SVEUČILIŠTE U SPLITU
FAKULTET ELEKTROTEHNIKE, STROJARSTVA I BRODOGRADNJE

Kvalifikacijski doktorski ispit:

# Korištenje heterogenih višeagentskih sustava sa mogućnošću učenja za rješavanje problema pretrage prostora stanja

Split, 1. rujna, 2014, Goran Zaharija

# Sadržaj

# 1. Uvod

Korištenje računala u svrhu rješavanja različitih problema je u današnje vrijeme svakodnevna praksa i koristi se u mnogim aspektima života. Da bi se računalo moglo koristiti za rješavanje nekog određenog problema potrebno je osmisliti i implementirati sustav koji ima sposobnost pronalaska željenog rješenja. Umjetna inteligencija je jedno od područja koje se upravo bavi dizajniranjem i implementacijom takvih inteligentnih sustava. Cilj ovoga rada je opisati osnovne pojmove i definicije vezane uz navedenu problematiku te dati kratki pregled provedenih istraživanja koja se bave tim područjem. To uključuje višeagentske sustave, algoritme pretrage te metode učenja.

U drugom poglavlju rada opisan je postupak odabira literature pri izradi ovog rada. Treće poglavlje se odnosi na područje agenata i višeagentskog sustava. Navodi se definicija agenta i njegove karakteristike, kao i različite arhitekture. Četvrto poglavlje opisuje algoritme pretrage i njihovu upotrebu u rješavanju problema. Peto poglavlje daje pregled metoda učenja koje se koriste za podučavanje agenata. U šestom poglavlju je iznesen zaključak rada.

# 2. Pregled literature

U ovom dijelu rada je opisan postupak koji se koristio prilikom pretrage relevantne literature. Svrha ovog postupka je identificirati i pristupiti literaturi koja je u skladu sa pravilima uključivanja i isključivanja koja su navedena kasnije u radu. Za pretragu se koristila kombinacija automatske pretrage elektroničkih baza i ručne pretrage određenih časopisa te ostalih izvora.

## 2.1 Područje interesa

Kao što je navedeno u uvodu, područje interesa ovog rada obuhvaća višeagentske sustave, algoritme pretrage i metode učenja. Iako su ova područja međusobno povezana, radi se o prilično širokim područjima te je zbog toga pretraživanje relevantne literature napravljeno odvojeno za svako područje, sa posebnim naglaskom na radove koji uključuju preostala područja.

## 2.2 Korišteni resursi prilikom pretraživanja

Sljedeći resursi su se koristili prilikom pretraživanje relevantne literature povezane sa zadanim područjima interesa:

### 2.2.1 Elektroničke baze:

Da bi se osigurala adekvatna pretraga relevantne literature, korištene su navedene elektroničke baze:

- ACM Digital Library
- CiteSeerX
- IEEE Xplore
- Science Direct
- Web of Science
- Google Scholar

Korišteno je automatsko pretraživanje za odabir relevantnih radova, strategija pretraživanja i kriterij odabira radova su navedeni u narednim poglavljima.

### 2.2.2 Časopisi:

Uz automatsku pretragu navedenih elektroničkih baza, također je izvršena i ručna pretraga nekoliko odabranih časopisa. Kriterij za odabir časopisa je bio da opseg časopisa uključuje područje umjetne inteligencije te da su teme radova usko vezane uz područja interesa ovog rada. Zbog većeg broja takvih časopisa, odabrani su samo oni koji su rangirani pri vrhu koristeći SJR metodu usporedbe časopisa [1]. Primjenom navedenih kriterija odabrani su sljedeći časopisi za pretraživanje:

- International Journal of Robotics Research (IJRR – SAGE Publications)
- Journal of Machine Learning Research (JMLR – Microtome Publishing)
- Artificial Intelligence (Elsevier)
- Autonomous Robots (Springer)
- Autonomous Agents and Multi-Agent Systems (Springer)

### 2.2.3 Ostali izvori:

Da bi se osigurala temeljitost pregleda literature uključeni su i dodatni izvori. Nakon što je određeni rad odabran navedenom strategijom pretrage,s i u slučaju da zadovoljava kriterij uključivanja, također je ručno pregledan popis literature tog odabranog rada kako bi se otkrili radovi koji su također povezani sa područjem interesa a nisu otkriveni tokom inicijalnog pretraživanja. Pretraživanje Interneta putem nekih od web tražilica se nije se koristilo zbog toga što takve izvore često nije moguće citirati. Pretraga Interneta je bila izvršena jedino u svrhu pronalaženja točno određenog rada čija cijela verzija možda nije dostupna putem unutar baze koja je služila za pretraživanje.

## 2.3 Strategija pretrage

Zbog širine odabranog područja, pretraga literature je odvojena u tri posebna dijela, te su za svaki dio definirane ključne riječi koje su se koristile prilikom pretrage navedenih izvora. Kako bi se osiguralo postizanje što boljih rezultata, testirano je nekoliko fraza koje su se koristile za upit prilikom pretrage. Prilikom pretrage digitalnih baza korištene su napredne metode pretrage sa logičkim operatorima OR, AND i NOT kako bi se dodatno proširili ili filtrirali rezultati, ovisno o potrebama.

### 2.3.1 Validacija strategije pretraživanja

Tokom prijašnjih istraživanja koja su provedena u ovim područjima identificirano je nekoliko radova koji su relevantni za navedena područja [2-4]. Da bi se provjerila ispravnost prethodno navedene strategije pretraživanja korišteno je testno pretraživanje u kojem su korištene odabrane ključne riječi za svako od tri područja interesa. Koristile su se elektroničke tražilice ACM i IEEExplore. Rezultat pretraživanja je da su navedeni radovi pronađeni u sklopu dobivenih rezultata i time je potvrđena ispravnost opisane strategije pretraživanja.

## 2.4 Kriteriji odabira literature

Da bi se osiguralo da se kod pregleda literature razmatraju samo relevantni radovi s obzirom na područje zanimanja, na rezultate pretrage primijenili su se uključujući i isključujući kriteriji koji su nam poslužili kao filter dobivenih rezultata.

### 2.4.1 Kriteriji uključivanja

Osnovni kriterij za uključivanje rada u pregled literature je da je tema rada usko povezana sa jednim od tri područja interesa koji su prethodno navedeni tj. kako se pretraga radi za svako područje posebno, rad se mora odnositi na točno zadano područje interesa (višeagentski sustavi, algoritmi pretrage te učenje i prikaz znanja). Također, radovi koji sadrže empirijsko istraživanje ili imaju eksperimentalni dio sa prikazom pripadajućih rezultata su bili uključeni u istraživanje. U slučaju da više radova obrađuje istu temi ili imaju identično istraživanje, u obzir se uzeo rad sa novijim datumom objave. Osim u navedenom slučaju, godina objave rada se nije koristila kao kriterij odabira. Literatura koja spada pod kategoriju „sive literature“ kao što su patenti ili tehnički izvještaji će biti uključena u istraživanje samo u slučaju da je relevantna u odnosu na odabrano područje istraživanja.

### 2.4.2 Kriterij isključivanja

Radovi nisu bili uključeni u istraživanje ako im glavni fokus nije na jednoj od tri područja interesa. Nedostatak empirijskog istraživanja ili prikaza rezultata je također jedan od kriterija isključivanja. U razmatranje se nisu uzeli radovi koji samo opisuju neki novi pristup ali ne prikazuju nikakve konkretne rezultate ili testiranja vezana uz taj novi pristup. Nepotpuni radovi tj. radovi koji nisu dostupni u punoj verziji nego u skraćenom obliku (samo sažetak ili dio teksta) su također bili isključeni iz istraživanja. Radovi koji nisu napisani na engleskom ili hrvatskom jeziku nisu uzeti u razmatranje.

# 3. Višeagentski sustav

Višeagentski sustavi su relativno novo područje računalne znanosti. Početak izučavanja datira oko 1980. godine, dok se značajnija zapaženost javlja sredinom 1990-tih i sve od tada međunarodni interes u ovom području u velikom je porastu. Ovaj brzi rast možemo, barem djelomično, pripisati vjerovanju da su upravo agenti odgovarajuća softverska paradigma koja će iskoristiti mogućnosti masivnih otvorenih raspodijeljenih sustava kao što je Internet. Iako će svakako igrati glavnu ulogu u iskorištavanju potencijala koje pruža Internet, višeagentski sustavi mogu pružiti puno više od toga, štoviše, možemo ih smatrati prirodnom metaforom za razumijevanje i stvaranje velikog broja, grubo rečeno, umjetnih društvenih sustava.

Ideja višeagentskih sustava nije vezana za samo jednu domenu, već pronalazi primjenu u različitim tipovima domena. Upravo zbog toga možemo reći kako je područje višeagentskih sustava izrazito interdisciplinarno. Naime, ono povlači inspiracije iz raznih područja kao što je ekonomija, filozofija, logika, ekologija i društvene znanosti pa nas ne treba iznenaditi postojanje različitih gledišta i definicija pojma višeagentskih sustava [3].

Usprkos različitim viđenjima teorije višeagentskih sustava, postoji općeprihvaćena definicija višeagentskog sustava kao sustava sastavljenog od većeg broja međudjelujućih inteligentnih agenata unutar nekog okruženja. Kao takav, sustav se može koristiti za rješavanje problema koji su preteški ili nemogući za pojedinog agenta ili monolitni sustav. Višeagentski sustavi imaju sposobnost samoorganizacije i samoupravljanja kao i neke druge kontrolne obrasce i povezana složena ponašanja čak i onda kada su individualne strategije svih agenata vrlo jednostavne. Kao jedna od glavnih karakteristika jest pronalaženje najboljeg rješenja određenog problema i to bez ikakvih vanjskih intervencija. Tu postoji velika sličnost sa fizikalnim fenomenom minimalne energije gdje fizikalni objekti nastoje postići najnižu moguću energiju unutar ograničenog fizikalnog svijeta.

Glavna značajka koja se postiže prilikom razvijanja višeagentskih sustava, ako dobro funkcioniraju, jest fleksibilnost koja omogućava nadopunu, izmjenu i rekonstrukciju sustava bez potrebe za detaljnim prerađivanjem same aplikacije. Također, ovi sustavi pokazuju svojstvo brzog samooporavljanja [5].

Tehnologija višeagentskih sustava počinje se probijati van sveučilišta i istraživačkih laboratorija te se primjenjuje u rješavanju stvarnih praktičnih problema u području industrije i komercijalnih aplikacija. Veliki broj takvih aplikacija već postoji i mogu se pronaći u

područjima kao što su kontrola procesa [6, 7], proizvodnja [8], kontrola zračnog prometa [9, 10], telekomunikacijski sustavi, uredska automatizacija, filtriranje i sakupljanje informacija [11], elektroničko trgovanje [12, 13], upravljanje poslovnim procesima [14, 15], zdravstvene aplikacije [16-18], zabavne aplikacije [19, 20].

Uočene su dvije glavne prepreke koje onemogućuju još širu primjenu višeagentskih sustava u svakodnevnom životu. To su:

- nedostatak sustavne metodologije koja bi dizajnerima omogućila jasno određivanje i izgradnju njihovih aplikacija kao višeagentskih sustava
- nedostatak široko dostupnih kvalitetnih alata za izgradnju višeagentskih sustava

Prva zapreka se odnosi na to da je većina postojećih aplikacija dizajnirana na sasvim *ad hoc* način – ili posuđujući metodologiju (u većini slučajeva objektno-orijentiranu) pokušavajući je prilagoditi i staviti u višeagentski kontekst, ili pak radeći bez ikakve metodologije gdje se sustav dizajnira na osnovu intuicije i vlastitog iskustava. Ovakav pristup dizajniranju višeagentskih sustava nije prikladan. Potrebno je predstaviti sustavan način analiziranja problema, razrade izvedbe u obliku višeagentskog sustava te određivanje načina izrade pojedinih agenata.

Druga zapreka ističe kako većina projekata višeagentskih sustava ulaže značajan razvojni napor u izgradnju osnovne strukture prije nego li se uopće istraže mogućnosti i odgovarajući alati potrebni za razvoj agenata. Prije početka razvoja sustava, potrebno je istražiti trenutno stanje i prilagoditi se postojećim tehnologijama. Tu se javlja velika potreba za odgovarajućim kvalitetnim alatima koji će omogućiti specificiranje različitih karakteristika inteligentnog agentovog kao što su način rješavanja problema, interakcija između agenata, vizualizacija problema i cjelokupnog sustava, te traženje pogrešaka u načinu rješavanja problema od strane agenata.

Usprkos ovim zaprekama, možemo reći da višeagentski sustavi imaju mnogo toga za ponuditi u vidu programskih rješenja. Ovo mišljenje je u skladu s činjenicom da je veliki broj organizacija aktivno uključen u istraživanje i razvoj ovog područja [21].

## 3.1 Definicija agenta

Višeagentskih sustavi, kako smo već naveli, sastoje se od određenog broja agenata koji međudjeluju jedan s drugim, najčešće izmjenjujući poruke putem infrastrukture računalne mreže. U najopćenitijem slučaju, agenti predstavljaju određeni interes, odnosno djeluju u korist korisnika s različitim ciljevima i motivacijama. Kako bi interakcija bila uspješna, agenti zahtijevaju sposobnost da međusobno surađuju, koordiniraju i pregovaraju kao što to ljudi čine u svakodnevnom životu [21].

Unatoč vrlo širokoj primjeni i čestom korištenju pojma „agent“, trenutno ne postoji univerzalno prihvaćena definicija tog pojma. Generalno je prihvaćeno da je glavna ideja agenta autonomnost, ali osim toga ne postoji neko općenito shvaćanje pojma agenta. Djelomičan uzrok tome je mnoštvo karakteristika koje mogu biti povezani sa agentima i različitih problema koji se rješavaju uz pomoć agenata. Određene karakteristike mogu biti od različitog značaja za rješavanje problema, ovisno o samoj domeni u kojoj se agent nalazi. Na primjer, u određenim primjenama mogućnost agenta da uči iz vlastitog iskustva je neophodna i ključna za rješavanje problema, dok je u drugima učenje ne samo ne važno, nego u nekim slučajevima može biti i nepoželjno.

Usprkos tome, potrebna je barem neka opća definicija da sami pojam „agent“ ne bi izgubio smisao. Jedna od općenitih definicija [22] navodi da je agent: „Računalni sustav koji se nalazi u nekom kompleksnom dinamičkom okruženju i koji je sposoban autonomno djelovati unutar tog okruženja sa namjerom izvršavanja zadanog cilja“. Može se primijetiti da je ova definicija prilično poopćena i ne previše detaljna. Za početak, definicija se odnosi na pojam „agent“, a ne na pojam „inteligentni agent“ što je važna razlika. Da bi se agent smatrao inteligentnim potrebno je da sadrži dodatne karakteristike koje će biti navedene kasnije u radu. Druga stvar je da definicija samo navodi da se agent nalazi u nekom okruženju ali ne navodi nikakve karakteristike tog okruženja. Agenti se mogu nalaziti u različitim tipovima okruženja te će se kasnije u radu opisati kakve sve mogu biti karakteristike okruženja u kojima se agent nalazi. Posljednja stvar je nedostatak definicije pojma autonomnosti, koje uvjete agent mora zadovoljavati da bi ga se smatralo autonomnim. To je također dosta općeniti pojam kojega nije tako lako definirati ali se u kontekstu agenata da autonomnost znači da agenti imaju sposobnost djelovati bez intervencije ljudi ili drugih sustava te da imaju kontrolu nad svojim unutarnjim stanjem i svojim ponašanjem.

Uz već navedenu definiciju, agenti se generalno mogu podijeliti na slabu i na jaku predodžbu agenta [2].

### 3.1.1 Slaba predodžba agenta

Predstavlja općeniti pristup prikazivanja agenta kao dio hardvera ili (češće) softverski temeljenog računalnog sustava koji sadrži sljedeća svojstva:

- autonomnost: agenti djeluju bez direktne intervencije ljudi ili drugih agenata, te imaju kontrolu nad svojim akcijama i unutarnjim stanjem [23].
- društvenost: agent posjeduje sposobnost interakcije sa drugim agentima (eventualno i sa ljudima) putem nekog komunikacijskog jezika [24].
- reaktivnost: agenti imaju sposobnost opažanja svog okruženja (neovisno o kakvom se okruženju radi) te pravovremenog djelovanja na promjene koje nastaju u okruženju.
- proaktivnost: agenti ne djeluju samo na temelju podražaja iz okruženja već imaju sposobnost prikazati ponašanje orijentirano na izvršavanje zadanog cilja.

