Vilniaus universitetas
Fizikos fakultetas

Zuikio klajonės

Kurso ”Dirbtinis intelektas” projektinė užduotis

Teorinės fizikos ir astrofizikos studijų programa

Studentai Donatas Liupševičius
Rokas Silkinis

Dėstytojas dr. Stepas Toliautas

Vilnius 2020

Turinys

1 Problemos apžvalga

1.1 Uždavinio formuluotė

Zuikio klajonių uždavinys skamba pakankamai paprastai, bet tai nereiškia, kad jis yra lengvai išsprendžiamas. Uždavinys formuluojamas taip: Agentas, kurį vadinsime Zuikiu, juda stačiakampiame lauke. Šiame lauke be Zuikio yra ir kitų agentų-priešininkų – Vilkų. Lauke taip pat yra atsitiktinai išdėstytų objektų – Morkų (vienas iš projektinės užduoties autorių augina triušį ir kategoriškai pareiškė, kad obuolius jie mėgsta labiau, nei morkas). Toliau pateikiamos užduoties sąlygos.

- Kas ėjimą Zuikis pajuda į vieną iš 8 kaimyninių langelių. Zuikis negali likti vietoje.
- Zuikis mato 4 (Manheteno) langelių atstumu visomis kryptimis.
- Zuikis neskiria kompaso krypčių ir nemoka braižyti žemėlapių, bet gali įsiminti, kaip jam sekėsi vienoje ar kitoje situacijoje, apibrėžtoje pagal regėjimo lauko turinį.
- Vilkai mato 4 (Manheteno) langelių atstumu, bet ne už savęs.
- Kol Vilkas nemato Zuikio, jis juda įstrižai per vieną laukelį ir atsimuša nuo sienų.
- Kol Vilkas mato Zuikį, jis juda jo kryptimi (iš 5 galimų) per lygiai du laukelius.
- Kai Vilkas pameta Zuikį, jis vėl pasirenka įstrižą kryptį (iš dešinės) ir eina toliau.
- Vilkai atminties neturi.
- Viena Morka suteikia Zuikiui M energijos.
- Vidutinis atstumas tarp morkų yra $0{,}7M$–$0{,}9M$.
- Suvalgyta Morka iš naujo padedama atsitiktiniame langelyje.
- Zuikis pradeda ėjimus, turėdamas N energijos, kur N – langelių skaičius.
- Viena ėjimas kainuoja 1 energijos vienetą.
- Susitikimas su Vilku kainuoja $N/4$ energijos, bet Zuikis perkeliamas per 4 langelius centro kryptimi. Jei langelyje atsiduria Zuikis, Morka ir Vilkas, Morka lieka nesuvalgyta.
- Praradęs visą energiją Zuikis baigia darbą.

Šio žaidimo tikslas – išmokyti Zuikį išgyventi ir surinkti kiek įmanoma daugiau energijos. Tolimesniuose poskyriuose bus aptariama, kokią metodiką taikant galima pabandyti išmokyti Zuikį išgyventi.

1.2 Markovo sprendimų procesas

Uždavinio formuluotėje yra pabrėžtas Agento noras išgyventi. Kaip jam tą atlikti? Prieš atsakant į šį klausimą reikia reikia apibrėžti kelias sąvokas. **Agentas** yra tam tikroje aplinkoje ir su ja sąveikauja. Šio uždavinio atveju, sąveikos vyksta paeiliui, bėgant laikui. Kiekvienu momentu, agentas turi tam tikrą aplinkos **būsenos** atvaizdą. Turint šį atvaizdą, agentas priima sprendimą ir atlieka **veiksmą**. Dėl šio veiksmo aplinka evoliucionuoja į naują būseną ir agentui yra duodamas **atlygis** už jo ankstesnius veiksmus. Agentas, esantis naujoje būsenoje, vėl priima sprendimą ir atlieką tam tikrą veiksmą, ir t. t. **Markovo sprendimų procesas** (MSP) yra klasikinis nuoseklių sprendimų formalizmas, kuomet priimami veiksmai įtakoja ne tik greitai pasiekiamus, bet ir tolimesnėje ateityje esančius atlygius. Viso šio proceso metu agento tikslas yra maksimizuoti gaunamus atlygius.

Analizuojant MSP, susiduriama su tokiais žymėjimais:

- S – aibė būsenų;
- A – aibė veiksmų;
- R – aibė atlygių.

Apie atlygį galima galvoti kaip apie abstrakčią funkciją $f(S_t, A_t) = R_{t+1}$, kuri atvaizduoja būsenos-veiksmo poras į atlygius. Čia t ir $t + 1$ žymi laiko momentus.

