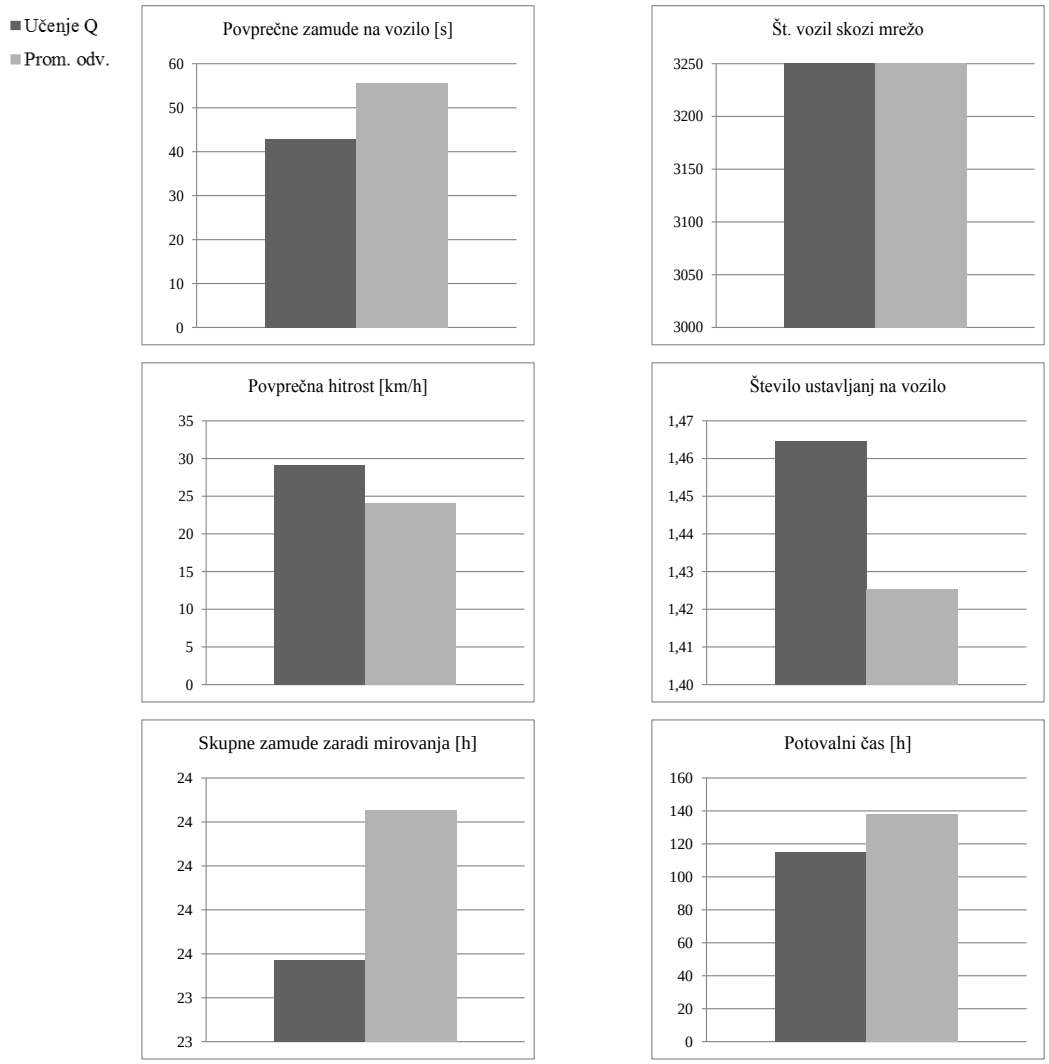
Figure 5.31: Comparison between Q learning algorithm and actuated traffic light optimization based on average delay time per vehicle and number of vehicles that have left the network (over-saturated traffic flow P3)

Preglednica 5.8 prikazuje vrednosti vseh izhodnih rezultatov, ki veljajo za zadnjih 20 iteracij ob prometnih obremenitvah P3. V primerjavi s klasičnimi krmiljenjem so z učenjem Q zmanjšane povprečne zamude na vozilo za 22,9 %, povečano je skupno število vozil, ki so zapustila mrežo, za 5,2 %, povprečna hitrost vozil na mreži je večja za 21,1 %, skupne zamude zaradi mirovanja so nižje za 2,8 % in za 16,8 % je skrajšan potovalni čas. Število ustavljanj se nam pri učenju Q zveča za 2,8 %. Za razliko od prometnih obremenitev P1 in P2 je skupna zamuda zaradi mirovanja nižja z učenjem Q kot pri prometno odvisnem krmiljenju. Spomnimo, da je bila razlika pri nižjih prometnih intenzitetah ravno obratna. Menimo, da je omenjena razlika nastala zato, ker je razmerje med glavnim in stranskim prometnim tokom v primeru P3 bistveno bolj uravnoteženo in je težje ustvariti linijsko koordinacijo na arteriji.

Preglednica 5.8: Rezultati simulacij za prometne obremenitve P3

Table 5.8: Simulation results based on traffic volume P3

Način krmiljenja		Povprečne zamude/voz [s]	Št. vozil skozi mrežo	Povprečna hitrost [km/h]	Število ustavljanj na vozilo	Skupne zamude zaradi mirovanja [h]	Potovalni čas [h]
Učenje Q	Povprečje	42,90	4078	29,12	1,46	23,57	115,15
	Standardni odklon	1,53	7	0,50	0,05	1,15	1,98
Prometno odvisno	Povprečje	55,66	3878	24,06	1,43	24,26	138,36
	Standardni odklon	5,41	36	1,21	0,10	1,47	6,84
Učenje Q/Prom.odv.	Razlika	-12,77	201	5,07	0,04	-0,68	-23,21
	Razlika [%]	-22,9%	5,2%	21,1%	2,8%	-2,8%	-16,8%



Slika 5.32: Rezultati simulacij za prometne obremenitve P3

Figure 5.32: Simulation results based on traffic volume P3

Preglednica 5.9 prikazuje rezultate testa *T* za vse preverjene izhodne rezultate med obema načinoma krmiljenja. Za povprečne zamude na vozilo, število vozil skozi mrežo, povprečno hitrost in potovalni čas lahko iz preglednice razberemo, da so potrjeno boljši, saj je v vseh primerih vrednost *p* manjša kot 0,05. Razlika v številu ustavljanj v korist prometno odvisnega krmiljenja ni statistično značilna, za kar lahko zapišemo, da rezultat z algoritmom Q ni slabši. Enako velja pri skupnih zamudah zaradi mirovanja, saj je vrednost p = 1,10 E-01.