Ovakav način predstavljanja agenata je našao upotrebu u velikom broju istraživanja. U računalnoj znanosti se predodžba agenta kao samostalnog softverskog procesa sa neprestanim izvršavanjem i mogućnošću komunikacije čini kao prirodni nastavak objektno-orijentirane programske paradigme [25, 26]. Slaba predodžba agenta se također koristi kod agentski temeljenog softverskog inženjerstva [24, 27].

### 3.1.2 Jaka predodžba agenta

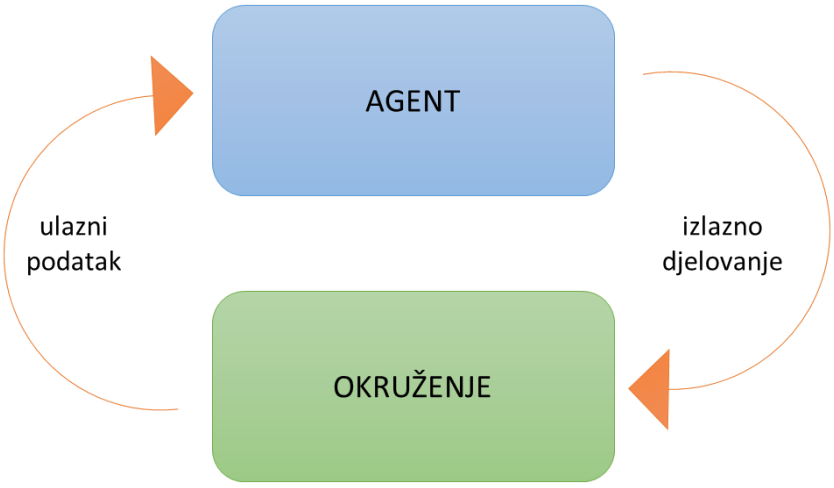
U određenim istraživanjima, pogotovo onima vezanim uz područje umjetne inteligencije, pojam agent ima detaljniju i konkretniju definiciju agenta. Kod takvih istraživanja agent se navodi kao računalni sustav koji, uz prethodno navedene karakteristike, je dizajniran i implementiran na način da koristi koncepte koji su općenito povezani sa ljudima. Na primjer, u području umjetne inteligencije, često se sa agentima povezuju razne mentalne karakteristike kao što su znanje, uvjerenja, namjere i obaveze [28]. Neki istraživači iz područja umjetne inteligencije čak uvode pojam emocionalnih agenta [29] koji su dizajnirani na način da njihova unutarnja stanja odgovaraju ljudskim mentalnim stanjima. Drugi pristup davanja agentima ljudske karakteristike je vizualno ih predstaviti, na primjer korištenjem crteža ili animacija [30]. Takvi agenti su posebno značajni u istraživanjima iz područja interakcije između čovjeka i računala.

### 3.1.3 Ostale karakteristike agenta

Uz prethodno navedene karakteristike, prilikom definiranja agenata često se navode i sljedeće karakteristike:

- mobilnost: predstavlja sposobnost agenta da se kreće po prostoru.
- vjerodostojnost: pretpostavlja se da agent neće svjesno proslijediti lažnu informaciju drugim agentima ili korisniku sustava [31].
- dobronamjernost: pretpostavlja se da agenti nemaju konfliktne interese i da će u skladu s tim uvijek izvršiti svoj zadani cilj [32].
- racionalnost: je pretpostavka se će agent djelovati sa ciljem izvršavanja svog zadatka, i da neće izvršavati akcije koje bi ga mogle spriječiti u tome.

Ovisno o kontekstu i području u kojem se koristi pojam agenta postoji još mnoštvo različitih karakteristika koje se navode pri samoj definiciji tog pojma [33]. Unatoč tome, na temelju svega navedenoga može se izvući opća definicija agenta. Slika 1. prikazuje apstraktno viđenje agenta. Agent je predstavljen kao neki entitet koji se nalazi u određenom okruženju. Uz pomoć osjetila (senzora) agent prima podatke iz svog okruženja. Na temelju tih podataka agent generira izlazno djelovanje kojim utječe na vlastito okruženje. U većini slučajeva agent nema kontrolu nad cijelim svojim okruženjem, već u najboljem slučaju ima kontrolu i mogućnost utjecaja nad samo jednim dijelom okruženja.

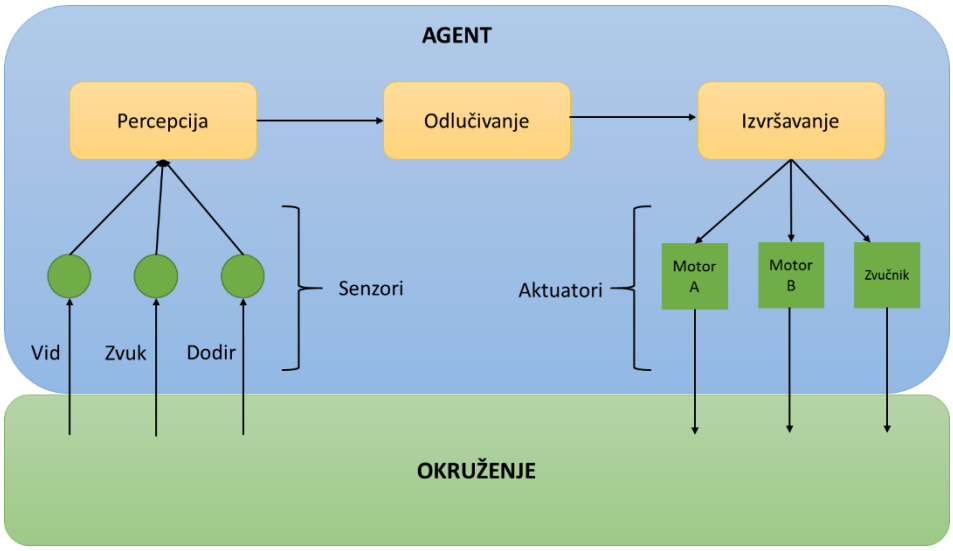



*Slika 1. Agent u svom okruženju. Agent uzima ulazni podatak iz okruženja i pretvara ga u izlazno djelovanje usmjereno na okruženje. Interakcija je obično neprekidna*

Agenti za djelovanje na okoliš imaju dostupan skup akcija koje mogu izvršiti. Skup akcija predstavlja sposobnost robota da djeluje na svoje okruženje. Naravno nije uvijek moguće izvršiti sve akcije, već to ovisi o trenutnom stanju okruženja ali i samog agenta. Također, ovisno o tipu okruženja, izvođenje iste akcije ne mora uvijek uzrokovati istim rezultatom. Tada se kaže da se radi o nedeterminističkom okruženju jer stanje okruženja ne ovisi samo o akcijama agenta ali detaljniji opis okruženja će se dati u narednim poglavljima

## 3.2 Agentske arhitekture

U prethodnim poglavljima su opisane karakteristike agenata i različite definicije tog pojma. Sada će se naglasak staviti na konkretnu implementaciju agenata u svrhu rješavanja nekog problema. U takvim slučajevima potrebno je definirati arhitekturu samog agenta. Meas definira arhitekturu agenta kao „Određenu metodologiju za izgradnju agenata. Ona definira kako se agent može raščlaniti na izgradnju skupa elementarnih modula i način interakcije između tih modula. Skup svih modula i njihove interakcije moraju pružiti odgovor na pitanje kako senzorske vrijednosti i trenutno unutarnje stanje agenta određuju akcije i buduće unutarnje stanje agenta. Arhitektura obuhvaća metode i algoritme koji podupiru ovu metodologiju“ [34]. Slika 2 ilustrira općeniti prikaz agentske arhitekture.




*Slika 2. Prikaz općenite arhitekture agenta*

Kaelbling u svom radu definira arhitekturu agenta kao: „Specifični skup softverskih (ili hardverskih) modula, sa definiranim smjerom kontrole toka i toka podataka između tih modula. Apstraktna definicija predstavlja arhitekturu kao generalnu metodologija za izradu modularnih dekompozicija za određene zadatke [35]“.

Agentske arhitekture se mogu podijeliti u tri opće kategorije: racionalni agenti, reaktivni agenti i hibridni agenti.

### 3.2.1 Racionalni agenti

Arhitektura racionalnih agenata se smatra i „od vrha prema dolje“ arhitekturom (*eng- top-down)*. Ovaj pristup raščlanjuje percepciju, zaključivanje i izvršni ciklus. Motivacija za ovakav tip arhitekture je da se apstrakcijom mogu sakriti funkcije niže razine od onih na višoj razini. Arhitekture racionalnih agenata imaju eksplicitni model svijeta i fokusiraju se na razvijanje jedne strategije te njeno izvršavanja do kraja izvođenja. Ovakvi sustavi su dobri kod planiranja i zaključivanja na višim razinama ali nisu dovoljno reaktivni za dinamička okruženja. Da bi se zaobišao ovaj problem, razvijene su ekstenzije čistih racionalnih arhitektura. Jedan od primjera tih ekstenzija su stabla ponašanja. Stablo ponašanja je skup definiranih ponašanja koja su organizirana u obliku stabla. Kompleksna ponašanja se predstavljaju grananjem u manja i jednostavnija ponašanja. Dubina stabla ovisi o kompleksnosti najsloženijeg oblika ponašanja, dok širina stabla ovisi o samom broju ponašanja. Stabla ponašanja su vrlo korisna za upravljanje kompleksnosti sustava jer je prijelaz iz jedne grane u drugu granu na istoj razini jednostavan. Nedostatak je što nije jednostavan prijelaz između dva stanja koja se nalaze u različitim granama ako grane nisu na istoj razini.

### 3.2.2 Reaktivni agenti

Arhitektura reaktivnih agenata predstavlja tzv. „od dna prema vrhu“ (*eng. bottom-up“*) arhitekturu. Ovakve arhitekture su suprotnost racionalnim arhitekturama. Umjesto višestrukih razina apstrakcije prilikom percepcije, odlučivanja i izvršavanja, ovakvi tipovi arhitekture sadrže jednostavna ponašanja koja su direktno vezana na aktuatorske naredbe (slično ljudskim refleksima). Kompleksnija ponašanja se stvaraju kombiniranjem više jednostavnijih ponašanja. Reaktivne arhitekture imaju sposobnost brzo reagirati na okruženje (npr. zaobići prepreku) zbog direktne veze između senzora i aktuatora. Nedostatak ovakvih arhitektura je da je u većini slučajeva teško unaprijed znati koje su akcije niske razine potrebne, te je također teško predvidjeti interakciju između više ponašanja.

Potpuno reaktivne arhitekture imaju skup ponašanja niske razine, koja mogu reagirati na određene situacije kada se dogode. Glavna prednost korištenja ovakvih sustava je da su

učinkoviti po pitanju računske složenosti. Glavni nedostatak je da u većini slučajeva ovakvi sustavi nemaju mehanizme planiranja ili donošenja odluka na višoj razini.

Jedan od prvih poznatijih primjera ovakve arhitekture je Brooks-ova slojevita arhitektura [36] za upravljanje robotima i njihovim ponašanjima. Ova arhitektura ne koristi eksplicitni model svijeta kao racionalne arhitekture. Glavni koncept ove arhitekture je da su ponašanja robota definirana na slojevit način. Svaki sloj predstavlja jedan asinkroni modul, te slojevi više razine imaju mogućnost premostiti slojeve niže razine. To se izvodi ili sprječavanjem ulaznih signala ili potiskivanjem izlaza. Ovaj pristup omogućuje robusnost sustava prilikom dodavanja novih ponašanja. Nedostatak je što se dodavanjem novih slojeva više razine uvelike povećava složenost dizajniranja sustava. Unatoč tome, arhitekture ovog tipa se i danas uspješno koriste u robotskom nogometu i drugim primjenama [37, 38].

### 3.2.3 Hibridni agenti

Hibridne arhitekture su mješavina između racionalnih i reaktivnih arhitektura. Ovo je najpopularnija vrsta arhitektura zato što iskorištavaju najbolje karakteristike oba pristupa [39, 40]. Unatoč tome, također posjeduju i nedostatke oba pristupa te je potrebno naći dobar balans između te dvije arhitekture.

Kod hibridnih arhitektura, dio senzora i percepata je direktno vezan uz aktuatore, dok se ostatak detaljnije obrađuje. Umjesto kompletnog modela svijeta, sustav se postavlja u konačan skup stanja. To znači da se percepti iz okoliša mapiraju u neka određena stanja u sustavu. Sustav zaključivanja postavlja agenta u željeno stanje. Na primjeru robota koji igra nogomet, ponašanje „pucaj na gol“ se sastoji od tri stanja (dođi do lopte, okreni se prema njoj, udari loptu). Glavna prednost ovakve arhitekture je da je jednostavno se prebacivati iz jednog u drugo stanje na istoj razini (npr. ako je robot već okrenut prema lopti u trenutku kada dođe u njenu blizinu, može je odmah udariti).

Jedna od najpoznatijih hibridnih arhitektura je BDI arhitektura [41]. BDI agenti imaju tri glavne karakteristike – uvjerenja (*eng. beliefs)*, želje *(eng. desires*) i namjere (*eng. intentions*). Uvjerenje se odnose na činjenice o sebi i svom okruženju za koje agent smatra da su ispravne. Želje se odnose na ciljeve agenta, dok namjere predstavljaju korake koja agent planira izvršiti da bi ispunio svoje ciljeve ili želje. Uvjerenja i želje pomažu agentima stvaranje dugotrajnih strateških planova dok kombinacija namjera i želja pridonosi agentovoj sposobnosti za improvizaciju.

Decugis i Ferber su također opisali jednu vrstu autonomne hijerarhijske distribuirane arhitekture koja podržava i reaktivnost i planiranje [42]. Njihova arhitektura koristi mrežu ponašanja sa fleksibilnom aktivacijskom funkcijom koja služi za promjenu trenutnog ponašanja.

### 3.3 Agenti u višeagentskom sustavu

Višeagentski sustavi, kao što im samo ime govori, sadrže više od jednog agenta. Svi agenti unutar jednog sustava čine populaciju tog sustava. Populacija agenata ima četiri glavne karakteristike [43]:

- Veličina : Broj agenata unutar sustava
- Raznolikost: Navodi da li su agenti koji čine populaciju istog tipa ili se radi o različitim tipovima agenata. Populacija je homogena ako su svi agenti istog tipa tj. heterogena ako sustav sadrži barem dva različita tipa agenata.
- Struktura cilja: Definira tipove i uniformnost ciljeva. U slučaju da sustav ima definiran samo jedan cilj, svi agenti unutar okruženja djeluju u skladu sa tim ciljem. Cilj može biti slojevito definiran, sa pod-ciljevima koji mogu varirati unutar populacije. Jedinstveni cilj unutar višeagentskog sustava ne znači nužno i kooperaciju agenata u svrhu postizanja tog cilja. Agenti u višeagentskom sustavu mogu imati različite ciljeve. U većini slučajeva se tada radi o heterogenim agentima ali moguće je i da homogeni agenti imaju različite ciljeve. Također je moguće i da jedan agent ima više ciljeve koje pokušava postići.
- Kooperativnost: Definira sklonost agenata da surađuju. Kod većine višeagentskih sustava dolazi do neke vrste interakcije između agenata. Kada je u pitanju suradnja, agente dijelimo na četiri osnovna tipa:
    1. Kooperativni – agent je voljan pomoći drugim agentima da postignu svoj cilj, potencijalno žrtvujući postizanje vlastitog cilja.
    2. Kompetitivni – agenti primarno djeluju sa svrhom izvršavanja svog cilja i ne surađuju ako to utječe na izvršavanje njihovih zadataka.
    3. Nezavisni – agenti djeluju neovisno, bez ikakvih mehanizama suradnje ili kooperacije.
    4. Prioritetni – ciljevi unutar sustava mogu imati zadane prioritete koji onda određuju ponašanje i suradnju agenata.

### 3.4 Okruženje

Okruženje možemo opisati kao sve što se nalazi u okolici agenta, ali je odvojeno od samog agenta i njegovog ponašanja, ili jednostavnije rečeno, okruženje je sve što se nalazi u okolici agenta a nije dio samog agenta. To je prostor u kojem agent "živi" i djeluje, kojeg percipira i u kojem se kreće i po tom pitanju okruženje je vrlo slično stvarnom svijetu koji nas okružuje.

Karakteristike okruženja kojeg agent percipira i u kojem manifestira svoje djelovanje igraju veliku ulogu u određivanju složenosti agenata prilikom procesa dizajniranja. Slijedi klasifikacija okruženja s obzirom na određene karakteristike:

- *Dostupna* naspram *nedostupnih*. Dostupno okruženje je ono u kojem agent ima pristup potpunim, preciznim i aktualnim informacijama o trenutačnom stanju okruženja. Takva okruženja se još nazivaju i potpuno vidljiva okruženja. Agenti u nedostupnim (djelomično vidljivim) okruženjima imaju dostupan samo dio informacija vezanih uz okruženje u kojem se nalazi. Većina okruženja u stvarnom svijetu (na primjer: svakodnevni fizikalni svijet, *Internet*) nisu dostupna na opisani način.
- *Deterministička* naspram *nedeterminističkih*. Determinističko okruženje je ono u kojem svako djelovanje ima pridruženu posljedicu – ne postoji neizvjesnost oko rezultirajućeg stanja nakon obavljenog djelovanja. Nedeterministička okruženja su ona kod kojih na temelju trenutnog stanja okruženja i agenta, te akcije koju će agent izvršiti, ne možemo unaprijed predvidjeti sljedeće stanje.
- *Statička* naspram *dinamičkih*. Statičko okruženje je ono za koje se pretpostavlja da se ne mijenja osim pod djelovanjem agenta. Suprotno tome, dinamičko okruženje je ono koje ima druge procese koji djeluju na njega te stoga može doživjeti promjene van kontrole agenta (na primjer, fizikalni svijet je visoko dinamičko okruženje).
- *Diskretna* naspram *kontinuiranih*. Okruženje je diskretno ako postoji nepromjenjiv, konačan broj akcija i uputa u njemu.

Najsloženija skupina okruženja su ona koja su *nedostupna*, *nedeterministička*, *dinamička* i *kontinuirana*. Okruženja s takvim karakteristikama često se nazivaju *otvorena* okruženja.

Fizikalna svojstva virtualnog okruženja mogu biti prilagođena kako bi zadovoljila potrebe dizajnera. Na primjer, moguće je da avatar (računalno generirani agent koji predstavlja čovjeka) leti unutar virtualnog okruženja, ili da, kao u nekim računalnim igrama, ima mogućnost teleportacije što je, naravno, nemoguće za ljude u stvarnom životu. Isto tako,

okruženje može biti prikaz nečega što se ne može jednostavno i uvjerljivo prikazati u stvarnosti kao na primjer računalna mreža. Okruženje može biti i simulacija realnog okruženja, gdje je cilj simulirati pojedine fizikalne pojave i svojstva što je bolje moguće. Problem sa simuliranim okruženjima je taj što je često vrlo teško postići realnost u simulaciji, naročito jer simulacija može divergirati od realnosti u nekom nepredvidivom smjeru.

### 3.4.1 Egocentrični pogled na svijet

Da bi se agenti mogli snalaziti u svom okruženju, moraju na neki način spremiti i prikazati informacije koje dobivaju iz okoliša putem svojih percepata. Ovisno o svojim karakteristikama, agenti mogu na različite načine percipirati svoje okruženje. Unutarnji prikaz svog okruženja kojeg svaki agent posjeduje se naziva mentalni model svijeta. Mentalni modela svijeta predstavlja prostorne relacije između objekata u okruženju. Način na koji se predstavljaju te prostorne relacije u odnosu na agenta određuje da li se radi o *egocentričnom* ili *alocentričnom* pogledu na svijet tj. okruženje. Kod alocentričnog prikaza, prostorne relacije su predstavljene između svakog objekta $\alpha$ i svih ostalih objekata unutar zapaženog okruženja. To znači da su svi objekti nezavisni o položaju samog agenta. Zbog toga, prostorne relacije objekata nije potrebno osvježavati, osim u slučaju da se radi o dinamičkom okruženju. Suprotno tome, egocentrični pristup definira prostorne relacije u ovisnosti o samom agentu tj. njegovom položaju unutar okruženja. Svaka promjena pozicije agenta u egocentričnom prikazu svijeta uzrokuje potrebu osvježavanja svih prostornih relacija [44]. Kako se kod konkretne primjene inteligentnih agenata u većini slučajeva radi o dinamičkom prostoru, prostorne relacije se svakako moraju osvježavati u određenom intervalu, neovisno o odabranom pristupu predstavljanja svijeta.

Za primjenu u daljnjem istraživanju odabran je egocentrični pristup zato što omogućuje agentu da pamti samo prostorne relacije koje su mu važne za rješavanje trenutnog problema. Ostale prostorne relacije se mogu zanemariti. Agent je na taj način fokusiran samo na postizanje svog trenutnog cilja. Korištenje alocentričnog pristupa bi zahtijevalo od agenta da pamti sve prostorne relacije, neovisno o njihovoj važnosti za rješavanje trenutnog problema. Uvođenjem egocentričnog pristupa moguće je poboljšati preciznost mentalnog modela jer se smanjuju memorijski zahtjevi agenta što omogućuje veću razinu diskretizacije prostora.

Egocentrični pristup pruža još nekoliko dodatnih prednosti u odnosu na alocentrični. Za početak, broj prostornih relacija koje se uzimaju u obzir pri donošenju odluka može biti

znatno veći kod alocentričnog prikaza [44], što naravno uzrokuje veću računsku složenost. Druga prednost je vezana uz definiranje ponašanja izbjegavanja prepreka. Mnogo je jednostavnije izračunati vektor smjera kojeg agent treba zaobići ako se prilikom izračuna koriste smjer i udaljenost objekata od samog agenta. [45]. Na kraju, egocentrični pristup omogućava jednostavnije shvaćanje i prikaz ograničenja samog agenta.

Egocentrični pogled se temelji na sljedećim svojstvima:

- Pozicioniranost. U svakom vremenskom trenutku ponašanje agenta je primarno određeno i ograničeno s obzirom na trenutni položaj samog agenta.
- Veza između fizičkih objekata i njihove mentalne reprezentacije. Kada agent primijeti neki objekt, prikazuje ga unutar svog mentalnog prikaza svijet koristeći određeni koncept. Agent ne zna točnu lokaciju objekta ali ima određenu aproksimaciju koja ovisi o preciznosti senzora i samom načinu mentalne reprezentacije svijeta.
- Važnost mentalnog modela. Prilikom određivanja sljedeće akcije, agent mora uzeti u obzir sve što je trenutno predstavljeno u njegovom mentalnom modelu svijeta.
- Promjenjivost okruženja i odnosa agent – okruženje. Promjene u pozicioniranju agenta i modifikacije određenih dijelova okruženja moraju biti uzete u obzir prilikom donošenja odluka. Agent može posjedovati djelomičnu ili potpunu vidljivost okruženja u kojem se nalazi, ovisno o tipu i dometu senzora koje posjeduje te o dimenzijama svijeta u kojem se nalazi. Navedena svojstva se temelje na radu [46].

### 3.4.2 Labirint

Labirint je jedan od primjera okruženja agenta koji se često koristi u različitim sustavima vezanim uz rješavanje problema i strojno učenje [47]. Ovakav tip okruženja je također prikladan za testiranje učinkovitosti različitih algoritama pretrage. Problem pretrage prostora stanja se može prikazati pomoću labirinta na način da svako polje u labirintu predstavlja jedno stanje sustava. Zbog navedenih karakteristika, labirint kao okruženje je posebno izdvojen u ovom radu te su navedene općenite karakteristike takvog okruženja.

Labirint (eng. *maze*) je dvodimenzionalno područje u obliku rešetke bilo koje veličine, obično pravokutnog oblika. Labirint se sastoji od ćelija i može sadržavati različite prepreke (zidove) u bilo kojoj količini. Agent se smješta u labirint na praznu ćeliju i smije se kretati u svim smjerovima, ali samo kroz prazan prostor.

U nastavku su navedene i objašnjene agentski neovisne karakteristike labirinta [47]:

- *Veličina* – Broj ćelija labirinta očito utječe na složenost. Veličinu labirinta $m$ označavamo $s_m$.
- *Udaljenost do cilja* – Prosječna udaljenost do cilja ($\phi_m$) u labirintu je važna karakteristika složenosti. Što je veća vrijednost, labirint je teži.
- *Prepreke* – Broj prepreka u labirintu, $o_m$, je definiran kao zbroj svih unutarnjih zidova i pola vanjskih zidova. Prema tome, za labirinte sa vanjskim zidovima, prilagođava se broj prepreka da bi se omogućila činjenica da vanjski zidovi predstavljaju prepreku samo iz jednog smjera.
- *Gustoća* – Gustoća labirinta je proporcija prepreka i veličine, $\delta_m = \frac{o_m}{s_m}$ . Složenost labirinta je veća kada je gustoća manja.

Najvažnije agentski neovisne karakteristike labirinta koje utječu na složenost problema su veličina, gustoća i udaljenost do cilja.

## 3.5 Predstavljanje agenta

Agenti, kao što je navedeno u prethodnim poglavljima, mogu biti realizirani kao hardverski ili softverski sustav. Također, postoje i sustavi kod kojih su agenti predstavljeni i u fizičkom i u simuliranom obliku. U ovom poglavlju će se opisati neki od navedenih načina predstavljanja agenata.

### 3.5.1 Roboti

Jedan od načina predstavljanja agenata u fizičkom svijetu je korištenjem robota. U današnje vrijeme se roboti najčešće koriste u industriji, prilikom različitih proizvodnih procesa. Roboti pri tome služe za obavljanje mnoštva zadataka kao što su sastavljanja dijelova, automatske proizvodne trake, transport materijala, kontrola procesa, skladištenja materijala i mnoge druge primjene [48]. Unatoč tome, u ovom radu će se naglasak staviti na primjenu robota u edukacijske svrhe i na taj način se povezati sa temom učenja kod inteligentnih agenata. Bolje

shvaćanje načina na koji se roboti koriste u edukaciji (prijenos znanja) može pomoći dizajniranju arhitekture agenta koji će imati sposobnost dijeljenja znanja sa ostalim agentima.

Seymour Papert je jedan od prvih koji koristio robota [49] za predstavljanje agenta u edukacijske svrhe. Danas postoji mnoštvo komercijalno dostupnih robota, kao i platformi za izgradnju vlastitih sustava. U ovom pregledu ćemo se fokusirati samo na jednu vrstu robota. Radi se o Lego Mindstorms robotima. Više je razloga za odabir ove platforme. Za početak, sam Papert je naveo Lego kao dobru podlogu za korištenje u sklopu sa simulacijama [50]. Osim toga, Lego Mindstorms roboti se planiraju koristiti kao nastavak već započetog istraživanja [51, 52], te je to glavni razlog zbog čega je poseban naglasak stavljen na ovaj tip robota. Također, Lego Mindstorms roboti pružaju mogućnost konstrukcije različitih robota što olakšava fizičku implementaciju višeagentskog heterogenog sustava. Slika 3. prikazuje jedan oblik konstrukcije robota.


*Slika 3. Primjer Lego Mindstorms robota.*

Lego Mindstorms roboti su često bili predmet istraživanja u prošla dva desetljeća. Većina tih istraživanja se fokusira na prednosti i nedostatke korištenja Lego Mindstorms robota u edukacijske svrhe, pri čemu je moguća primjena u različitim područjima, od računalne znanosti [53], strojarstva [54], programiranja [53, 55], mehatronike [56], umjetne inteligencije [57] i mnogim sličnim. Također, ovi roboti se mogu integrirati u sklopu nastavnog programa na svim edukacijskim razinama, počevši od osnovne škole [58] pa sve do fakulteta [53, 54, 56, 57].

Pregled relevantne literature upućuje na to da se Lego Mindstorms roboti mogu na više načina koristiti u edukacijske svrhe: za podučavanje osnovnih koncepata vezanih uz računalnu znanost, programiranje, umjetnu inteligenciju, strojarstvo i robotiku; za razvoj matematičkog i logičkog razmišljanja; razvoj vještina potrebnih za rješavanje problema; te kao motivacijski alat kako bi se postigao veći angažman učenika za vrijeme nastave. Korištenje Lego robota u nastavi je inspirirano konstruktivističkom teorijom učenja [59] koja tvrdi da se znanje aktivno stvara od strane studenta, a da nije samo pasivno apsorbirano iz udžbenika i predavanja [60].

Cliburn u svom istraživanju [53] koristi Lego Mindstorms robote da bi studente podučio osnovnim konceptima računalne znanosti te uvodnom ali i naprednom programiranju. Lego roboti u pravilu povećavaju užitak slušanja kolegija računalne znanosti ali se također navodi na nisu baš uvijek prikladni za primjenu u nastavi. Kod primjene robota za podučavanje osnovnog ili naprednog programiranja, učitelj mora pronaći odgovarajuće sučelje koje će se koristiti za programiranje robota te također mora pažljivo odabrati projekte tj. zadatke koje će koristiti u takvom obliku nastave. Najuspješnija implementacija robota u nastavi se odnosila na osnovne koncepte računalne znanosti. Roboti mogu biti vrlo korisni za korištenje prilikom podučavanja ali samo u odgovarajućem kontekstu.

Fagin i Merkle su proveli jednogodišnji eksperiment [61] u sklopu kojega su koristili robota za podučavanje računalne znanosti. Studenti su podijeljeni u dvije grupe, jednu koja je koristila robote te drugu bez robota. Rezultati istraživanja pokazuju da je grupa kod koje su korišteni roboti na kraju imala lošije rezultate od grupe bez robota.

Klassner je proveo analizu [57] u kojoj ispituje prikladnost Lego Mindstorms robota kao hardverske platforme za integriranje robotike u kolegij umjetne inteligencije. Studenti su radili na nekoliko timskih projekata. Neki od navedenih projekata su se temeljili na Mindstorms i NQC programskom jeziku, dok su neki bili napravljeni kodiranjem u Lisp-u. Studenti su izjavili da je uvođenje problematike robota u sklopu kolegija pozitivno djelovalo na njihovo učenje. Zaključak istraživanja je da su projekti vezani sa robotom doprinijeli vrlo dobrom razumijevanju koncepata koji su bili određeni kao projektni zadaci.

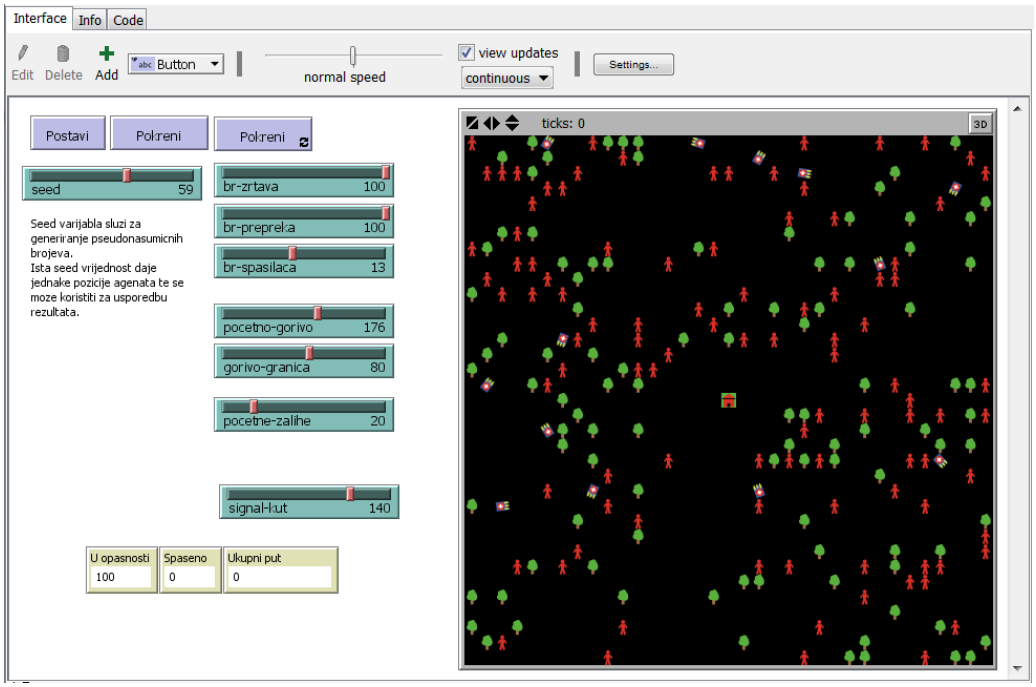
### 3.5.2 Simulacije

Osim već opisanog fizičkog predstavljanja agenata, agenti mogu biti potpuno virtualni (softverski agenti). Često se pri izradi nekog sustava (bilo da se radi o jednom agentu ili višeagentskom sustavu) koristi simulacija samog tog sustava. Kada se radi o simulaciji softverskog agenta, tada nam simulacija služi da testiramo različite parametre i karakteristike

samog agenta u zadanom i sigurnom okruženju. Dodatna prednost kod simulacija višeagentskih sustava je da se mogu testirati interakcije između različitih agenata i na taj način otkriti moguće probleme, ali i neočekivane obrasce ponašanja [62].