Preglednica 5.9: Rezultati testa *T* za prometne obremenitve P3

Table 5.9: *T*-test results based on traffic volume P3

Način krmiljenja	Stat.	Povprečne zamude/voz [s]	Št. vozil skozi mrežo	Povprečna hitrost [km/h]	Število ustavljanj na vozilo	Skupne zamude zaradi mirovanja [h]	Potovalni čas [h]
Učenje Q/Prom.odv.	Vrednost *p*	9,15E-10	7,12E-15	2,03E-15	7,80E-02	1,10E-01	8,68E-13
	Primerjava	Boljše	Boljše	Boljše	Ni razlike	Ni razlike	Boljše

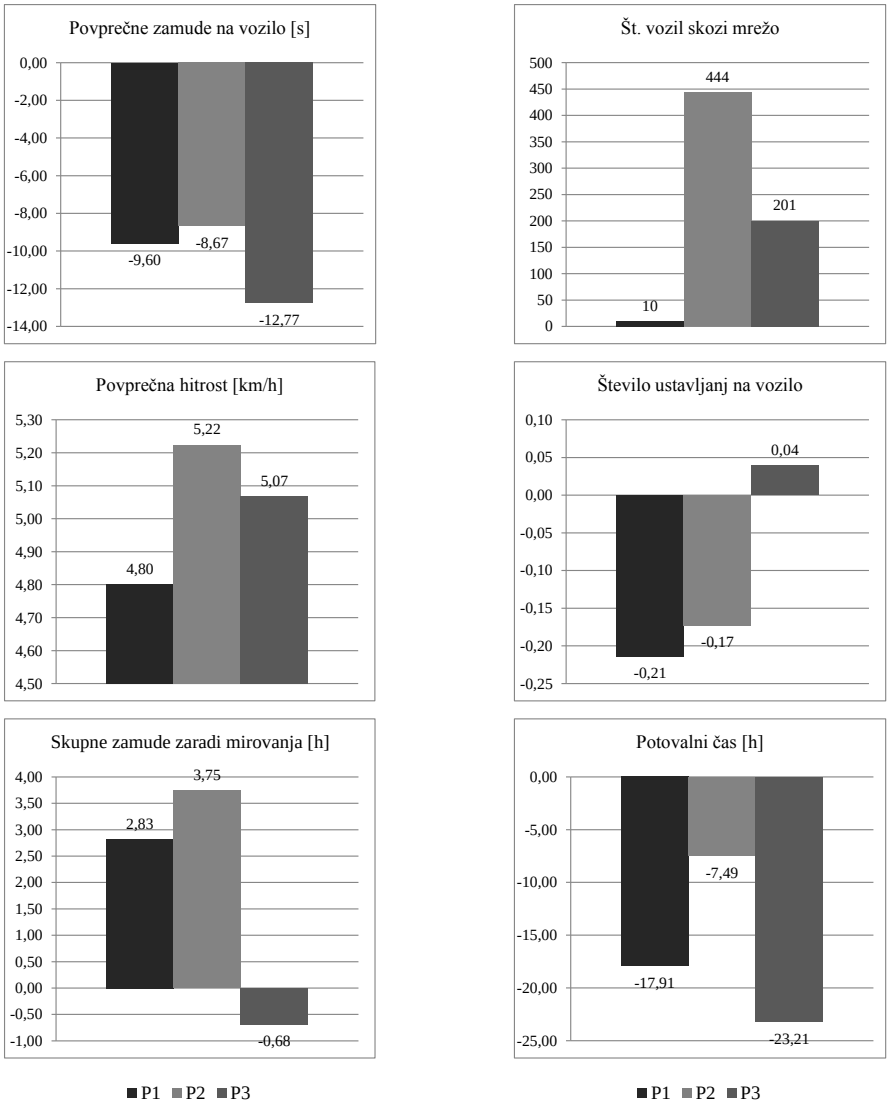
5.3.2.4 Povzetek rezultatov primerjave

Rezultati testa *T* v preglednicah (preglednica 5.5, preglednica 5.7 in preglednica 5.9) kažejo, da za vse tri prometne obremenitve dosežemo z učenjem Q v primerjavi s prometno odvisnim krmiljenjem SSN boljše rezultate skoraj v vseh primerjanih merilih. Edine izjeme so skupne zamude zaradi mirovanja pri P1 in P2, kjer se nam te zvišajo. Vendar si to v teh dveh primerih lahko razložimo tako, da so s tako koordinacijo dosežene boljše razmere za linijsko koordinacijo. Tretja izjema je še poslabšanje rezultatov v številu ustavljanj na vozilo ob prometnih obremenitvah P3. V tem primeru se nam število ustavljanj za malenkost zveča, vendar razlika z 1,43 na 1,46 ni statistično značilna. To pomeni, da ne poslabšamo ali izboljšamo merila ustavljanj.

Naslednja primerjava uspešnosti predlaganega algoritma krmiljenja s prometnimi obremenitvami P1, P2 in P3 je prikazana na sliki 5.33. Prikazana je razlika med krmiljenjem z učenjem Q in prometno odvisnim krmiljenjem. Povzetek primerjanih meril je narejen za povprečne zamude, število vozil skozi mrežo, povprečno hitrost, število ustavljanj na vozilo, skupne zamude zaradi mirovanja in potovalni čas. Glede na sliko 5.33 se zdi, da ni neposredne povezave med prometnimi obremenitvami oziroma skupnim številom vozil (slika 5.6) in razlikami v povprečnih zamudah na vozilo. Podrobnejša analiza slike 5.33 in števila vozil skozi mrežo razkrije, da je zmanjšanje povprečnih zamud na vozilo odvisno od števila vozil, ki so dosegla cilj. Npr. ob prometnih obremenitvah P2 z učenjem Q prepeljemo bistveno več vozil skozi mrežo kot s klasičnim krmiljenjem. V nobenem od treh primerov (P1, P2 in P3) ni ostalo več kot nekaj 10 vozil zunaj mreže zaradi daljših zajezitvenih dolžin. Seveda imamo tukaj v mislih samo končne iteracije, ko agenti izkoriščajo svoje znanje in ne preizkušajo več. Na drugi strani pa je s prometno odvisnim krmiljenjem v primerih P2 in P3 ostalo od 100 do 300 vozil zunaj mreže. To pomeni, da krmiljenje ni bilo dovolj učinkovito za servisiranje tako velike količine prometa. Pojasniti je treba razliko med prometnimi obremenitvami na sliki 5.6 in številom vozil skozi mrežo v preglednicah (preglednica 5.5, preglednica 5.7 in preglednica 5.9). Število vozil predstavlja vsa vozila, ki so med iteracijo zapustila mrežo, do številk na sliki s

prometnimi obremenitvami pa je treba upoštevati še vsa vozila, ki so bila tik pred koncem simulacije še na mreži. Preostalo število vozil na mreži se je gibalo cca. 300 vozil, odvisno od primera.