Simulacije se također mogu koristiti i za prikaz sustava koji uključuje fizičke agente. Pri tome se opet kao glavni motiv ističe testiranje sustava prije nego se krene sa njegovim korištenjem. Korištenjem simulacija za testiranje sustava sa fizičkim agentima mogu se izbjeći mnogi problemi kao što su oštećenja opreme, nepotrebna potrošnja energije ili prostorno-vremenska složenost.

U prethodno dijelu je stavljen naglasak na fizičke agente (robote) koji se koriste u edukacijske svrhe. Sukladno tome, biti će navedeni primjeri simulacijskih okruženja koja služe za prikaz robota koji se koriste u edukaciji. Glavni motiv za izradu ovakvih simulacijskih okruženja je veća dostupnost u odnosu na stvarne robote. Da bi se roboti mogli uspješno implementirati u učionicama potrebno je da svaki učenik ima svoj primjerak robota te dovoljno vremena za konstrukciju tj. izradu svog robota. Naravno, to predstavlja problem za manje škole i fakultete koji nemaju dovoljno sredstava ali i za one koji imaju veliku broj upisanih studenata/učenika. Da bi se zaobišao taj problem, napravljena su mnoga okruženja koja dozvoljavaju simulirani prikaz robota, kao što su Netlogo [63], EDURobot [64], RoboKol [65], RoboXAP [66] i slični. Slika 4. prikazuje simulaciju robota koji se nalazi u višeagentskom sustavu. Navedeni primjer je realiziran unutar NetLogo okruženja.




*Slika 4. Simulacija robota u višeagentskom sustavu unutar Netlogo okruženja*

Abiyev, Erin i Ibrahim su razvili EDURobot [64], edukacijski softverski alat namijenjen kojemu je cilj unaprijediti shvaćanje pojmova iz robotike studentima preddiplomskih i diplomskih studija. EDURobot je razvijen da bi podučio studente različitim algoritmima koji se koriste prilikom problema navigacije robota kroz prostor uz izbjegavanje prepreka. Autori su proveli istraživanje da utvrde mišljenje studenata za vrijeme korištenja EDURobot simulacije. Rezultati pokazuju da je navedeni softverski alat povećao razinu znanja i razumijevanja studenata iz područja robotike, te im pružio bolji uvid u različite algoritme za planiranje putanje i navigaciju. Svi studenti su bili složni da je korištenje simulacije učinkovit način podučavanja.

Yadin [67] je osmislio kolegij koji koristi vizualno okruženje za podučavanje koje simulira dvodimenzionalni svijet koji sadrži robota sposobnog izvršavati različite zadatke, istovremeno izbjegavajući prepreke koje mu se nalaze na putu. Studenti su dobili nekoliko različitih zadataka. Za svaki zadatak koji su dobili, bilo je potrebno verbalno definirati problem te predstaviti koncept potencijalnog rješenja. Nakon toga, studenti su definirali niz instrukcija za robota. Istraživanje je pokazalo da vizualno okruženje nije dovoljno za pojednostavniti apstraktne koncepte.

### 3.5.3 Hibridni oblik predstavljanja agenta

Glavni nedostatak većine simulacijskih okruženja je što nemaju povratnu vezu sa stvarnim sustavom, neovisno da li se radi o virtualnom ili fizičkom sustavu. Upravo zbog toga, jedan od ciljeva budućeg istraživanja bi bio predstaviti hibridni model predstavljanja agenta koji bi uključivao simulirano okruženje, ali i fizičkog agenta koji bi bili međusobno povezani.

Svrha predstavljanja hibridnog modela agenta je iskoristi sve prednosti koje pruža simulirano okruženje te tome pridodati mogućnost demonstracije ponašanja agenta na način da se koristi robot kao fizička reprezentacija navedenog agenta.

Druga prednost ovakvog pristupa je da pruža razmjenu informacija između simuliranog i fizičkog dijela agenta na način da je omogućena dvosmjerna komunikacija. Ovime se izbjegava pretjerano pojednostavljivanje modela koje se često događa kada se koristi savršeno okruženje unutar simulacija. Ovakav pristup omogućava ne samo slanje podataka iz simulacije do fizičkog agenta nego i slanje povratne informacije natrag u simulirano okruženje (npr. očitane vrijednosti senzora, trenutni smjer ili koordinate). Dobiveni podaci se mogu koristiti unutar simulacije te imati utjecaja na konačni rezultat. Time se stvara hibridno okruženje koje nam dozvoljava da se do određene mjere premoste razlike između simuliranog i fizičkog okruženja. Posebno treba obratiti pozornost na sljedeće karakteristike

- **Nesavršeni senzori**: Model svijeta je ostvaren pomoću informacija dobivenih od strane senzora. Zbog toga preciznost mentalnog modela ovisi o preciznosti senzora. Nesavršeni senzori mogu uzrokovati da se mentalni model svijeta kojeg agent posjeduje razlikuje od stvarnog okruženja u kojem se agent nalazi.
- **Nesavršeni aktuatori**: Prilikom definiranja agenta u simuliranom okruženju često se pretpostavlja da agent posjeduje savršene aktuatori. Nažalost, to u stvarnom svijetu nije tako. Uvijek postoji mogućnost neke neočekivane pogreške kao što je npr. proklizavanje kotača. U tom slučaju bi agent pomakao kotače točno onoliko koliko mu je zadano ali se zbog proklizavanja ne bi našao na lokaciji na kojoj očekuje da se nalazi nakon izvršavanja te akcije. Na taj način agent više ne posjeduje točnu informaciju o svojoj stvarnoj lokaciji.
- **Zastarjeli model svijeta** (vremenski ograničena ili nesavršena memorija): Da bi agent imao sposobnost navigacije kroz dinamičko okruženje, njegov mentalni model svijeta se mora neprestano osvježavati. Time se podrazumijeva dodavanje novih informacija

u mentalni model svijeta, kao i brisanje zastarjelih podataka iz modela. U slučaju da se model svijeta ne osvježava, može doći do prevelike količine zastarjelih podataka i time učiniti sami model beskorisnim. To zatim može uzrokovati neočekivano ponašanje agenta i tako utjecati na njegovu sposobnost izvršavanja akcija i postizanja zadanog cilja.

### 3.6 Komunikacija između agenata

Važno svojstvo kod rješavanja problema korištenjem višeagentskog sustava je komunikacija između pojedinih agenata. Komunikacija se može definirati kao promjena stanja okoliša na način da ostali agenti mogu opaziti tu promjenu i njenim dekodiranjem dobiti informaciju. Komunikacija između agenata je nužna da bi agenti međusobno mogli bolje koordinirati, dijeliti podatke o modelu svijetu i učiti jedni od drugih. U kontekstu višeagentskih sustava postavlja se pitanje da li je sustav sa agentima koji komuniciraju stvarno višeagentski sustav. Stone i Veloso [68] u svome radu navode da neograničena komunikacija reducira višeagentski sustav na nešto slično jednoagentnom sustavu. Svoj zaključak temelje na tome da u slučaju komunikacije bez ikakvih ograničenja, agenti mogu poslati informaciju o cjelovitom vanjskom stanju nekom centralnom agentu i izvršavati akcije u slijednim koracima na taj način djelujući kao aktuatori tog centralnog agenta. Korištenje centralnog agenta čak nije niti potrebno, dok god agenti mogu primati sve potrebne informacije o stanjima svih drugih agenata, sposobni su donositi samostalne odluke zato što mogu znaju što će ostali agenti napraviti. Na taj način zapravo svaki agent posjeduje centralni upravljački sklop koji odabire prikladnu pod-akciju potrebnu za izvršavanje zajedničke akcije. Takav pristup zahtijeva razmjenu velike količine informacija u kratkom vremenu što za sobom povlači veću složenost samog sustava. Zbog toga je potrebno definirati ograničenja komunikacije između agenata u višeagentskom sustavu. Eksplicitna komunikacija također može uzrokovati značajno povećanje prostora stanja pojedinog agenta (mora zapamtiti sva stanja svih ostalih agenata) i samih akcija koje može izvršiti. U tom slučaju se može dogoditi povećanje prostora stanja ima veći negativni utjecaj na pronalazak optimalnog rješenja nego što agent ima koristi od uvođenja komunikacije [69]. Može se zaključiti da je u slučajevima kada je komunikacija između agenata nužna za pronalazak optimalnog rješenja potrebno tu komunikaciju pojednostaviti ili djelomično zanemariti.

### 3.6.1 Direktna komunikacija

Većina implementacija komunikacijskih metoda u višeagentskim sustavima koristi neku vrstu eksterne komunikacijske koja dozvoljava da agenti međusobno dijele informacije između sebe. Takve metode mogu biti ograničene u smislu propusnosti podataka, latencije, lokaliteta, vrste agenata itd. Primjeri direktne komunikacije su signaliziranje i razmjena poruka. Kod implementacije komunikacije sustava, potrebno je razlikovati unaprijed definirane metode komunikacije te naučene metode komunikacije.

Tan [70] u svom istraživanju navodi da kooperativni agenti sa sposobnošću učenja mogu koristiti međusobnu komunikaciju na razne načine kako bi unaprijedili uspješnost izvođenja zadataka cijele grupe. Agenti tako mogu obavještavati jedni druge o svome trenutnom stanju na način da dijele trenutno očitane vrijednosti senzora. Drugi način bi bio kada agenti dijele informacije o prijašnjim iskustvima u obliku epizoda ( niz uređenih trojki <*stanje, akcija, nagrada*> koje je agent prethodno iskusio) koje koriste agentima koji se do tada nisu našli u navedenim situacijama.

Određena istraživanja, pogotovo u grani poduprtog učenja, pretpostavljaju da agenti imaju pristup zajedničkoj tablici sredstava ili zajedničkoj tablici pravila ponašanja kojima svatko može u novom koraku dodavati nove unose, iako se radi o različitim agentima. Berenji i Vengerov [71] istražuju okruženje kooperativnog učenja gdje više agenata koristi komunikaciju da bi koristili i osvježavali isti skup pravila. Autori tvrde da istovremeno osvježavanje centralnih pravila smanjuje ranu tendenciju konvergencije prema suboptimalnim ponašanjima. U ovom slučaju se radi o unaprijed definiranim komunikacijskim procedurama, agenti uče jedni druge već naučene informacije. Većina drugih istraživanja definira agente sa komunikacijskim kanalom ali bez unaprijed zadane svrhe. Kod jednostavnih implementacija, rječnik agenata se može sastojati samo od jednog signala kojega mogu detektirati ostali agenti [72]. U drugim radovima [73, 74], mobilni roboti posjeduju zadani ali nedefinirani komunikacijski rječnik. Roboti tada svakoj pojedinoj riječi dodjeljuju pripadajuće značenje koje dobiju nizom pokusa. Kako se mijenjaju okolnosti u kojima se robot nalazi, tako roboti prilagođavaju svoj rječnik novonastalim okolnostima. Steels u svom radu [75] ističe pojavljivanje spontanog koherentnog leksikona koji se za vrijeme životnog vijeka agenta može prilagoditi novim značenjima riječi. Steels i Kaplan se nadovezuju na to istraživanje [76] i prikazuju agente koji su sposobni definirati općenite riječi na način da se zajednički sporazume oko njihovog značenja i upotrebe. Sličan pristup evolucije komunikacijskog jezika

je opisan u velikom broju radova [77-81]. Mnoge slične metode ističu potrebu za motivacijom kako bi se uspostavila komunikacija između agenata (najčešće u vidu dijeljenja nagrade). Unatoč tome, Ackley i Littman su u svom radu [82] pokazali da motivacija u vidu nagrade nije uvijek nužna za uspostavu komunikacije između agenata. Rezultati njihovog istraživanja pokazuju da agenti uče međusobno komunicirati čak i kada nemaju nikakve izravne koristi od same komunikacije.

### 3.6.2 Indirektna komunikacija

Indirektne komunikacijske metode se definiraju kao one metode koje uključuju implicitni prijenos informacija od jednog agenta do drugog putem modifikacije okoliša. Primjeri indirektne komunikacije u stvarnom svijetu su ostavljanje tragova u snijegu ili kamenčića po putu.

Većina radova na temu indirektne komunikacije inspiraciju nalazi u korištenju feromona kod insekata koji označavaju put ili traže pomoć od drugih jedinki (agenata) [83]. Feromoni su zapravo kemijski spojevi čiju prisutnost i razina koncentracije mogu osjetiti drugi insekti [84]. Feromoni, slično kao i ostali oblici indirektne komunikacije, mogu duže vremena biti prisutni u okolišu ali isto tako mogu sa vremenom oslabiti ili potpuno nestati. Nekoliko različitih algoritama temeljenih na feromonima je predloženo, pogotovo u području problema sakupljanja (*eng. foraging problem*). Niz algoritama temeljenih na poduprtom učenju koriste fiksnu proceduru za postavljanje feromona i koriste trenutnu razinu feromona kao dodatnu senzorsku informaciju za vrijeme istraživanja prostora. [85-87]. Također su se koristile metode evolucijskog izračuna za učenje strategija istraživanja/iskorištavanja korištenjem ugrađenim mehanizama za polaganje feromona. Sauter i dr. u svom radu [88, 89] pokazuju kako se evolucijsko izračunavanje može koristiti za podešavanje pravila ponašanja agenta u sustavu sa više digitalnih feromona. Sličan princip je primijenjen i na problemu mrežnog usmjeravanja [90].

Druga istraživanja se bave problemom da li agenti mogu uz samo dobivanje informacija iz feromona, naučiti i racionalno polagati feromone u okruženju. Taj problematika je prvo obrađena u sustavu AntFarm koji kombinira komunikaciju putem feromona i genetsko računanje [91, 92]. AntFarm sustav koristi jedan oblik feromona za označavanje traga prema izvoru hrane, ali se za izračun najkraćeg puta natrag prema leglu koristi kompas. Panait i Luke [93, 94] su predstavili algoritam koji pruža dobre performanse u različitim zadacima

skupljanja resursa u okruženju sa preprekama. Algoritam koristi više različitih feromona te na taj način eliminira eksplicitni uvjet da mravi u svakom trenutku moraju znati put natrag prema leglu. Umjesto toga, mravi koriste informacije iz feromona da bi našli put prema hrani i natrag do legla. Autori u radu spominju efekt penjanja na brdo (*eng. hill-climbing*) koji vodi do pravocrtnih (i lokalno optimalnih) putanja te pokazuju unaprijed zadani, ali i naučeno ponašanje koje se koristi za izgradnju tih putanja.

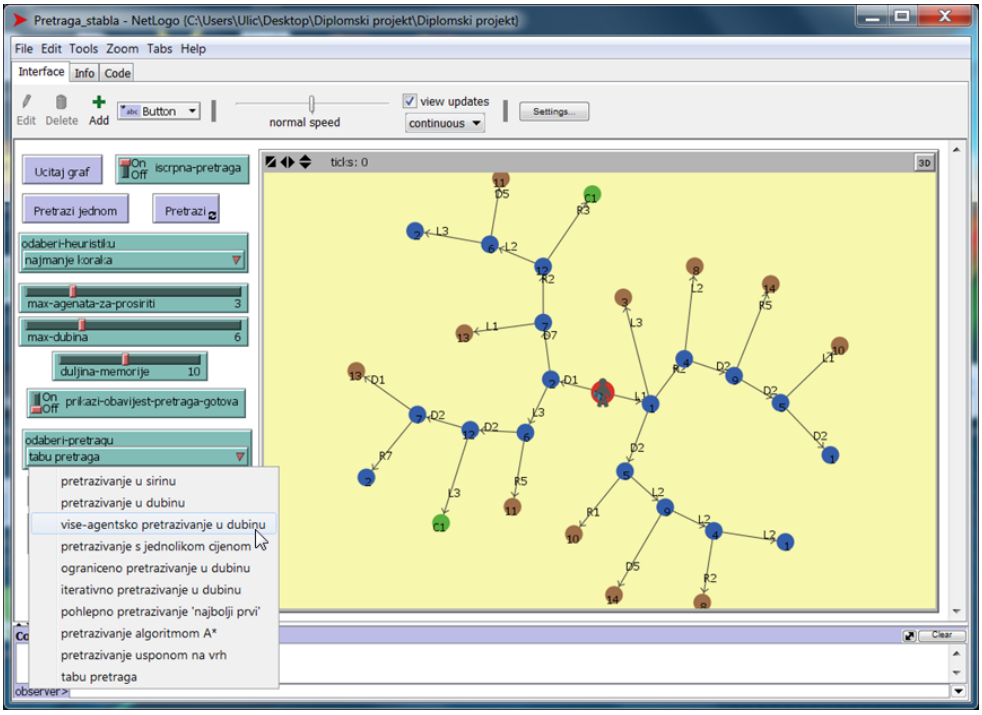
Werger i Matarić [95] predstavljaju drugačiji pristup indirektnoj komunikaciji među agentima. Agenti koriste položaje tijela da bi međusobno prenijeli neko značenje. Autori predstavljaju primjer sustava za skupljanje gdje određeni broj robota ima zadatak pokupiti neke objekte i ostaviti ih na unaprijed zadanoj lokaciji. Roboti koriste svoja tijela kako bi označili put drugim robotima. U određenom smislu bi se moglo reći da tijela robota služe kao trag feromona za druge robote. Quinn navodi [96] da je to također oblik komunikacije i poziva se na definiciju od strane Wilsona koja navodi da „komunikacija sama po sebi nije niti signal, niti odgovor na taj signal, već odnos između to dvoje“ [97]. Ova definicija sugerira da za komunikaciju nije nužan uvjet imati zajednički pridijeljeni kanal. U svom istraživanju, Quinn ispituje evoluciju strategija u sustavu sa dva kolaborativna agenta te navodi da su se agenti uspješno koordinirali tako što su međusobnom komunikacijom kroz niz pokreta uspostavili uloge vođe i sljedbenika.

# 4. Rješavanje problema pretragom prostora stanja

Rješavanje problema u umjetnoj inteligenciji se može predstaviti kao sistematično pretraživanje mogućih ishoda u svrhu pronalaska nekog predefiniranog cilja ili rješenja. U umjetnoj inteligenciji rješavanje problema putem algoritama za pretraživanje (eng. *search algorithm*) je vrlo česta tehnika [98]. U računalnoj znanosti algoritam za pretraživanje u širem kontekstu je algoritam kojim se izvršava vrednovanje skupa svih mogućih rješenja i na taj način se dobiva željeno rješenje. Skup svih mogućih rješenja nekog problema se naziva prostor pretraživanja (eng. *search space*) [99].

Cilj svakog algoritma pretrage je doći iz početnog stanja *(*eng. *initial state*) do ciljnog stanja (eng. *goal state*), koje predstavlja rješenje problema. Promjene stanja su uzrokovane slijedom događaja. Skup stanja između početnog i ciljnog stanja nije uvijek jednoznačno određen, već postoji više različitih puteva od početnog do ciljnog stanja, pri čemu se ti putevi često razlikuju po učinkovitosti ili broju potrebnih akcija. Da bi se uspješno implementirao neki sustav za rješavanje određenog problema potrebno je u njega ugraditi algoritme koji su sposobni provesti navedenu pretragu. Problem kod izrade takvih sustava je nepostojanje formalnih postupaka koji bi na temelju početnog i krajnjeg stanja sustava određivali skup stanja na putu između njih. Jedan on načina rješavanja tog problema je korištenje metode pretraživanja prostora stanja (eng. *state-space search*) [100]. Takva metoda je vrlo jednostavna ali istovremeno i vremenski zahtjevna jer je potrebno istražiti sva moguća stanja nekog sustava. Sami postupak se provodi na način da se slijedno prolazi kroz sva stanja koja može sustav poprimiti te se radi usporedba trenutnog stanja sa ciljnim stanjem da bi se utvrdilo da li je postupak došao do kraja. Učinkovitost pretraživanja u pogledu brzine izvršavanja ovisi o načinu na koji se vrši pretraživanje stanja tj. koja se strategija koristi za odabir sljedećeg stanja koje će se usporediti sa ciljnim. Jedna od osnovnih strategija je nasumični odabir sljedećeg stanja [101]. Sustavi iz jednog stanja u većini slučajeva mogu prijeći samo u jedan manji podskup ukupnog broja stanja. Zbog toga je skup stanja pogodno organizirati u strukturu grafa ili stabla te na taj način pretraživanje prostora stanja svesti na problem pronalaska puta kroz stablo ili graf. U takvim slučajevima, čvorovi grafa predstavljaju stanje sustava. Dva čvora su povezana granom u grafu samo u slučaju da je moguć prijelaz između ta dva stanja. Svaki pojedini čvor u grafu na taj način ima vezu prema svome čvoru prethodniku te se nakon pronalaska konačnog (ciljnog) stanja lako može odrediti put od početnog do završnog čvora. Put se gradi tako se krene od ciljnog čvora, te se sljedeći

veze na prethodnike, dođe do početnog čvora grafa [102]. Slika 5 daje grafički prikaz prostora stanja jednog problema. Konkretno, radi se o problemu pronalaska puta kroz labirint, gdje su svi mogući koraci (prijelaz sa jednog polja na drugo) predstavljeni putem stabla. Navedeni prikaz je realiziran u NetLogo okruženju.




*Slika 5. Prikaz pretrage stabla prostora stanja unutar NetLogo simulacije*

Problem, općenito, može imati i složeniju strukturu. Kod složenijih sustava postoji cijeli niz različitih događaja od kojih niti jedan ne garantira da će nas baš taj dovesti do rješenja. Opći pristup rješavanja takvog procesa jest generirati i ispitati (eng. *generate and test*) [103], tj. primijeniti neku akciju kako bi proizveli novo stanje, a zatim ispitati je li to stanje ciljno stanje i ako nije ponoviti proceduru. Sama ta procedura generiranja i ispitivanja stanja predstavlja proces pretraživanja. Vremensko trajanje pretraživanja može biti dugo ili kratko, ovisno o složenosti problema te učinkovitosti same strategije tj. algoritma pretraživanja koji se koristi. Kod problema koji imaju više mogućih rješenja, potrebno je nekako odrediti koje je od navedenih rješenja bolje. Tada se uvodi pojam vrijednosti (cijene) puta (eng. *path cost*) koja predstavlja zbroj vrijednosti pojedinih koraka putanje [104].

Za vrijeme pretraživanja, generiraju se svi moguće ishode nekog događaja, te se odabiru oni koji odgovaraju za pronalaženje navedenog cilja (rješenja), nakon čega se izvršava niz koraka počevši od početnog (inicijalnog) stanja kako bi se stiglo do cilja [105].

Pri rješavanju problema potrebno je razmišljati o prostoru stanja u smislu broja akcija koje možemo poduzeti i novim stanjima okoline nakon obavljanja tih akcija. Prilikom izvršavanja jedne ili više mogućih akcija (pri čemu svaka ima svoje troškove), okolina sustava se mijenja te otvara mogućnosti za nove akcije (dinamičko okruženje). Kao što je slučaj s mnogim vrstama rješavanja problema, neki putevi vode do slijepe ulice (nije moguće doći do ciljnog stanja) dok će drugi dovesti do željenog rješenja. Problem pretraživanja se svodi na pronalazak slijeda operacija koji prolaze od početnog do ciljnog stanja. Taj slijed operacija predstavlja rješenje zadanog problema. Način na koji izbjegavamo slijepe ulice, a zatim odabiremo najbolje dostupno rješenje je proizvod naše posebne strategije pretraživanja [106].

## 4.1 Pregled algoritama

Kao što je navedeno u prethodnom odlomku, pretraživanje možemo definirati kao postupak rješavanja određenih zadataka ili problema. Ono uključuje traženje puta od početnog do ciljnog/ciljnih stanja kroz prostor stanja unutar kojeg se zadatak rješava.

Da bi se započeo sam proces pretraživanja potrebno je izvršiti prvi korak, a to je definiranje problema. Problem definiramo pomoću četiri komponente. To su redom početno stanje, skup mogućih akcija, test cilja i cijena puta. Početno stanje definira stanje sustava na početku, prije početka rješavanja problema i korištenja nekog od algoritama pretrage. Skup svih mogućih akcija definira sve akcije koje se mogu izvršiti unutar nekog sustava. Ovisno o trenutnom stanju sustava, nije uvijek moguće izvršiti sve akcije iz skupa mogućih akcija. Tada imamo skup dozvoljenih akcija koji je podskup svih mogućih akcija. Svako stanje sustava ima svoj pripadajući skup dozvoljenih akcija koje se mogu izvršiti ako se sustav nalazi u tom stanju. Unija svih podskupa dozvoljenih akcija čini skup mogućih akcija cijelog sustava. Test cilja nam služi za ispitivanje da li je algoritam pronašao rješenje tj. da li je došao do ciljnog stanja. Test cilja se svodi na usporedbu trenutnog i ciljnog stanja. Ukoliko trenutno stanje odgovara zadanim parametrima koji su postavljeni na početku kao traženo rješenje tada se smatra da je pronađeno ciljno stanje. U tom slučaju više nije potrebno pretraživati, osim ako se radi o metodi iscrpnog pretraživanja koja pretražuje cijeli prostor stanja. Iscrpne metode pretraživanja nisu prigodne za sustave sa velikim skupom stanja zbog velike vremenske i računske složenosti pretrage takvog skupa stanja. Zadnji parametar kod definiranja problema je cijena puta. Cijena puta je predstavljena funkcijom koja definira numeričku vrijednost puta tj. cijenu prelaska iz jednog stanja u drugo. Postoji više kriterija za usporedbu metoda pretraživanja [107]. Najčešće korišteni kriteriji za usporedbu algoritama su:

1. Potpunost – definira da li određeni algoritam uvijek pronalazi rješenje zadanog problema. Ukoliko se ne radi o iscrpnoj pretrazi. moguće je da neki algoritmi ne pronađu svaki puta rješenje (naravno, pod pretpostavkom da se ciljno stanje nalazi unutar skupa svih stanja).
2. Vremenska kompleksnost – brzina izvođenja algoritma, odnosno koji je ukupan broj čvorova koji se trebaju otvoriti prije dolaska do rješenja (broj koraka algoritma).
3. Prostorna kompleksnost – memorija, odnosno koji je maksimalan broj čvorova koji se moraju memorirati u pojedinom stupnju rješavanja zadatka
4. Optimalnost – da li je rješenje koje smo dobili optimalno, u odnosu na tražene parametre.

Postoji nekoliko različitih podjela algoritama pretraživanja. Jedna od najčešćih je podjela algoritama na slijepe i usmjerene strategije pretraživanja [4]. Kod slijepih strategija pretraživanja dostupne informacije su početno stanje, dozvoljene operacije nad stanjima i test na rješenje problema. Ovim pretraživanjima napreduje se sistematski kroz prostor i pri tome se pretražuje čvorove nekim prethodno definiranim redoslijedom ili slučajnim odabirom. Ovakve strategije pretraživanja na početku ne znaju gdje se nalazi ciljno stanje. Suprotnost tome su algoritmi koji pripadaju usmjerenom pretraživanju. U ovoj vrsti pretraživanja na raspolaganju nam se nalazi više informacija koje nam služe za odabir sljedećeg stanja prilikom pretraživanja. Cijena prijelaza iz jednog stanja u drugo je definirana te je poznata udaljenost do ciljnog stanja. Usmjereno pretraživanje možemo podijeliti na postupke heurističkog i optimalnog pretraživanja. Optimalna pretraživanja uvijek pronalaze najbolji (optimalni) put do ciljnog stanja, dok heuristički algoritmi ne garantiraju optimalno rješenje (iako ga mogu pronaći) već njegovu aproksimaciju unutar zadanih parametara.

Postoji veliki broj različitih algoritama pretraživanja te kako tema ovog rada nije ograničena samo na pregled algoritama pretrage, opisati će se samo nekoliko najčešće korištenih algoritama iz obje navedene kategorije (slijepi i usmjereni algoritmi pretraživanja).

## 4.2 Strategije slijepog pretraživanja

Strategije slijepog pretraživanja ne uzimaju u obzir specifičnu prirodu problema. Na taj način "slijepi" algoritmi pretraživanja mogu se poopćiti, odnosno na isti način se mogu primijeniti na široko područje problema. Međutim većina prostora pretraživanja je vrlo velika, te ovakvi

algoritmi nisu efikasni osim za male primjere (većinom za edukacijske svrhe). Slijepo pretraživanje napreduje sistematski kroz prostor pretraživanja po nekom prethodno definiranom redoslijedu ili slučajnim odabirom redoslijeda evaluacije [99].

### 4.2.1 Pretraživanje u širinu

Pretraživanje u širinu (*eng. breadth-first search*) je vrsta pretraživanja u kojoj se kreće od korijena grafa. Vrši se proširivanje korijena na iduću razinu,, potom ispitivanje i proširivanje sve njegove djece i tako sve dok se ne pronađe cilj. To možemo definirati tako da se prvo ispituju i proširuju svi čvorovi na *n* dubini, pa tek nakon toga prelazimo na *n+1* dubinu.

Algoritam pretrage po širini je sljedeći:

a) Formirati listu čvorova koja na početku sadrži samo korijen
b) Dok god lista nije prazna provjeravati da li je prvi element ciljni čvor. Ukoliko je, vratiti uspješno rješenje i prekinuti pretraživanje.
c) Inače ukloniti prvi element iz reda a njegovu djecu dodati na kraj reda

Rješenje dobiveno algoritmom pretrage je potpuno jer će algoritam uvijek naći rješenje. Dobiveno rješenje je također i optimalno.

### 4.2.2 Pretraživanje u dubinu

Pretraživanje u dubinu (*eng. depth-first search*) je vrsta pretrage u kojoj se u svakom trenutku proširuje otvoreni čvor koji se nalazi na najvećoj dubini. Počinje se od korijena koji se ispituje, te u slučaju da on nije ciljni čvor vrši se proširivanje. U većini slučajeva je sljedeći na redu za ispitivanje krajnje lijevi čvor koji se proširuje ako on nije ciljni čvor. Postupak se ponavlja sve dok se ne dođe do cilja ili lista. Ako smo došli do lista, vraćamo se unatrag do prvog plićeg čvora koji ima neispitane djece te se nastavlja ispitivanje i proširivanje po istom postupku.

Algoritam pretrage u dubinu je sljedeći:

a) Pohraniti početni čvor u red
b) Ako je prvi element reda ciljni čvor onda vratiti uspješno rješenje i stati
c) Inače ukloniti prvi element, a njegovu djecu dodati na početak reda

Rješenje je potpuno, ali ova vrsta algoritma ne garantira optimalno rješenje. Kako algoritam pretražuje jednu po jednu granu stabla u dubinu. moguće je da stablo sadrži neku granu u kojoj je ciljni čvor na manjoj dubini od čvora koji je prvi pronađen. Da bi ovaj algoritam garantirao optimalno rješenje potrebno je izvršiti iscrpnu pretragu stabla, što je ovisno o veličini stabla, zahtijevan proces, a u nekim slučajevima i nemoguć.

### 4.2.3 Višeagentsko pretraživanje u dubinu

Višeagentsko pretraživanje u dubinu (eng. multi agent depth-first search) je isto kao i pretraživanje u dubinu, samo se u jednom koraku ne pretražuje samo jedan čvor nego zadani broj čvorova.

Pseudokod algoritma:

a) Generiraj listu agenata.
b) Ubaci sve agente u listu agenata i sortiraj ih prema dubini. Prednost imaju agenti koji se nalaze dublje u stablu pretrage.
    1. Ako nema agenata u listi, pretraživanje nije uspjelo.
    2. Ako je neki agent na ciljnom čvoru, pretraga je uspjela.
c) Odaberi prvih *n* agenta i kloniraj nove agente na sve čvorove sljedbenike i ukloni trenutne agente.
d) Vrati se na korak b).