Razlike v povprečnih hitrostih so bile v vseh intenzitetah prometa boljše s predlaganim algoritmom krmiljenja. Največja razlika je bila pri P2, vendar lahko razberemo iz vrednosti na sliki, da je razlika v vseh treh primerih cca. 5 km/h. Z učenjem Q se je pri P1 in P2 zmanjšalo število ustavljanj na vozilo, pri prometnih obremenitvah P3 pa nekoliko zvečalo. A razlika je statistično značilna samo v prvem primeru, v drugih dveh pa ne. Tako lahko zapišemo, da se je v prvem primeru stanje izboljšalo, v drugih dveh pa se ni spremenilo.



Slika 5.33: Primerjava izhodnih rezultatov krmiljenja z učenjem Q in klasičnega prometno odvisnega krmiljenja za vse tri prometne obremenitve

Figure 5.33: Comparison of simulation results between Q learning algorithm and actuated traffic light optimization for all three traffic volumes

Skupne zamude zaradi mirovanja so edino merilo, pri katerem smo v P1 in P2 poslabšali vrednosti. V tretjem primeru so skupne zamude zaradi mirovanja za malenkost nižje, vendar razlika ni statistično značilna. Poslabšanje rezultatov si lahko razložimo, če pogledamo skupaj še število ustavljanj. Tako kot smo že zapisali, se v obeh primerih število ustavljanj zmanjša. Na glavni smeri algoritem namreč podaljša zeleni čas in v ciklusu manjkrat preklopi med fazama. S tem se ustvarijo skupine vozil, ki nato skupaj prevozijo arterijo. Seveda je tako krmiljenje možno, če imamo take prometne razmere oziroma obremenitve, kot so v P1 in P2, tj. bistveno večje obremenitve na glavni smeri. Pri prvih dveh prometnih intenzitetah je razmerje med vstopi vozil z zahoda in vzhoda proti vstopom s severa in juga 74 : 26 oziroma 77 : 23. Pri prometnih volumnih P3 pa je to razmerje 55 : 45. Razmerja zavijalcev pri prometnih obremenitvah P3 so prav tako spremenjena v primerjavi s prvima dvema primeroma. Pri slednjem si ne moremo privoščiti tako velike razlike v trajanju zelenega časa med glavno in prečno smerjo. Prometni volumni in deleži so prikazani v naslednji preglednici 5.10. Potovalni časi so v vseh treh primerih skrajšali v primerjavi s prometno odvisnim krmiljenjem.

Preglednica 5.10: Vhodne prometne obremenitve glede na smer vstopa
Table 5.10: Total vehicle entrance by directions

Prometni volumen	P1	P2	P3
Skupni vhodni promet (I)	3400	3900	4400
Skupni vhodni promet z V in Z (II)	2500	3000	2400
Skupni vhodni promet s S in J (III)	900	900	2000
III/I*100	26,5	23,1	45,5
Skupni promet zavijalcev s S in J (IV)	720	720	1440
IV/I*100	21,2	18,5	32,7

Iz rezultatov lahko razberemo, da je razlika med obema krmiljenjema v nenasičenih razmerah manjša kot pri večjih prometnih intenzitetah. To nakazuje, da prometno odvisno krmiljenje deloma zadovoljivo vzdržuje ustrezen prometni nivo uslug, po prekoračitvi razmerja volumen proti kapaciteti pa drastično pade. Dodaten dokaz so vozila, ki niso mogla vstopiti v mrežo zaradi prevelikih zajezitvenih dolžin. Predlagan nov način krmiljenja je veliko fleksibilnejši od klasičnega prometno odvisnega.

V tem poglavju smo analizirali učinkovitost predlaganega algoritma krmiljenja s pomočjo učenja Q. Vrednotenje je bilo narejeno na cestni arteriji treh križišč. Simulacije krmiljenja so bile narejene s pomočjo programskega orodja VISSIM. Za potrditev uspešnosti nove metode krmiljenja smo analizirali simulacije v treh različnih variantah oziroma prometnih obremenitvah. Prve prometne obremenitve so predstavljale nenasičene prometne razmere, druge nasičene in tretje prav tako

nasičene, vendar z različnim deležem prometa na glavnih in prečnih smereh. Po intenziteti prometa vsi trije primeri predstavljajo karakteristike koničnih ur. Za enake prometne razmere smo simulirali in vrednotili rezultate, dobljene s klasičnim prometno odvisnim krmiljenjem. Slednje smo naredili zaradi primerjave z rezultati učenja Q.

Rezultati kažejo, da je krmiljenje SSN z učenjem Q veliko učinkovitejše in bolj prilagodljivo, kot je to s klasičnim prometno odvisnim krmiljenjem. Za vse prometne obremenitve in za večino primerjanih meril (povprečnih zamud na vozilo, števila prepeljanih vozil, povprečne hitrosti in potovalnega časa) se je algoritem izkazal za boljšega. Na drugi strani rezultati kažejo poslabšanje ob skupnih zamudah zaradi mirovanja ob prometnih obremenitvah P1 in P2. Razloge za slednje smo predhodno razložili in sklepali, da je v primeru zagotavljanja sinhronizacije zelenih časov na arteriji to celo boljše. Kljub vsemu bi bilo smiselno na to temo narediti v prihodnosti dodatne analize in raziskave. Rezultati nam kažejo, da so koristi uporabe novega pristopa krmiljenja velike in vredne nadaljnjih raziskav.