## 4.3 Strategije usmjerenog pretraživanja

Za razliku od slijepog pretraživanja, usmjereno pretraživanje koristi informacije o samom problemu za kojeg se traži rješenje u svrhu bržeg pronalaska istog - što je više podataka dostupno postupku pretraživanja to je pretraživanje efikasnije. Takve informacije obično se nazivaju heurističke i temelje se na iskustvenim pravilima i tehnikama prosuđivanja koje mogu pomoći, ali nužno ne osiguravaju pronalaženje rješenja. Heurističke informacije mogu se oblikovati u heurističku evaluacijsku funkciju koja zavisi od pojedinog čvora i od cilja koji se traži. Većina algoritama za usmjereno pretraživanje namijenjena je za strukture stabla, npr. algoritam "najboljeg prvog" (eng. *best-first search*) koji zadržava procjene vrijednosti heurističkih funkcija svih prethodno generiranih čvorova i izabire najbolji za nastavak potrage [99].

### 4.3.1 Tabu pretraživanje

Tabu pretraživanje kombinira lokalno pretraživanje (vrsta pretraživanja kojom se nastoji naći najbolje rješenje smanjujući prostor pretraživanja) i kratkotrajne memorije (za što nam služi tabu lista). U kratkotrajnu memoriju pamte se rješenja zadnjih nekoliko koraka, te se ta rješenja ne mogu pojaviti kao rješenje sljedećeg koraka. Važno je pravilno odrediti veličinu memorije. Ako je memorija premala dolazi do cikličkog ponavljanja rješenja, a ako je prevelika dolazi do usporenja programa. Memorija radi na principu FIFO (*eng. first in, first out*), pa se određivanjem novog rješenja i njegovim spremanjem izbacuje rješenje koje je prvo ušlo u memoriju.

Algoritam tabu pretrage je sljedeći:

a) Sortirati agente po prijeđenom putu
b) Provjeriti da li se došlo do ciljnog stanja
c) Ukoliko nismo došli do ciljnog stanja pretraga nije gotova i provjeravamo da li se sljedeća lokacija nalazi u memoriji pretraživača
d) Ako su svi uvjeti ispunjeni pretraživač se pomiče i lokacija se unosi u memoriju
e) Ponavljati postupak dok ne dođemo do ciljnog stanja

Sva potencijalna rješenja označavaju se kao tabu (zabranjeno) kako se više ne bi vraćali na njega. Tabu rješenja nalaze se u memoriji. Algoritam će uvijek pronaći rješenje ali nije garantiran pronalazak optimalnog rješenja.

### 4.3.2 A* pretraživanje

A* (A-star) pretraživanje je vrsta pretraživanja kojom se izbjegavaju skupi putovi. Pretraga će se proširiti na čvor ne samo na temelju njegove udaljenosti od cilja, već i na temelju cijene puta. Cijena puta se dobiva sljedećom formulom:

$$f(n) = g(n) \; i \; h(n),$$

gdje g(n) predstavlja udaljenost od početnog čvora do n-tog (trenutnog), a h(n) udaljenost n-tog čvora do ciljnog čvora. Pomoću toga dobivamo cijenu puta. Zbog svoje implementacije, algoritam nudi potpuno i optimalno rješenje.

### 4.3.3 Algoritam valne fronte

Algoritam valne fronte (*eng. wavefront algorithm*) je jedan od osnovnih, ali i moćnijih algoritama. Može se podijeliti u tri glavna koraka. To su stvaranje diskretne mape prostora, ispunjavanje matrice i pomicanje agenta do cilja. Prvi korak je stvaranje diskretne mape u kojoj se agent kreće. Na toj mapi označujemo sav prazan prostor, prepreke koje se nalaze u prostoru, početnu i ciljnu poziciju. Svakom području, ovisno o vrsti, se dodjeljuje numerička vrijednost. Prepreke imaju visoku numeričku vrijednost, dok se na početku poljima koja predstavljaju prazan prostor dodjeljuje vrijednost 0.

U drugom koraku moramo ispuniti matricu kako bi imali put od agenta do cilja. Matrica se ispunjava tako da počnemo od cilja kojeg označimo sa vrijednosti 1. Zatim se označavaju sva susjedna polja, a njihova numerička vrijednost se uvećava za 1 u odnosu na ciljno polje (susjedi poprimaju vrijednost 2). Koja su mu susjedna polja ovisi o tipu susjedstva u okruženju. Nakon toga se ponavlja postupak označavanja polja. Sva neoznačena polja koja su susjedi od polja označenih u prethodnom koraku dobivaju numeričku vrijednost uvećanu za 1. Taj postupak ponavljamo sve dok ne dođemo do početne pozicije na kojoj se nalazi agent ili dok se ne označe sva polja unutar okruženja. Kada je cijela matrica ispunjena put se odabire tako da se krene od polja na kojem se nalazi agent te se u svakom koraku odabere polje koje ima numeričku vrijednost za 1 manju od trenutnog polja, sve dok se ne dođe do cilja.

Algoritam valne uvijek daje potpuno i optimalno rješenje.

# 5. Učenje inteligentnog agenta

## 5.1 Vrste učenja

Učenje je samo po sebi vrlo širok pojam te ga je vrlo teško precizno opisati, posebno ako uzmemo u obzir da sa kao pojam pojavljuje u mnogim različitim disciplinama [108]. Jedan od načina na koji možemo definirati učenje jest da je učenje modifikacija ponašanja koje je rezultat stečenog iskustva. Također iz perspektive umjetne inteligencije imamo drugu definiciju koju su postavili Mendel i McClaren u kontekstu neuronskih mreža i koja glasi: *"Učenje je proces kojim se slobodni parametri neuronske mreže prilagođavaju kroz stimulacijski proces preko okoline koju mreža sadrži. Vrsta učenja se odlučuje preko načina promjene parametra." [109]*. Kako se u ovom radu učenje gleda kroz područje umjetne inteligencije tako će se uz kratki prikaz osnovnih tipova učenja dodatni naglasak staviti na učenje putem umjetnih neuronskih mreža.

Općenito govoreći možemo reći da sva živa bića i strojevi prikazuju neki tip ponašanja. Svi na neki način reagiraju na određeni podražaj koji dobiju iz okruženja u kojem se nalaze. Neka živa bića i strojevi tokom vremena mijenjanju način na koji se ponašaju. Jednaki podražaj može uzrokovati reakciju različitu od prethodnog puta kada se javio taj podražaj. U području umjetne inteligencije za takve agente kažemo da imaju sposobnost učenja. Taj se proces naziva strojno učenje i postoji posebno znanstveno područje koje se bavi proučavanjem algoritama strojnog učenja [110]. Algoritmi strojnog učenja predstavljaju na koji način promjene u ponašanju učenika ovise o primljenim podražajima te o povratnim informacijama dobivenim iz okruženja.

Algoritme učenja možemo podijeliti u tri kategorije, ovisno o vrsti povratne informacije koju subjekt učenja dobiva natrag. Te tri kategorije su nadzirano učenje, učenje bez nadzora i poduprto učenje [111]. Osim toga, potrebno je spomenuti da se sve tri navedene metode učenja mogu realizirati pomoću neuronskih mreža [112-114].

## 5.2 Nadzirano učenje

Kod nadziranog učenja (*eng. supervised learning)*, subjekt učenja (učenik) za svaki ulazni podatak dobiva željeni cilj tj., dobiva informaciju koja bi mu trebala biti reakcija za taj ulaz. Tada učenik uspoređuje stvarnu reakciju sa onom željenom te podešava svoju unutrašnju memoriju na način da poveća vjerojatnost odabira prikladne reakcije kada idući put dobije jednake podatke na ulazu.

## 5.3 Učenje bez nadzora

Učenje bez nadzora (*eng. unsupervised learning)* je potpuno suprotno od prethodno navedene vrste učenja. Kod ovakvog načina učenja učenik iz okoline ne dobiva nikakve povratne informacije nego sam mora obaviti proces učenja. To radi na način da ulazne podatke predstavi na učinkovitiji način, kao skupine ili kategorije ili koristeći reducirani skup dimenzija. Učenje bez nadzora se temelji na sličnostima i razlikama među ulaznim uzorcima. Ovakvo učenje ne rezultira toliko očitim promjenama u ponašanju već su njegovi izlazi zapravo unutarnji prikaz znanja koji se tada mogu koristiti u drugim sustavima koji utječu na ponašanje. Učenje bez nadzora ima veliku ulogu u perceptualnim sustavima. Na primjer, informacije dobivene iz očiju čovjeka su inicijalno presložene za čovjekov živčani sustav te on smanjuje količinu informacija izdvajanjem nekih pravilnosti kao što su tendencije da rubovi budu neprekinuti i da određene regije imaju konstantnu boju.

## 5.4 Poduprto učenje

Treći pristup učenju je poduprto učenje (*eng. reinforced learning*) koje možemo smatrati kao nekom sredinom između dva prethodno navedena tipa učenja. Prilikom poduprtog učenja učenik dobiva povratnu informaciju koja mu govori koliko je prikladan njegova reakcija na dobiveni podražaj. Za ispravne reakcije, ovaj tip učenja se podudara sa metodom nadziranog učenja, učenik dobiva povratnu informaciju da je reakcija bila prikladna. Unatoč tome, ova dva načina učenja se značajno razlikuju kod situacija kod kojih je reakcija na podražaj neprimjerena. U takvom slučaju kod nadziranog učenja učenik dobije informaciju koji bi bio prikladni odgovor dok kod poduprtog učenja učeniku samo daje informaciju da je ta reakcija (dakle takvo ponašanje) neprimjerena za navedeni podražaj te u nekim slučajevima koliko je neprimjerena bila ali mu eksplicitno ne navodi koja bi bila ispravna reakcija. U prirodi je poduprto učenje puno češći slučaj od nadziranog učenja jer nije u svakom trenutku dostupan učitelj koji može reći što bi bila ispravna reakcija, a i kada postoji ne možemo biti sigurni da će učenik pravilno interpretirati dobivenu povratnu informaciju.

Postoji nekoliko različitih definicija poduprtog učenja. Jednu od njih navode Hertz i dr. te definiraju poduprto učenje kao proces u kojem sustav poboljšava svoje performanse na određenom zadatku bez dodatnog programiranja [115].

Michalsky i dr. [116] definiraju poduprto učenje kao određeni skup procesa pod koji spadaju prikupljanje novog deklarativnog znanja, razvoj i usavršavanje motornih i spoznajnih sposobnosti kroz praksu, strukturiranje postojećeg znanja i otkrivanje novih činjenica i teorija promatranjem i aktivnim eksperimentiranjem.
Većina ovih definicija razlikuje dvije odvojene faze u procesu učenja:

- *prikupljanje znanja* – ova faza predstavlja učenje nove, simboličke informacije, na način da se ta informacija može efektivno primijeniti.
- *uvježbavanje* – ova faza obuhvaća akcije koje podrazumijevaju poboljšavanje nekog stečenog znanja, mentalne ili motorne koordinacije, kroz praktično ponavljanje i korekciju odstupanja od željenog ponašanja.

### 5.4.1 Neuronske mreže

Unaprijed zadani skup dobro definiranih pravila koja predstavljaju rješenje nekog problema naziva se *algoritam učenja* [117]. Kod konstrukcije umjetnih neuronskih mreža, ne postoji jedinstveni algoritam učenja koji bi se mogao univerzalno primjenjivati. Umjesto toga, na raspolaganju nam je velika količina algoritama učenja koje možemo primijeniti na umjetne neuronske mreže, svaka sa svojim prednostima. Algoritmi učenja se zapravo razlikuju jedni od drugih po načinu na koji definiraju funkciju prijelaza i kako podešavaju težinske koeficijente između čvorova. Također možemo razlikovati i načine na koji su neuronske mreže povezane sa svojom okolinom. U tom kontekstu govorimo o paradigmi učenja koja se odnosi na model okoline u kojoj neuronska mreža djeluje.

Proces učenja korištenjem neuronskih mreža se sastoji od procesa modifikacije specifičnih težina neuronske mreže primjenom algoritma učenja nad skupinom obrazaca. Unatoč tome, ne postoji univerzalni algoritam učenja kojeg možemo primijeniti za dizajn neuronskih mreža. U prethodnom poglavlju smo kod metoda učenja imali podjelu na tri osnovna tipa (nadzirano, nenadzirano i poduprto učenje). Slična situacija je i kada učenje stavimo u kontekst neuronskih mreža, gdje imamo podjelu na dvije osnovne kategorije:

- umjetne neuronske mreže s nadziranim učenjem te
- umjetne neuronske mreže s nenadziranim učenjem.

Kao glavnu razliku između ovih navedenih načina učenja možemo navesti postojanje odnosno nepostojanje vanjskog sudionika (agenta, učitelja) koji kontrolira sam proces učenja neuronske mreže.

Također možemo navesti još jednu podjelu umjetnih neuronskih mreža koja se često koristi, a koja dijeli neuronske mreže na mreže koje imaju sposobnost učenja prilikom svog normalnog rada (*on-line)* te na mreže kod kojih nije moguće izvršavati proces učenja za vrijeme rada mreže već zahtijevaju isključivanje mreže (*off-line*).

Kod procesa učenja koje zahtijeva isključivanje mreže (*off-line*) imamo strogo odvojene dvije faze, fazu učenja te fazu djelovanja (operacije). Prilikom prve faze koristimo skupinu podataka koja bi trebala sadržavati primjer znanja kojeg pokušavamo prenijeti našoj neuronskoj mreži. Nakon *off-line* faze učenja veze između čvorova imaju određene težinske vrijednosti koje nakon što završi faza učenja ostaju nepromijenjene tokom faze djelovanja. Mreže koje koriste *on-line* učenje (bez isključivanja) nemaju definiranu razliku između faza učenja i djelovanja te zbog toga težinske vrijednosti između čvorova neprestano variraju tokom izvođenja tj. dinamički im se mijenja vrijednost kada sustav tokom izvođenja primi novu informaciju [109].

Pridržavajući se definicija procesa učenja navedenih ranije u radu implicira se da tokom procesa dolazi do sljedećih događaja:

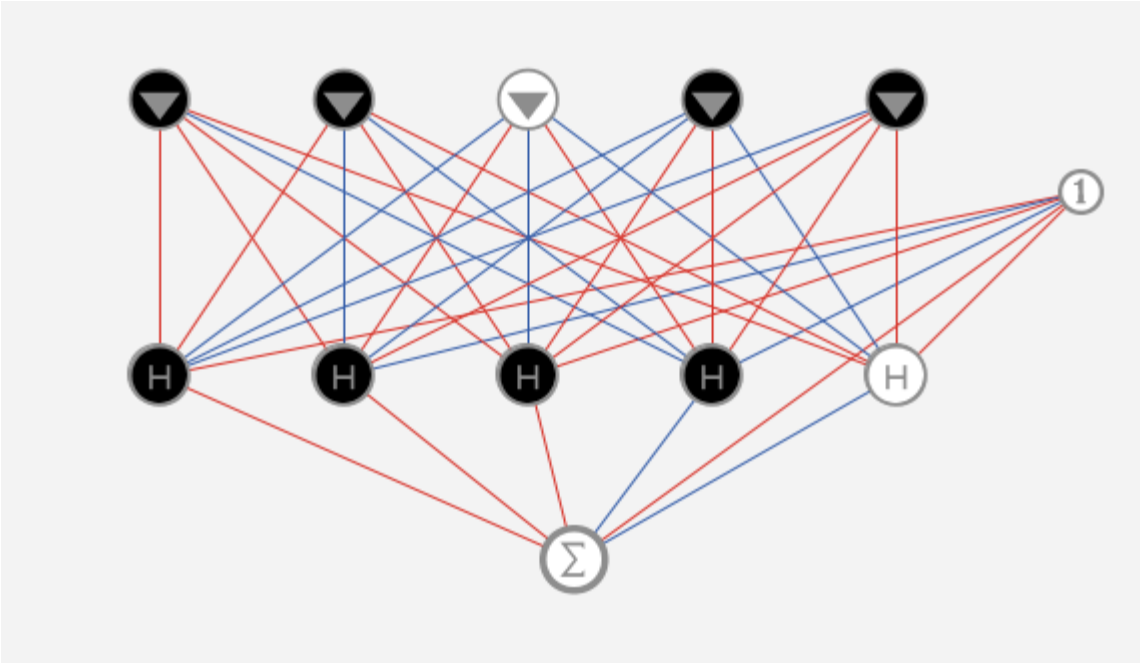
- Neuronska mreža dobiva podražaj iz okoline
- Neuronska mreža prolazi kroz promjene uzrokovane tim podražajem
- Neuronska mreža na drugi način djeluje na okolinu, zbog promjena koje su se dogodile u njenoj unutarnjoj strukturi

### 5.4.2 Višeslojna perceptronska mreža

Unatoč velikom broju različitih modela neurona, moguće ih je svrstati u dvije osnovne skupine: statičke i dinamičke modele neurona. U primjeni se najčešće koristi statički model neurona, McCulloh-Pittsov umjetni neuron ili perceptron [118]. Ovaj umjetni neuron, iako jednostavan, sa sigmoidalnom aktivacijskom funkcijom predstavlja vrlo korisnu aproksimaciju biološkog neurona. Takav model neurona ne sadrži dinamičke članove, njegov izlaz ovisi isključivo o trenutnim vrijednostima ulaznih signala i težinskim koeficijentima, stoga se ovaj neuron u literaturi najčešće naziva statičkim neuronom. Veliki broj neuronskih mreža izgrađen je od perceptrona organiziranih u tri ili više slojeva, te se mreže nazivaju višeslojne perceptronske mreže (*eng. MultiLayer Perceptron networks*, MLP) [119].

Višeslojne perceptronske neuronske mreže izgrađene su od perceptron neurona organiziranih u serijski povezane slojeve. Slojevi se označuju brojevima 0, 1, … L. Nulti sloj samo prosljeđuje vektor ulaza na ulaz prvog sloja. L-ti sloj je ujedno i izlazni sloj mreže, a slojevi

između njih nazivaju se unutarnjim ili skrivenim slojevima (eng. *Hidden layers*) jer ne prosljeđuju ni primaju vanjske signale. Svi neuroni u nekom sloju povezani su sa svim neuronima u dva susjedna sloja, preko jednosmjernih, unaprijednih veza. Druge veze nisu dopuštene, to jest ne postoje veze između neurona u istom sloju niti između neurona koji nisu u susjednim slojevima. Veze između neurona susjednih slojeva predstavljene su sinaptičkim težinskim koeficijentima koji djeluju kao pojačala signala na odgovarajućim vezama. Iznos sinaptičkih težinskih koeficijenata određuje vladanje mreže, odnosno njezinu sposobnost aproksimacije nelinearne funkcije. Izračunavanje njihovih odgovarajućih iznosa ostvaruje se algoritmima učenja. Iako nema teoretskog ograničenja na broj unutarnjih slojeva neurona, uglavnom se koriste MLP mreže s jednim ili dva unutarnja sloja, to jest dvoslojne ili troslojne MLP mreže. Teoretski je dokazano da MLP mreža s jednim unutarnjim slojem može aproksimirati proizvoljnu kontinuiranu nelinearnu funkciju [120, 121]. Slika 6 prikazuje jedan primjer višeslojne perceptronske neuralne mreže.




*Slika 6. Višeslojna perceptronska neuronska mreža*

### 5.4.3 Algoritam povratnog prostiranja izlazne pogreške

Neuronska je mreža u potpunosti određena tek kada je uz njezinu strukturu (određena brojem, tipom i organizacijom neurona u mreži) definiran i algoritam učenja. Algoritme učenja je moguće po načinu učenja mreže podijeliti na algoritme učenja temeljene na pogrešci (eng.

*Error-based learning algorithms*), algoritme učenja temeljene na izlazu mreže (eng *output-based learning algorithms*) i algoritme učenja s ojačanjem (eng. *reinforcement algorithms*).
Algoritmi učenja temeljeni na pogrešci često se nazivaju i algoritmi s "učiteljem" jer zahtijevaju vanjski referentni signal s kojim uspoređuju dobiveni odziv neuronske mreže generirajući signal pogreške. Na temelju signala pogreške algoritam mijenja sinaptičke težinske koeficijente neuronske mreže s ciljem poboljšanja njezina vladanja, to jest smanjenja pogreške. Prema tome, ovi se algoritmi mogu primijeniti samo ako je unaprijed poznato željeno vladanje neuronske mreže, to jest podaci na osnovi kojih se mreža uči moraju sadržavati vrijednosti ulazno-izlaznih signala.
Kod učenja algoritam podešava parametre mreže dok kriterij kakvoće ne poprimi minimalan iznos odnosno iznos manji od unaprijed zadanog iznosa. Dakle problem podešavanja parametara neuronske mreže svodi se na problem nelinearnog optimiranja s kriterijskom funkcijom. Kao ciljnom funkcijom. Optimiranje se provodi na temelju skupa ulazno-izlaznih podataka dobivenih eksperimentom na stvarnom procesu, odnosno funkciji koju neuronska mreža treba aproksimirati. Osnovni problem koji kod primjene postupka nelinearnog optimiranja za učenje neuronskih mreža treba razriješiti jest izračunavanje gradijenta kriterijske funkcije po parametrima mreže. Taj je problem dulje vrijeme usporavao istraživanja i primjenu neuronskih mreža, ali je uspješno razriješen primjenom algoritma povratnog prostiranja izlazne pogreške kroz mrežu (eng. *Back-Propagation algorithm*).

Prvi je BP algoritam primijenio Werbos 1974. godine [122] za izračunavanje kriterija kakvoće za MLP neuronske mreže, dok je njegova intenzivna primjena počela tek 1986. godine kada su ga objavili Rumelhart i dr.[123]. BP algoritam izračunava parcijalne derivacije kriterija kakvoće po parametrima mreže rekurzivnim postupkom koji se odvija povratno kroz mrežu od izlaznoga prema ulaznom sloju mreže. Algoritam se temelji na pretpostavci da je prostiranje derivacije pogreške kroz mrežu linearno.

### 5.5 Pohrana znanja

Prilikom definiranja modela agenta također je potrebno definirati način pohrane znanja. Prilikom odabira načina pohrane znanja agenta potrebno je uzeti u razmatranje sljedeće kriterije odabira:

### 5.5.1 Svrha modela znanja

Model znanja se koristi za stvaranje i prikaz hibridnog inteligentnog agenta. Uloga agenta da je da se može snalaziti u okruženju u kojem se nalazi te da vrši interakciju sa drugim agentima svojoj blizini. Rezultati te interakcije bi trebali biti bolje snalaženje u okruženju, usvajanje novog znanja ili prijenos znanja na drugog agenta. Osnovni cilj ovakvog modela znanja je što vjerodostojnije predstaviti način učenja i pohrane znanja kod čovjeka, sa svim prednostima i manama koje to uključuje, samo na primjeru inteligentnog agenta.

### 5.5.2 Sadržaj modela znanja

Odabrani model znanja bi se trebao koristiti za opisivanje okruženja u kojem se agenti nalaze te za prikaz osnovnih pojmova vezanih uz same agente, od osnovnih karakteristika pa do pravila ponašanja i njihovih ciljeva. Također je potrebno definirati model samog korisnika te model interakcije između agenta i korisnika te model interakcije između samih agenata, imajući u vidu da je moguća interakcija između agenata iste vrste i agenata različite vrste.

### 5.5.3 Zahtjevi modela znanja

Model znanja mora imati podršku za povezivanje sa neuronskim mrežama jer se one koriste prilikom uvježbavanja i učenja agenta. Potrebno je omogućiti što lakše spremanje i pristup naučenom znanju uz korištenje neuronskih mreža.

### 5.5.4 Organizacija znanja

Znanje mora biti organizirano modularno, korištenjem višeslojne arhitekture pri čemu svaki modul predstavlja elemente znanja na jednoj razini apstrakcije. Moduli su međusobno hijerarhijski organizirani tako da više modula koji se nalaze u istoj razini zajedno predstavljaju jedan segment znanja.

## 5.6 Predstavljanje znanja

U skladu sa zahtjevima navedenim u prethodnom dijelu rada, za pohranu i predstavljanje znanja u modelu inteligentnog agenta su odabrane semantičke mreže, točnije mreža koncepata i relacija. Prilikom odabira modela znanja u razmatranje su uzeti neki već postojeći modeli [124] ali se pokazalo da su ti modeli u većini slučajeva prilično složeni i imaju veliki broj pojmova te samim time nisu praktični za upotrebu. Također, nedovoljna fleksibilnost već

postojećih modela znanja i ontologija nam ne dozvoljava jednostavnu promjenu niti nadogradnju tih modela čime se gubi sposobnost prilagodbe agenta na različita okruženja i zadatke.

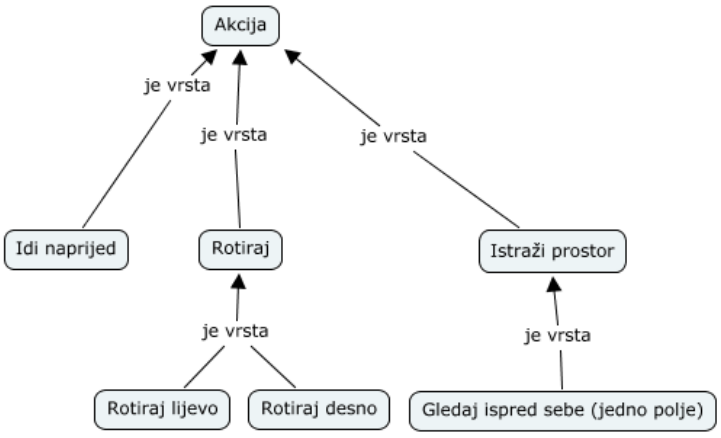
Uzevši u obzir sve naveden zahtjeve i ograničenja, predstavljena je sljedeća struktura modela znanja inteligentnog agenta koja za prikaz znanja koristi određene koncepte i relacije između tih koncepata.

### 5.6.1 Koncepti

Svaki element znanja je predstavljen pomoću koncepta. Jedan koncept može predstavljati točno određeni objekt (npr. *zid, kuća, robot*) ali isto tako i bilo kakvu apstraktnu pojavu, ideju ili akciju (npr. *pomakni se, cilj, cijena*). Ne postoji strogo definirana razlika između ova sva tipa koncepata jer su svi koncepti predstavljeni na isti način i imaju ista svojstva.

### 5.6.2 Relacije

Različiti koncepti su međusobno povezani relacijom. Relacije služe za definiranje odnosa tj. interakcije između dva određena koncepta. Na taj način se konceptima dodjeljuje konkretna primjena. Relacije između koncepata su usmjerene i jednosmjerne, čime se predstavljanje znanja svodi na usmjereni graf, gdje su koncepti predstavljeni čvorovima grafa dok relacije odgovaraju granama grafa. Slika prikazuje jedan primjer prikaza znanja korištenjem mape koncepata i relacija.




Količina znanja koju jedan agent posjeduje odgovara broju različitih koncepata koje je agent naučio, s obzirom na ukupan broj koncepata dostupnih u njegovom okruženju. Kvaliteta tog

znanja ovisi o uspješnosti prepoznavanja svakog individualnog koncepta koji je prethodno naučen te o ispravnoj primjeni prepoznatog koncepta. Samo poznavanje velikog broja koncepata je beskorisno ukoliko agent ispravno ne razlikuje te koncepte i nije ih u stanju prepoznati.

Kao što je prethodno navedeno, cilj ovakvog modela pohrane i prikaza znanje je olakšati povezivanje sa neuronskim mrežama koje služe za učenje inteligentnog agenta. Svaki agent bi trebao imati sposobnost proširiti svoje znanje tj. naučiti nove koncepte i ukomponirati ih u već postojeće znanje. Ovakav model znanja bi osim toga također trebao i olakšati razmjenu znanja između različitih agenata korištenjem procesa mapiranja znanja.

# 6. Zaključak

Računalni sustavi temeljeni na agentima su dobro poznato područje i koriste se već dugo vremena. Unatoč tome, i dalje nema striktne definicije agenta ili jedinstvene arhitekture koja bi se koristila pri dizajniranju takvog sustava. Pojam agenta je prilično fleksibilan i može uključivati različite karakteristike. Sustavi sa većim brojem agenata se nazivaju višeagentski sustavi i također imaju mnoštvo primjena u svakodnevnom životu.

Algoritmi pretrage se često koriste u višeagentskim sustavima i služe za pronalazak rješenja uz pomoć pretrage prostora stanja. Postoji mnoštvo različitih algoritama pretrage. Odabir algoritma ovisi o njegovim karakteristikama, ali i o tipu problema za koji se algoritam koristi. U ovom radu je opisano samo nekoliko najčešće korištenih algoritama. Daljnja istraživanja na tu temu bi trebala uključivati detaljniji prikaz većeg broja algoritama, njihovu implementaciju na konkretnom problemu te usporedbu dobivenih rezultata.

Mogućnost učenja je jedna od važnih karakteristika inteligentnih agenata. U ovom radu su opisane tri različita načina učenja te je stavljen naglasak na učenje i predstavljanje znanja pomoću umjetnih neuronskih mreža. Kao i kod algoritama pretrage, idući korak je implementacija sustava sa opisanim karakteristikama te njegovo testiranje na konkretnom problemu.

# 7. Literatura


1. Guerrero-Bote, V.P. and F. Moya-Anegón, *A further step forward in measuring journals' scientific prestige: The SJR2 indicator.* Journal of Informetrics, 2012. **6**(4): p. 674-688.
2. Wooldridge, M. and N.R. Jennings, *Intelligent agents: Theory and practice.* The knowledge engineering review, 1995. **10**(02): p. 115-152.
3. Wooldridge, M., *An introduction to multiagent systems*. 2009: John Wiley & Sons.
4. Russell, S. and P. Norvig, *AI a modern approach.* Learning, 2005. **2**(3): p. 4.
5. Ghosh, D., et al., *Self-healing systems—survey and synthesis.* Decision Support Systems, 2007. **42**(4): p. 2164-2185.
6. Cockburn, D. and N.R. Jennings, *ARCHON: A distributed artificial intelligence system for industrial applications.* Foundations of Distributed Artificial Intelligence, 1996: p. 319-344.
7. Jennings, N.R., *An agent-based approach for building complex software systems.* Communications of the ACM, 2001. **44**(4): p. 35-41.
8. Monostori, L., J. Váncza, and S.R. Kumara, *Agent-based systems for manufacturing.* CIRP Annals-Manufacturing Technology, 2006. **55**(2): p. 697-720.
9. Tumer, K. and A.K. Agogino, *Improving Air Traffic Management with a Learning Multiagent System.* IEEE Intelligent Systems, 2009. **24**(1): p. 18-21.
10. Tumer, K. and A. Agogino. *Distributed agent-based air traffic flow management*. in *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*. 2007. ACM.
11. Chen, J.R., S.R. Wolfe, and S.D. Wragg. *A distributed multi-agent system for collaborative information management and sharing*. in *Proceedings of the ninth international conference on Information and knowledge management*. 2000. ACM.
12. Ronglong, K. and J. Tongqiang. *E-Commerce Automated Negotiation Model Research Based on Multi-agent*. in *Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2013 5th International Conference on*. 2013. IEEE.
13. More, A., S. Vij, and D. Mukhopadhyay. *Agent Based Negotiation Using Cloud–An Approach in E-Commerce*. in *ICT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of India-Vol I*. 2014. Springer.
14. Yan, Y., Z. Maamar, and W. Shen. *Integration of workflow and agent technology for business process management*. in *Computer Supported Cooperative Work in Design, The Sixth International Conference on, 2001*. 2001. IEEE.
15. Jennings, N.R., et al., *Autonomous agents for business process management.* Applied Artificial Intelligence, 2000. **14**(2): p. 145-189.
16. Corchado, J.M., et al., *Intelligent environment for monitoring Alzheimer patients, agent technology for health care.* Decision Support Systems, 2008. **44**(2): p. 382-396.
17. Su, C.J., *Mobile multi-agent based, distributed information platform (MADIP) for wide-area e-health monitoring.* Computers in Industry, 2008. **59**(1): p. 55-68.
18. Su, C.-J. and C.-Y. Wu, *JADE implemented mobile multi-agent based, distributed information platform for pervasive health care monitoring.* Applied Soft Computing, 2011. **11**(1): p. 315-325.
19. Riedl, M., C.J. Saretto, and R.M. Young. *Managing interaction between users and agents in a multi-agent storytelling environment*. in *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*. 2003. ACM.