6 RAZPRAVA IN SKLEPI

6.1 Razprava in ugotovitve

Optimalno prilagajanje SSN na cestni arteriji v realnem času je zelo zahtevna in kompleksna naloga. Doktorska disertacija predstavlja sodoben pristop koordinacije SSN na cestni arteriji z uporabo SU. Razvili smo učinkovit algoritem za optimizacijo krmiljenja prometa, ki smo ga modelirali kot VAS. Z uporabljenim pristopom se agenti nenehno učijo in prilagajajo stohastični naravi prometa. Pridobljene rezultate smo primerjali z rezultati klasičnega prometno odvisnega krmiljenja. Dokazali smo, da je algoritem boljši. Zelo učinkovito se je izkazal v različnih prometnih razmerah. Pristop krmiljenja prometa, predstavljen v doktorski disertaciji je nov in ima potencial za nadaljnje še kompleksnejše raziskave z bolj zapletenimi in večjimi prometnimi omrežji. Z manjšimi prilagoditvami in optimizacijo skoraj ni ovir glede velikosti prometne mreže. Med drugim predstavlja osnovo za morebitno implementacijo v realno okolje.

V predhodnih delih drugih avtorjev smo zasledili različne poenostavitve, kot so npr. omejitev obravnavanega števila križišč (Lu in sod., 2008), neupoštevanje zavijalcev (Gregoire in sod., 2007), poenostavljen model prometnega toka (Wiering, 2000) itd. Optimizacija krmiljenja SSN v stohastičnem okolju zahteva hitro odzivnost, zato se srečamo z velikim številom podatkov, ki jih je treba upoštevati. Z naraščanjem števila obravnavanih križišč se kompleksnost problema veča, zato so razumljive določene poenostavitve pri reševanju problema. Kljub temu da je disertacija naš prvi poskus optimiziranja krmilnih programov na arteriji s pomočjo SU, smo se poslužili le malo poenostavitev. Pravzaprav so te zanemarljive v primerjavi s prej omenjenimi. Na vseh krakih križišč smo upoštevali prometne tokove v vseh smereh, uporabili smo spreminjajoče se dolžine semaforskih ciklusov, algoritem smo testirali v nasičenih in nenasičenih prometnih razmerah ter uporabili mikrosimulacijsko orodje, ki natančno ponazarja odvijanje realnega prometnega toka. Poleg simulacijskega orodja VISSIM se je kot zelo učinkovit in prilagodljiv izkazal tudi vmesnik COM.

Predlagan algoritem je sestavljen iz dveh faz. Prva predstavlja učenje agenta/-ov. Iz izvedenih preizkusov v poglavju 5.3 je razvidno, da se je sistem sposoben naučiti v 20 do 30 iteracijah (simulacijskih urah), kar pomeni, da po tej fazi krivulja povprečnih zamud na vozilo oscilira minimalno in blizu optimalnih zamud. Druga faza je izkoriščanje pridobljenega znanja in po potrebi prilagajanje novim razmeram v prometu. Torej, po učnem procesu se je agent sposoben odzvati v vsakem koraku na spremembe prometnega volumna in zagotoviti kar najboljši krmilni program v

danih okoliščinah v realnem času. Agent se lahko odloča za spremembo akcije na 4 sekunde. Ob popolnoma spremenjenih prometnih obremenitvah, npr. v primeru nepredvidenega dogodka ali pri preusmeritvah prometa iz glavne na stranske smeri, agent reagira takoj in ponovno išče optimum glede na nove prometne razmere. Agenti se brez težav prilagajajo novim razmeram kljub spreminjajoči se naravi prometa. V primeru implementacije predlaganega algoritma v realno okolje bi se agent nenehno prilagajal glede na prometne spremembe.

Glede na izbrane parametre stanj križišč lahko zapišemo, da je sistem neodvisen in tudi zahtevnost preračunavanja je linearna v odvisnosti od števila obravnavanih križišč. Čeprav se kompleksnost preračunavanja veča s številom križišč, uporablja sistem zelo majhno količino procesorskega časa za določitve v realnem času. Tudi v stotih križiščih se zaradi osnovne narave izkoriščanja algoritma učenja Q problem ne poveča bistveno.

Kot so v raziskavi Abdulhai in sod. (2003) vrednotili vpliv globalnih in lokalnih nagrad, smo tudi v naši raziskavi izvedli analize rezultatov z obema načinoma nagrajevanja. Iz analize lahko povzamemo, da globalno nagrajevanje ne prinese nujno optimalnih rezultatov na celotni mreži. Rezultati so bili v vseh prometnih razmerah boljši z uporabo lokalne nagrade. Pri globalni nagradi lahko izbiro akcije posameznega agenta druga dva agenta razvrednotita.

Povprečno zamudo na vozilo in število prepeljanih vozil skozi mrežo smo primerjali s klasično prometno odvisnim krmiljenjem. Predlagani stohastični algoritem učenja Q zmanjšuje povprečno zamudo na vozilo in zvišuje število prepeljanih vozil skozi mrežo v primerjavi s klasičnim prometno odvisnim krmiljenjem.

V vseh testih z različnimi prometnimi obremenitvami se je predlagani algoritem izkazal za učinkovitega. Z novo metodo je možno znatno zmanjšati povprečne zamude na vozilo v primerjavi s prometno odvisnim krmiljenjem. Prav tako velja, da smo dobili boljše rezultate v večini ostalih primerjanih meril. Primerjavo smo naredili za nenasičene in nasičene prometne razmere. Primerjani rezultati obenem kažejo, da je učinkovitost predlaganega algoritma Q v primerjavi s klasičnim krmiljenjem večja pri večjih prometnih obremenitvah, kar nakazuje, da je možno v prometnih konicah doseči zmanjšanje zamud in povečati število prepeljanih vozil, kar je glede na problematiko prometnih zastojev ključnega pomena. Ob nenasičenih prometnih razmerah so razlike povprečnih zamud na vozilo v primerjavi s prometno odvisnim krmiljenjem za 25 % manjše z uporabo algoritma učenja Q. V nasičenih prometnih razmerah so 22 % manjše, prav tako v korist učenja Q. Predlagani

način krmiljenja s stohastičnim učenjem Q je veliko bolj fleksibilen od klasičnega prometno odvisnega krmiljenja.

6.2 Prispevek disertacije

Izvirni prispevek k razvoju znanstvenega področja je razvit algoritem prilagodljivega krmiljenja SSN na cestni arteriji, s katerim smo dosegli boljši nivo uslug in večjo učinkovitost prometne mreže. Po našem vedenju je to prvi poskus uporabe stohastičnega večagentnega sistema za krmiljenje SSN na cestni arteriji. Značilnost sistema je tudi poenostavljen opis stanj posameznega agenta in omejena komunikacija med agenti, kar omogoča implementacijo v realnem okolju. Z omenjenim razvojem algoritma smo dokazali, da je to možno, in s tem podali osnove za prihodnji razvoj SSN, ki bi znatno zmanjšal zamude, povečal število prepeljanih vozil, zvišal povprečno hitrost, zmanjšal čas potovanj, število ustavljanj, porabo goriva, emisije in ne nazadnje povečal prometno varnost. Cilj je bil razviti sistem, ki bi se sam nenehno učil in prilagajal spreminjajočim se prometnim razmeram na terenu.

Nova predlagana metoda temelji na učenju Q kot posebni veji SU. Z uporabo stohastičnega algoritma učenja Q smo razvili in preverili učinkovitost krmiljenja SSN v različnih prometnih razmerah.

Razvili smo algoritem, ki je sposoben krmiliti SSN na cestni arteriji kot VAS s pomočjo metode SU. Brez uporabe uteži za favoriziranje glavne smeri se agenti nekoliko počasneje učijo, vendar so končni rezultati boljši v primerjavi s klasičnim krmiljenjem. Pregledali smo različne strategije za koordinacijo agentov ter analizirali prednosti in slabosti. Na koncu smo za krmiljenje in koordinacijo SSN določili preprosto, vendar robustno strategijo, ki se je izkazala za zelo učinkovito.

S predstavljenimi rezultati smo prikazali konvergenco algoritma v različnih prometnih razmerah brez predhodnega znanja in z njim. S predlaganim algoritmom je omogočeno, da se agenti hitro in učinkovito naučijo pravilne strategije, ki se odraža v manjših zamudah.

Agenti se naučijo učinkovite strategije krmiljenja cestne arterije že po 20 do 30 simulacijskih urah. Sposobni so izkoristiti tudi predznanje iz situacij, ki so lahko različne od trenutnih. Našteto nakazuje, da je razvit stohastični algoritem učenja Q zmožen krmiljenja v realnem času.

Analize učinkovitosti krmiljenja s predlaganim algoritmom smo izvedli z mikrosimulacijskim orodjem, s katerim je mogoče z veliko natančnostjo simulirati realne prometne situacije. Simulacije smo izvedli z uveljavljenim mikrosimulacijskim programskim orodjem VISSIM. V večini

predhodnih raziskav za vrednotenje učinkovitosti algoritmov niso uporabili splošno znanih in uporabljenih mikroskopskih simulacijskih orodij, ki lahko natančno ponazarjajo prometno dogajanje. Samostojno razvita simulacijska orodja po eni strani ponujajo uporabnikom v simulacijskem procesu lažje, poenostavljeno upravljanje, vendar se simuliran promet ne vede tako realno kot v preverjenih in uveljavljenih programskih orodjih.

Na enaki prometni mreži in v enakih prometnih razmerah smo naredili obsežno primerjavo krmiljenja s stohastičnim algoritmom učenjem Q in klasičnim prometno odvisnim krmiljenjem. Dokazali smo, da je algoritem v vseh primerih boljši od klasične tehnike krmiljenja. V večini pregledanih del ni bilo narejenih primerjav s prometno odvisnim načinom krmiljenja, ki je neprimerno učinkovitejši kot časovno odvisno krmiljenje. Zaradi tega imajo predstavljeni rezultati še toliko večjo težo.

Rezultati raziskave potrjujejo hipotezo, saj z uporabo algoritma spodbujevanega učenja z upoštevanjem stohastičnega večagentnega sistema, s predlaganim opisom okolja, definiranjem stanj (dolžina kolone v glavni smeri, dolžina kolone v prečni smeri, trenutna faza semaforja in trajanje zelenega signala), možnih akcij (podaljšanje trenutne faze ali njena zamenjava) ter načina nagrajevanja problem krmiljenja SSN rešujemo uspešneje, kot klasično prometno odvisno krmiljenje.

6.3 Ideje za nadaljnje delo

Kljub temu da smo s preizkusi dosegli zavidljive rezultate, smo med raziskavo dobili ideje za morebitno izboljšanje in nadgradnjo sistema.

V prihodnje bi bilo treba preizkusiti algoritem na arteriji z več križišči oziroma na mreži križišč z različnimi prometnimi obremenitvami na posameznih krakih, spreminjati celotno strukturo prometa med učenjem in optimizirati prilagodljivost sistema ob spremembah prometnih obremenitev.

Tako kot v našem primeru je večina študij upoštevala krmiljenje prometa z dvema semaforskima fazama. Le redke študije, ki so zasnovane na nekonvencionalnih tehnikah krmiljenja, so upoštevale več faz. Če je zavijalcev veliko, je navada, da je krmilni program sestavljen iz več kot dveh faz. Zaradi tega bi bilo zanimivo vključiti v algoritem dodatne faze in analizirati prilagajanje agentov. Poleg dodatnih faz bi lahko v prihodnje SU uporabili za optimizacijo zaporedja posameznih faz.

V četrtem poglavju smo opisali, da smo definirali tri možne dolžine kolon, in sicer kratko, srednje dolgo in dolgo kolono. To pomeni, da v praksi potrebujemo tri detektorje na vsakem kraku oziroma prometnem kraku. S ciljem čim nižjih stroškov za implementacijo algoritma v realnem okolju je v prihodnje smiselno raziskati, kako dodatno optimizirati oziroma opisati stanja. S tem bi lahko zmanjšali število detektorjev.

Na učenje in prilagajanje algoritma vpliva veliko število faktorjev. V poglavju z rezultati smo zapisali, da smo poleg analiziranih parametrov preizkušali tudi različne tipe učenja. Zaradi znatnega porabljenega časa pri programiranju in razhroščevanju algoritma bi bilo smiselno usmeriti dodaten napor v analizo učinka različnih tipov učenja, nagrajevanja in preostalih faktorjev.

7 POVZETEK

Uporabniki cestne infrastrukture se posebno v mestnih središčih zaradi sprememb potovalnih navad, večjih ter spreminjajočih se prometnih obremenitev srečujemo z zastoji, kar se odraža v višjih stroških potovanj, nižji prometni varnosti in večji onesnaženosti okolja. Križišča so zaradi učinkovitejšega in varnejšega vodenja prometa opremljena s svetlobnosignalnimi napravami. Krmiljenje svetlobnosignalnih naprav lahko upravljamo s časovno ali prometno odvisnimi sistemi. Časovno odvisne naprave uporabljajo prednastavljene krmilne programe, ki so narejeni na podlagi predhodnega zbiranja prometnih podatkov. Prometno odvisne naprave so v primerjavi s časovno odvisnimi bolj prilagodljive, saj na podlagi detektiranih vozil spreminjajo krmilno strategijo. Vendar se ob večjih prometnih spremembah tudi pri slednjih pojavijo omejitve zaradi prednastavljenih robnih razmer in omejenih zmogljivosti. Težave s klasičnimi metodami krmiljenja so še posebej opazne pri nepredvidenih prometnih dogodkih in v nasičenih prometnih razmerah. Občutna sprememba prometnega povpraševanja zahteva popolno prilagoditev krmilne strategije.