20. Adobbati, R., et al. *Gamebots: A 3d virtual world test-bed for multi-agent research*. in *Proceedings of the second international workshop on Infrastructure for Agents, MAS, and Scalable MAS*. 2001. Montreal, Canada.
21. Jennings, N., et al. *Applications of multi-agent systems: Prospects and promises*. in *International Workshop on Multi-Agent Systems (IWMAS). Cambridge, MA: Massachusetts Institute of Technology*. 1997.
22. Maes, P., *Artificial life meets entertainment: lifelike autonomous agents.* Communications of the ACM, 1995. **38**(11): p. 108-114.
23. Castelfranchi, C., *Guarantees for autonomy in cognitive agent architecture*, in *Intelligent agents*. 1995, Springer. p. 56-70.
24. Genesereth, M.R. and S.P. Ketchpel, *Software agents.* Commun. ACM, 1994. **37**(7): p. 48-53.
25. Agha, G., *Concurrent object-oriented programming.* Communications of the ACM, 1990. **33**(9): p. 125-141.
26. Agha, G., P. Wegner, and A. Yonezawa, *Research directions in concurrent object-oriented programming*. 1993: Mit Press.
27. Etzioni, O. and D. Weld, *A softbot-based interface to the internet.* Communications of the ACM, 1994. **37**(7): p. 72-76.
28. Shoham, Y., *Agent-oriented programming.* Artificial intelligence, 1993. **60**(1): p. 51-92.
29. Bates, J., A.B. Loyall, and W.S. Reilly, *An architecture for action, emotion, and social behavior*. 1994: Springer.
30. Maes, P., *Agents that reduce work and information overload.* Communications of the ACM, 1994. **37**(7): p. 30-40.
31. Gollmann, D., *Veracity, plausibility, and reputation*, in *Information Security Theory and Practice. Security, Privacy and Trust in Computing Systems and Ambient Intelligent Ecosystems*. 2012, Springer. p. 20-28.
32. Carley, K., D. Park, and M. Prietula, *Agent honesty, cooperation and benevolence in an artificial organization.* Institute for Software Research, 1993: p. 30.
33. Goodwin, R., *Formalizing properties of agents.* Journal of Logic and Computation, 1995. **5**(6): p. 763-781.
34. Maes, P., *The agent network architecture (ANA).* ACM SIGART Bulletin, 1991. **2**(4): p. 115-120.
35. Kaelbling, L.P., *A situated-automata approach to the design of embedded agents.* ACM SIGART Bulletin, 1991. **2**(4): p. 85-88.
36. Brooks, R.A., *A robust layered control system for a mobile robot.* Robotics and Automation, IEEE Journal of, 1986. **2**(1): p. 14-23.
37. Gorton, T. and B. Mikhak. *A tangible architecture for creating modular, subsumption-based robot control systems*. in *CHI'04 Extended Abstracts on Human Factors in Computing Systems*. 2004. ACM.
38. Rojas, R., et al., *FU-Fighters 2001 (Global Vision)*, in *RoboCup 2001: Robot Soccer World Cup V*. 2002, Springer. p. 571-574.
39. Low, K.H., W.K. Leow, and M.H. Ang Jr. *A hybrid mobile robot architecture with integrated planning and control*. in *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1*. 2002. ACM.
40. Nicolescu, M.N. and M.J. Matarić. *A hierarchical architecture for behavior-based robots.* in *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1*. 2002. ACM.

41. Iglesias, C.A., M. Garijo, and J.C. González, *A survey of agent-oriented methodologies,* in *Intelligent Agents V: Agents Theories, Architectures, and Languages*. 1999, Springer. p. 317-330.
42. Decugis, V. and J. Ferber. *Action selection in an autonomous agent with a hierarchical distributed reactive planning architecture*. in *Proceedings of the second international conference on Autonomous agents*. 1998. ACM.
43. Moya, L.J. and A. Tolk. *Towards a taxonomy of agents and multi-agent systems*. in *Proceedings of the 2007 spring simulation multiconference-Volume 2*. 2007. Society for Computer Simulation International.
44. Wagner, T., U. Visser, and O. Herzog, *Egocentric qualitative spatial knowledge representation for physical robots.* Robotics and Autonomous Systems, 2004. **49**(1): p. 25-42.
45. Chronis, G. and M. Skubic. *Robot navigation using qualitative landmark states from sketched route maps*. in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*. 2004. IEEE.
46. Pederson, T., L.-E. Janlert, and D. Surie. *Towards a model for egocentric interaction with physical and virtual objects*. in *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*. 2010. ACM.
47. Bagnall, A.J. and Z.V. Zatuchna, *On the classification of maze problems,* in *Foundations of learning classifier systems*. 2005, Springer Berlin Heidelberg: New York. p. 307-316.
48. Groover, M.P., *Automation, production systems, and computer-integrated manufacturing*. 2007: Prentice Hall Press.
49. Papert, S., *Mindstorms: Children, computers, and powerful ideas*. 1980: Basic Books, Inc.
50. Resnick, M., S. Ocko, and S. Papert, *LEGO, Logo, and design.* Children's Environments Quarterly, 1988: p. 14-18.
51. Zaharija, G., S. Mladenović, and I. Boljat, *Introducing basic Programming Concepts to Elementary School Children.* Procedia-Social and Behavioral Sciences, 2013. **106**: p. 1576-1584.
52. Zaharija, G., S. Mladenović, and I. Boljat. *Use of robots and tangible programming for informal computer science introduction*. in *International Conference on New Horizons, INTE 2014*. 2014.
53. Cliburn, D.C. *Experiences with the LEGO Mindstorms throughout the undergraduate computer science curriculum*. in *Frontiers in Education Conference, 36th Annual*. 2006. IEEE.
54. Khalaf, K., et al., *Innovation in teaching freshman engineering design: an integrated approach.* EDULEARN10 Proceedings, 2010: p. 709-720.
55. Ewert, D., D. Schilberg, and S. Jeschke, *Problem based learning of object-oriented Programming with LEGO Mindstorms and leJOS,* in *Automation, Communication and Cybernetics in Science and Engineering 2011/2012*. 2013, Springer. p. 315-323.
56. Tokuyasu, T. *Installation of Mechatronics Education Using the MindStorms for Dept. of Mechanical Engineering, ONCT*. in *Mechatronics, ICM2007 4th IEEE International Conference on*. 2007. IEEE.
57. Klassner, F. *A case study of LEGO Mindstorms'™ suitability for artificial intelligence and robotics courses at the college level*. in *ACM SIGCSE Bulletin*. 2002. ACM.
58. Hixon, R., *Teaching software engineering principles using Robolab and Lego Mindstorms.* International Journal of Engineering Education, 2007. **23**(5): p. 868-873.
59. Alimisis, D., et al. *Robotics & constructivism in education: The TERECoP project*. in *EuroLogo*. 2007.

60. Fosnot, C.T., *Constructivism: Theory, perspectives, and practice*. 2013: Teachers College Press.
61. Fagin, B. and L. Merkle. *Measuring the effectiveness of robots in teaching computer science*. in *ACM SIGCSE Bulletin*. 2003. ACM.
62. Wavish, P., *Exploiting emergent behavior in multi-agent systems.* ACM SIGOIS Bulletin, 1992. **13**(3): p. 16.
63. Wilensky, U., *NetLogo: Center for connected learning and computer-based modeling.* Northwestern University, 1999.
64. Erin, B., R. Abiyev, and D. Ibrahim, *Teaching robot navigation in the presence of obstacles using a computer simulation program.* Procedia-Social and Behavioral Sciences, 2010. **2**(2): p. 565-571.
65. Conkur, E.S., *RoboKol: A computer program for path planning for redundant and mobile robots.* Computer Applications in Engineering Education, 2006. **14**(3): p. 198-210.
66. Chu, K.-H., R. Goldman, and E. Sklar. *Roboxap: an agent-based educational robotics simulator*. in *Agent-based Systems for Human Learning Workshop at AAMAS-2005*. 2005.
67. Yadin, A. *A virtual learning environment for enhancing students' understandability*. in *Conference proceedings of" eLearning and Software for Education"(eLSE)*. 2013.
68. Stone, P. and M. Veloso, *Multiagent systems: A survey from a machine learning perspective.* Autonomous Robots, 2000. **8**(3): p. 345-383.
69. Durfee, E.H., V.R. Lesser, and D.D. Corkill, *Coherent cooperation among communicating problem solvers.* Computers, IEEE Transactions on, 1987. **100**(11): p. 1275-1291.
70. Tan, M. *Multi-agent reinforcement learning: Independent vs. cooperative agents*. in *Proceedings of the tenth international conference on machine learning*. 1993. Amherst, MA.
71. Berenji, H.R. and D. Vengerov. *Advantages of cooperation between reinforcement learning agents in difficult stochastic problems*. in *Fuzzy Systems, 2000. FUZZ IEEE 2000. The Ninth IEEE International Conference on*. 2000. IEEE.
72. Wagner, K., *Cooperative strategies and the evolution of communication.* Artificial Life, 2000. **6**(2): p. 149-179.
73. Yanco, H. and L.A. Stein. *An adaptive communication protocol for cooperating mobile robots*. in *Meyer, JA, HL Roitblat, and S. Wilson (1993) From Animals to Animats 2. Proceedings of the Second International Conference on Simulation of Adaptive Behavior. The MIT Press, Cambridge Ma*. 1993.
74. Jim, K.-C. and C.L. Giles, *Talking helps: Evolving communicating agents for the predator-prey pursuit problem.* artificial life, 2000. **6**(3): p. 237-254.
75. Steels, L., *Emergent adaptive lexicons.* From animals to animats, 1996. **4**: p. 562-567.
76. Steels, L. and F. Kaplan, *Collective learning and semiotic dynamics*, in *Advances in artificial life*. 1999, Springer. p. 679-688.
77. Cangelosi, A., *Evolution of communication and language using signals, symbols, and words.* Evolutionary Computation, IEEE Transactions on, 2001. **5**(2): p. 93-101.
78. De Boer, B. *Generating vowel systems in a population of agents*. in *Fourth European Conference on Arficial Life, Brighton*. 1997. Citeseer.
79. Steels, L., *Synthesising the origins of language and meaning using co-evolution, self-organisation and level formation.* Approaches to the Evolution of Language, 1998: p. 384-404.
80. Steels, L., *The puzzle of language evolution.* Kognitionswissenschaft, 2000. **8**(4): p. 143-150.

81. Williams, A.B., *Learning to share meaning in a multi-agent system.* Autonomous Agents and Multi-Agent Systems, 2004. **8**(2): p. 165-193.
82. Ackley, D.H. and M.L. Littman. *Altruism in the evolution of communication*. in *Artificial life IV*. 1994.
83. Wilson, E.O. and B. Hölldobler, *The rise of the ants: a phylogenetic and ecological explanation.* Proceedings of the National Academy of Sciences of the United States of America, 2005. **102**(21): p. 7411-7414.
84. Bonabeau, E., M. Dorigo, and G. Theraulaz, *Swarm intelligence: from natural to artificial systems*. 1999: Oxford university press.
85. Monekosso, N. and P. Remagnino, *Phe-Q: A pheromone based Q-learning*, in *AI 2001: Advances in Artificial Intelligence*. 2001, Springer. p. 345-355.
86. Monekosso, N. and P. Remagnino, *An analysis of the pheromone Q-learning algorithm*, in *Advances in Artificial Intelligence—IBERAMIA 2002*. 2002, Springer. p. 224-232.
87. Monekosso, N., P. Remagnino, and A. Szarowicz, *An improved Q-learning algorithm using synthetic pheromones*, in *From Theory to Practice in Multi-Agent Systems*. 2002, Springer. p. 197-206.
88. Sauter, J.A., et al. *Evolving adaptive pheromone path planning mechanisms*. in *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1*. 2002. ACM.
89. Sauter, J., et al., *Tuning synthetic pheromones with evolutionary computing.* Ann Arbor, 2001. **1001**: p. 48113-4001.
90. White, T., B. Pagurek, and F. Oppacher, *ASGA: Improving the ant system by integration with genetic algorithms.* Genetic Programming, 1998: p. 610-617.
91. Collins, R.J. and D. Jefferson, *Antfarm: Towards simulated evolution*. 1990: Computer Science Department, University of California.
92. Collins, R.J. and D.R. Jefferson, *An artificial neural network representation for artificial organisms*, in *Parallel problem solving from nature*. 1991, Springer. p. 259-263.
93. Panait, L. and S. Luke. *Learning ant foraging behaviors*. in *Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems (ALIFE9)*. 2004.
94. Panait, L. and S. Luke. *A pheromone-based utility model for collaborative foraging*. in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*. 2004. IEEE Computer Society.
95. Werger, B.B. and M.J. Matari'c, *Exploiting embodiment in multi-robot teams.* 1999.
96. Quinn, M., *Evolving communication without dedicated communication channels*, in *Advances in Artificial Life*. 2001, Springer. p. 357-366.
97. Wilson, E.O., *Sociobiology: The new synthesis*. 2000: Harvard University Press.
98. Zaheer, K., *Artificial Intelligence Search Algorithms In Travel Planning.* Department of Computer sciences and Electronics Mälardalen University Västerås Sweden, 2006.
99. Pavković, N., D. Marjanović, and N. Bojčetić, *Programiranje i algoritmi, skripta*. 2005, Zagreb: Fakultet strojarstva i brodogradnje.
100. Anderson, J.R., *Problem solving and learning.* American Psychologist, 1993. **48**(1): p. 35.
101. Pelánek, R., et al. *Enhancing random walk state space exploration*. in *Proceedings of the 10th international workshop on Formal methods for industrial critical systems*. 2005. ACM.
102. Škvorc, D. and E. Štifanić *Metode pretraživanja prostora stanja*. 2002.

103. Mitchell, T.M., *Generalization as search.* Artificial intelligence, 1982. **18**(2): p. 203-226.
104. Heylighen, F., *Problem solving*. 1998.
105. Šuljeg, A. *Inteligentni agenti za pretraživanje Web-a*. 2008.
106. Jones, M.T., *Artificial intelligence: A systems approach*. 2008, Hingham: Infinity Science Press LLC.
107. Zhang, W., *State-space search: Algorithms, complexity, extensions, and applications*. 1999: Springer.
108. Twigg, C.A., *The Changing Definition of Learning.* Educom Review, 1994. **29**(4): p. 23-25.
109. Mendel, J.M., *A prelude to neural networks: Adaptive and learning systems*. 1994: Prentice Hall Press.
110. Ayodele, T.O., *Introduction to machine learning.* New Advances in Machine Learning. InTech, 2010.
111. Kotsiantis, S.B., I. Zaharakis, and P. Pintelas, *Supervised machine learning: A review of classification techniques*. 2007.
112. Engelbrecht, A.P., *Supervised Learning Neural Networks.* Computational Intelligence: An Introduction, Second Edition, 2007: p. 27-54.
113. Melin, P. and O. Castillo, *Unsupervised Learning Neural Networks*, in *Hybrid Intelligent Systems for Pattern Recognition Using Soft Computing*. 2005, Springer. p. 85-107.
114. He, P. and S. Jagannathan, *Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints.* Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 2007. **37**(2): p. 425-436.
115. Hertz, J., *Introduction to the theory of neural computation*. Vol. 1. 1991: Basic Books.
116. Anderson, J.R., et al., *Machine learning: An artificial intelligence approach*. Vol. 2. 1986: Morgan Kaufmann.
117. Blum, A.L. and P. Langley, *Selection of relevant features and examples in machine learning.* Artificial intelligence, 1997. **97**(1): p. 245-271.
118. Basheer, I. and M. Hajmeer, *Artificial neural networks: fundamentals, computing, design, and application.* Journal of microbiological methods, 2000. **43**(1): p. 3-31.
119. Haykin, S.S., et al., *Neural networks and learning machines*. Vol. 3. 2009: Pearson Education Upper Saddle River.
120. Chen, T. and H. Chen, *Approximation capability to functions of several variables, nonlinear functionals, and operators by radial basis function neural networks.* Neural Networks, IEEE Transactions on, 1995. **6**(4): p. 904-910.
121. Ruck, D.W., et al., *The multilayer perceptron as an approximation to a Bayes optimal discriminant function.* Neural Networks, IEEE Transactions on, 1990. **1**(4): p. 296-298.
122. Werbos, P.J., *The roots of backpropagation: from ordered derivatives to neural networks and political forecasting*. Vol. 1. 1994: John Wiley & Sons.
123. Rumelhart, D.E., G.E. Hinton, and R.J. Williams, *Learning representations by back-propagating errors.* Cognitive modeling, 1988.
124. Sowa, J.F., *Knowledge representation: logical, philosophical, and computational foundations.* 1999.