Objavljeni rezultati raziskav drugih avtorjev nakazujejo, da je kompleksne in zahtevne procese krmiljenja mogoče uspešno reševati z uporabo umetne inteligence, natančneje spodbujevanega učenja. Z relativno novim pristopom na področju krmiljenja svetlobnosignalnih naprav je mogoče doseči učenje in posodabljanje krmilne strategije. Pristopi se razlikujejo predvsem v zmogljivosti sistema krmiljenja glede števila obravnavanih križišč. Večje število analiz je narejenih za samostojna križišča, v zadnjem obdobju je opaziti porast poskusov krmiljenja večjih prometnih mrež. Skupna točka pristopov je zmožnost samostojnega učenja sistema krmiljenja na podlagi interakcije med agenti in okoljem. Vendar moramo pri tem dodati, da smo v predhodnih delih zasledili tudi različne poenostavitve. Med katerimi najbolj izstopajo poenostavljene geometrije mrež, intenzitete prometa in uporabljeni simulacijski modeli. Za vse pristope prav tako ne velja, da so primerni za krmiljenje prometa v realnem času. Želja vsakega upravljalca prometa je takojšnje prilagajanje s čim manjšimi izgubami časa.

V doktorski disertaciji smo predstavili teoretična izhodišča in razvoj algoritma krmiljenja prometa na osnovi spodbujevanega učenja. Nazorno smo opisali vse elemente spodbujevanega učenja, ki so nujni za delovanje sistema krmiljenja svetlobnosignalnih naprav. S predlaganim pristopom smo dosegli, da se sistem samostojno uči in prilagaja. Optimizacijo krmiljenja cestne arterije smo zasnovali kot večagentni sistem. S tako zasnovo in optimalno definiranimi stanji okolja algoritem ni omejen zgolj na samostojna križišča, temveč je zmožen krmiliti večje število križišč.

Eksperimente in vrednotenje rezultatov algoritma smo izvedli s pomočjo uveljavljenega mikrosimulacijskega orodja. Mikroskopska računalniška simulacija prometa nam omogoča ponazoritev realnih prometnih situacij z veliko natančnostjo. Analize učinkovitosti predlaganega algoritma smo preverjali za različne prometne intenzitete, ki so predstavljale konične ure. Preizkušali smo jakosti prometnih tokov, za katere veljajo značilna razmerja na cestni arteriji, kot tudi take, ki veljajo za cestne mreže ali so posledica nepredvidenih dogodkov. Za vse naštete prometne razmere smo izvedli analizo parametrov in konvergenco algoritma. Izbrani parametri so bili uporabljeni za nadaljnjo primerjavo in oceno uspešnosti novega pristopa krmiljenja svetlobnosignalnih naprav.

V doktorski disertaciji smo razvili in predstavili stohastični algoritem učenja Q, ki je sposoben učinkovito krmiliti promet na cestni arteriji. Algoritem je sposoben na podlagi prejetih nagrad prilagoditi strategijo krmiljenja tako, da zmanjša zamude in poveča število prepeljanih vozil skozi križišča. Večagentni sistem se hitro in učinkovito prilagaja nenehnim spremembam v prometnem toku. Dokazali smo, da je z novim predlaganim pristopom mogoče doseči boljše nivoje uslug v primerjavi s klasičnim prometno odvisnim krmiljenjem. Primerjani rezultati so bili boljši z uporabo učenja Q tako v nenasičenih kot tudi nasičenih prometnih razmerah. Predstavljen pristop in rezultati nakazujejo potencial za nadaljnje kompleksnejše raziskave optimizacije krmiljenja prometa. Uspešno izvedene simulacije z mikrosimulacijskim orodjem, hitra konvergenca k optimalnim rezultatom in prilagajanje v najrazličnejših prometnih razmerah nakazujejo, da je algoritem sposoben krmiliti cestno arterijo v realnem času. Disertacija je obenem tudi osnova za morebiten poskus implementacije algoritma v realni svet.

8 SUMMARY

Users of road infrastructure in urban areas often face traffic delays, higher travel costs, lower safety, and increased pollution due to changes in travel behaviour and consequently increased traffic volumes. For higher efficiency and safety the traffic flows in intersections are usually controlled with traffic lights. For traffic light coordination most commonly two types of controllers are used, namely pre-timed signal controllers and actuated traffic light controllers. Signal plans for pre-timed traffic light controllers are defined on the basis of historical traffic volume data. Greater flexibility to actual traffic flows is achieved with actuated traffic light controllers, which continuously detect traffic flow and adjust to the current traffic volume. However, this adjustment is limited due to predefined boundary conditions and limited capacity of the system. The problem is even more evident when unexpected traffic events occur and in oversaturated traffic flow. Significant changes in traffic flow require adaptation of control strategy.

Literature review identifies that complex problem of traffic lights optimization can be efficiently optimized with the use of artificial intelligence, namely reinforcement learning. With this relatively novel approach the traffic control strategy can be learned and adapted for different traffic volumes. Approaches described in recent literature differ mainly in number of intersections the system can process. Majority of research is done for a single intersection, however in last decade number of research on road network with several intersections increased. These modern approaches have the ability of self-learning control system based on interaction between the agent and the environment, but researchers propose several simplifications. Among them the simplified road network, low intensity of traffic volume, and macroscopic simulation models are the most common. Not all proposed approaches can be used in real-time, what should be the core goal of the approaches.

In the thesis, theoretical framework of reinforcement learning and development of traffic control optimization algorithm are presented. All elements of reinforcement learning used in algorithm for solving problem of traffic controller optimization are systematically described. Proposed approach enables that the system is self-learning, and it adapts to the changes in traffic. Optimization of traffic controller is designed as a multi-agent system, and as such is not limited to a single intersection, but can be implemented on an arbitrary number of intersections.

Experiments and evaluation of the algorithm was performed using microscopic simulation tool which illustrates the real-life traffic nature with great accuracy. Efficiency of proposed algorithm was tested

for different traffic volumes presenting different peak hours. Tests were performed for typical rates of traffic flow, as well as for those that apply to unpredicted events. For all traffic conditions the parameter setting and the convergence of the results were investigated. Results obtained with the proposed algorithm with selected parameters were compared with results obtained with actuated signal controller.

In the thesis we have developed and introduced stochastic Q learning algorithm for optimization of traffic controller on a road artery. Algorithm adapts the strategy of traffic control based on received reinforce signal (reward) in a way that the average delay per vehicle are minimized and the number of vehicles that left road network are maximized. Proposed multi-agent algorithm quickly and efficiently adapts to changes in traffic flow. Results prove that the level of service obtained with proposed approach is better than one obtained with actuated signal controller, in both saturated and oversaturated traffic flows. Results of the proposed algorithm indicate the potential for further researches on optimization of traffic controllers with use of artificial intelligence. Successful implementation of microscopic simulation tool, fast convergence to the optimal results, and successful adaption to different traffic conditions (traffic volumes) indicates that the algorithm could be used in a real world and in real-time.

VIRI

Abdulhai, B., Kattan, L. 2003. Reinforcement learning: Introduction to theory and potential for transport applications. Canadian Journal of Civil Engineering, 30(6); 981–991. doi:10.1139/l03-014

Abdulhai, B., Pringle, R., Karakoulas, G. J. 2003. Reinforcement Learning for True Adaptive Traffic Signal Control. Transportation, (June); 278–285.

Arel, I., Liu, C., Urbanik, T., Kohls, A. G. 2010. Reinforcement learning-based multi-agent system for network traffic signal control. IET Intelligent Transport Systems, 4(2); 128–135. doi:10.1049/iet-its.2009.0070

Bazzan, A. L. C. 2005. A Distributed Approach for Coordination of Traffic Signal Agents. Autonomous Agents and Multi-Agent Systems, 10(1); 131–164. doi:10.1007/s10458-004-6975-9

Bingham, E. 2001. Reinforcement learning in neurofuzzy traffic signal control. European Journal Of Operational Research, 131; 232–241.

Birst, S., Baker, J., Shouman, K. 2007. Comparison of Traffic Simulation Models with HCM 2000 Methodology Using Various Traffic Levels Under Pretimed Signal Control. Presented at 86th Annual Meeting of the Transportation Research Board. Washington, D.C; 18.

Bullock, D., Abbas, M. 2001. A Real-Time Offset Transitioning Algorithm for Coordinating Traffic Signals. Purdue University; 421. doi:10.5703/1288284313130

Bullock, D., Urbanik, T. 2000. Traffic Signal Systems: Addressing Diverse Technologies and Complex User Needs; 9.

Cahill, V., Salkham, A. 2008. A collaborative reinforcement learning approach to urban traffic control. IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology; 560–566.
http://www.tara.tcd.ie/handle/2262/32669 (Pridobljeno 26. 9. 2013)

Cai, C., Wong, C. K., Heydecker, B. G. 2009. Adaptive traffic signal control using approximate dynamic programming. Transportation Research Part C, 17; 456–474. doi:10.1016/j.trc.2009.04.005

Cai, C. 2007. An approximate dynamic programming strategy for responsive traffic signal control. Proceedings of 2007 IEEE international Symposium on Approximate Dynamic Programming and Reinforcement Learning; 303–310.

Crites, R. H., Barto, A. G. 1998. Elevator Group Control Using Multiple Reinforcement Learning Agents, 262; 235–262.

Diakaki, C., Papageorgiou, M., Aboudolas, K. 2002. A multivariable regulator approach to traffic-responsive network-wide signal control. Control Engineering Practice, 10; 183–195. doi:10.1016/S0967-0661(01)00121-6

Gartner, N., Pooran, F., Andrews, C. 2001. Implementation of the OPAC adaptive control strategy in a traffic signal network. Proceedings of the 2001 IEEE Intelligent Transportation Systems Conference. Oakland, California. IEEE; 195-200.

Gordon, R. L., Tighe, W., Siemens ITS. 2005. Traffic control systems handbook. Federal Highway Administration Office of Transportation Management; 368.

Gregoire, P.-L., Desjardins, C., Laumonier, J., Chaib-draa, B. 2007. Urban Traffic Control Based on Learning Agents. 2007 IEEE Intelligent Transportation Systems Conference. Bellevue, WA, USA. IEEE; 916–921. doi:10.1109/ITSC.2007.4357719

HCM. (2010). Highway Capacity Manual. Transportation Research Board. Washingotn, DC.

Henry, J., Farges, J. 1990. Prodyn. Control, Computers, Communications in Transport; 253–255.

Heydecker, B., Cai, C., Wong, C. 2007. Adaptive dynamic control for road traffic signals. Proceedings of 2007 IEEE International Conference on Networking, Sensing and Control, London, United Kingdom, (April); 15–17. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4238988 (Pridobljeno 20. 3. 2013)

Houli, D., Zhiheng, L., Yi, Z. 2010. Multiobjective Reinforcement Learning for Traffic Signal Control Using Vehicular Ad Hoc Network. EURASIP Journal on Advances in Signal Processing, (1); 7. doi:10.1155/2010/724035

Jin, X., Zhang, Y., Wang, F., Li, L., Yao, D., Su, Y., Wei, Z. 2009. Departure headways at signalized intersections: A log-normal distribution model approach. Transportation Research Part C: Emerging Technologies, 17(3); 318–327. doi:10.1016/j.trc.2009.01.003

Khamis, M. a., Gomaa, W. 2014. Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework. Engineering Applications of Artificial Intelligence, 29; 134–151. doi:10.1016/j.engappai.2014.01.007

Kohl, N., Stone, P. 2004. Policy gradient reinforcement learning for fast quadrupedal locomotion. IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 (Vol.3). IEEE; 2619–2624. doi:10.1109/ROBOT.2004.1307456

Li, H., Prevedouros, P. 2002. Detailed Observations of Saturation Headways and Start-Up Lost Times. Transportation Research Record: Journal of the Transportation Research Board, 1802(12); 44–53. doi:10.3141/1802-06

Liao, L. C. 1998. A Review of the Optimized Policies for Adaptive Control Strategy (OPAC). California Path Program Institute Of Transportation Studies University Of California, Berkeley; 11.

Lowrie, P. R. 1999. Scats, sydney co-ordinated adaptive traffic system : a traffic responsive method of controlling urban traffic. Roads and Traffic Authority NSW, Darlinghurst, NSW Australia; 28.

Lu, S., Liu, X., Dai, S. 2008. Q-Learning for Adaptive Traffic Signal Control Based on Delay Minimization Strategy. 2008 IEEE International Conference on Networking, Sensing and Control; 687–691. doi:10.1109/ICNSC.2008.4525304

Mauro, V., Di Taranto, C. D. 1989. Utopia. In Proceedings of the IFAC-IFIP-IFORS Conference on Control, Computers, Communication in Transportation, Paris; 245–252.

Mikami, S., Kakazu, Y. 1994. Genetic reinforcement learning for cooperative traffic signal control. Proceedings of the First IEEE Conference, IEEE World Congress on Computational Intelligence; 223–228.

Mirchandani, P., Head, L. 2001. A real-time traffc signal control system: architecture, algorithms, and analysis, 9; 415–432.

Mitchell, T. 1997. Machine Learning. Annual Review of Computer Science (Vol. 4). McGraw-Hill; p. 432. doi:10.1146/annurev.cs.04.060190.002221

Moody, J., Saffell, M. 2001. Learning to trade via direct reinforcement. IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council, 12(4); 875–89. doi:10.1109/72.935097

Morgan, J. T., Little, J. D. C. C. 1964. Synchronizing Traffic Signals for Maximal Bandwidth. Operations Research; 896 –912.

Nakamiti, G., Freitas, R. 2002. Adaptive, real-time traffic control management. International journal of automotive technology; 1–6.

Oliveira, D. de, Bazzan, A. 2006. Traffic lights control with adaptive group formation based on swarm intelligence. Proceedings of the 5th International Workshop on Ant Colony Optimization and Swarm Intelligence, ANTS; 12. http://link.springer.com/chapter/10.1007/11839088_61 (Pridobljeno 18. 9. 2013)

Oliveira, D. de, Bazzan, A., Lesser, V. 2005. Using cooperative mediation to coordinate traffic lights: a case study. Proceedings of the 4th International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS); 463–470.

Oliveira, D. de, Ferreira, P. R. Jr., Bazzan, A. L. C., Klügl, F. 2004. A swarm-based approach for selection of signal plans in urban scenarios. Ant Colony Optimization and Swarm Intelligence—ANTS 2004; 416–417. doi:10.1007/978-3-540-28646-2_43

Oliveira, L. B. de, Camponogara, E. 2010. Multi-agent model predictive control of signaling split in urban traffic networks. Transportation Research Part C: Emerging Technologies, 18(1); 120–139. doi:10.1016/j.trc.2009.04.022

Pappis, C., Mamdani, E. 1977. A fuzzy logic controller for a trafc junction. IEEE Trans. Systems, Man and Cybernetics; 10.

Pendrith, M. D. (2000). Distributed reinforcement learning for a traffic engineering application. Proceedings of the fourth international conference on Autonomous agents - AGENTS; 404–411. doi:10.1145/336595.337554

Perez, A. 1998. Introduction to reinforcement learning; 6. http://ape.iict.ch/teaching/AIGS/AIGS_Labo/Labo5-RL/sarsa.html (Pridobljeno 3. 10. 2013)

Prabuchandran, K. J., N, H. K. A., Bhatnagar, S., Member, S. 2014. Multi-agent Reinforcement Learning for Traffic Signal Control. Intelligent Transportation Systems (ITSC), IEEE 17th International Conference; 2529–2534.

Prashanth, L. A., Bhatnagar, S. 2011. Approximation for Traffic Signal Control. Transportation, 12(2); 412–421.

Robertson, D. 1969. TRANSYT: A TRAFFIC NETWORK STUDY TOOL - Transport Research International Documentation - TRID. RRL report, 253; 37.

Robertson, D.I., Bertherton, R. D. 1974. Optimum control of an intersection for any known sequence of vehicular arrivals. Proceedings of the 2nd IFAC-IFIP-IFORS Symposium on Traffic Control and Transportation system, Monte Carlo; 8.

Russell, S. J., Norvig, P. 2003. Artificial Intelligence A Modern Approach. (S. J. Russell P. Norvig, Eds.) (2nd Edition). Prentice Hall; 1132.

SCOOT. (n.d.). Split, Cycle and Offset Optimization Technique. http://www.scoot-utc.com/SCOOTMMX.php?menu=Versions (Pridobljeno 12. 9. 2011)

Spall, J., Chin, D. 1997. Traffic-responsive signal timing for system-wide traffic control. Transportation Research Part C; 153–163.

Srinivasan, D., Choy, M., Cheu, R. 2006. Neural networks for real-time traffic signal control. IEEE Trans. Intelligent Transport Systems, 7(3); 261–272.

Stone, P., Veloso, M. 2000. Multiagent systems: A survey from a machine learning perspective. Autonomous Robotics, 8; 57.

Sutton, R., Barto, A. 1998. Reinforcement Learning: An Introduction. Trends in Cognitive Sciences (Vol. 3). A Bradford Book; 360. doi:10.1016/S1364-6613(99)01331-5

Tesauro, G. 1995. Temporal difference learning and TD-Gammon. Communications of the ACM, 38(3); 58–68. doi:10.1145/203330.203343

Thorpe, T. L., Anderson, C. W. 1996. Traffic Light Control Using SARSA with Three State Representations. IBM Corporation; 7.

VAP. 2011. VAP User Manual, Version 2.16. PTV Planung Transport Verkehr AG, Karlsruhe, Germany; 42.

VISSIM. 2011. VISSIM User Manual, Version 5.40. PTV Planung Transport Verkehr AG, Karlsruhe, Germany; 730.

VISSIMCOM. 2011. VISSIM COM Interface Manual, Version 5.40. PTV Planung Transport Verkehr AG, Karlsruhe, Germany; 287.

Wiering, M. 2000. Multi-agent reinforcement learning for traffic light control. Proceedings of the 17th International Conference on Machine Learning; 8.

Wiering, M., Veenen, J. van, Vreeken, J., Koopman, A. 2004. Intelligent traffic light control. Institute of Information and Computing Sciences, Utrecht University, Technical Report; 31.

Wunderlich, R., Elhanany, I., Urbanik, T. 2008. A Novel Signal-Scheduling Algorithm With Quality-of-Service Provisioning for an Isolated Intersection. IEEE Transactions on Intelligent Transportation Systems, 9(3); 536–547. doi:10.1109/TITS.2008.928266