Viena iš svarbiausių sąvokų yra **tikėtinas atlygis** G_t, kas tiesiog yra suma visų atlygių pradedant laiko momentu $t + 1$ ir baigiant paskutiniu laiko momentu T:

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + \ldots + R_T. \tag{1}$$

Tikėtinas atlygis yra labai svarbus, nes būtent tai ir verčia agentą elgtis taip, kaip jis elgiasi. Tačiau kas būtų tuo atveju, jei neegzistuotų paskutinis laiko momentas T, o procesas tęstųsi be galo ilgai? Tuomet tikėtinas atlygis galėtų neturėti baigtinės vertės. Šiai problemai išspręsti naudojama modifikuota tikėtino atlygio sąvoka, apibrėžta taip:

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}, \tag{2}$$

čia γ yra numatymo daugiklis, esantis intervale $0 < \gamma < 1$. Esant tokiai atlygio sąvokai, agentui didesnė įtaką daro artimesni atlygiai, nei tolimesni. Iš šio apibrėžimo išplaukia labai svarbi išraiška, susiejant skirtingų laiko momentų atlygius:

$$G_t = R_{t+1} + \gamma G_{t+1}. \tag{3}$$

1.3 Strategija ir naudingumo funkcija

Agentas gali būti įvairiose būsenose, kiekvienoje kurių jis gali pasirinkti vieną iš galimų veiksmų. Sudėtingiems modeliams galimų procesų skaičius smarkiai išauga. Kadangi šiuo atveju būtų sunku nuspėti, koks kelias yra geriausias, ieškomas ne geriausias kelias, bet ėjimas. Būsenos ar ėjimo gerumą nusako **naudingumo funkcijos**, o tinkamiausią ėjimų visumą – **strategijos**.

Ėjimo ar veiksmo gerumas yra išreikštas atlygiu. Atlygis priklauso nuo to, kokius veiksmus agentas priima įvairiose būsenose. Todėl naudingumo funkcijos priklauso nuo to, kaip elgiasi agentas, t. y. naudingumo funkcijos priklauso nuo strategijos.

Šiame darbe mus domina **veiksmo naudingumo funkcija** (kokybės funkcija) $q_\pi(s, a)$, kuri nusako konkretaus veiksmo a esamoje būsenoje s naudingumą. Ji apibrėžiama taip:

$$q_\pi(s, a) = E_\pi[G_t | S_t = s,\ A_t = a] \tag{4}$$

Kokybės funkcija dažnai vadinama Q-funkcija, o šios funkcijos išvestis bet kuriai būsenos-veiksmo porai vadinama Q-verte.

Kad agentas gautų kaip įmanoma didžiausią atlygį, svarbu rasti **optimaliausią strategiją** π. Optimali strategija tuo pačiu metu turi ir **optimalią Q-funkciją** $q_*(s, a)$, kuri lygi

$$q_*(s, a) = \max_\pi q_\pi(s, a). \tag{5}$$

visiems $s \in S$ ir $A \in a$. Kitais žodžiais tariant, $q_*(s, a)$ duoda didžiausią tikėtiną atlygį kiekvienai būsenos-veiksmo porai (s, a).

Viena iš fundamentaliausių $q_*(s, a)$ savybių yra ta, kad ji turi tenkinti **Belmano optimalumo lygtį**:

$$q_*(s, a) = E\left[R_{t+1} + \gamma \max_{a'} q_*(s', a')\right], \tag{6}$$

čia s' žymi po būsenos s einančią būseną, o a' – būsenoje s' atliekamą veiksmą. Ši lygtis yra kritiškai svarbi norint rasti optimalią Q-funkciją, iš kurios galima rasti optimalią strategiją π.

1.4 Q-mokymasis

Q-mokymasis yra vienas iš mašininio mokymosi metodų norint gauti optimalią strategiją. Q-mokymosi tikslas – rasti optimalią strategiją apskaičiuojant optimalias Q-vertes kiekvienai būsenos-veiksmo porai.

Kaip veikia Q-mokymasis? Agentas pradeda žaidimą nieko nežinodamas apie aplinką, todėl visos Q-vertės yra prilyginamos 0. Šios Q-vertės sudaro taip vadinamą Q-lentelę, kurios dimensijų ilgiai yra būsenų skaičius ir veiksmų skaičius. Q-mokymosi algoritmas iteraciškai atnaujina Q-vertes kiekvienai būsenos-veiksmo porai naudojant Belmano optimalumo lygtį iki tol, kol Q-funkcija $q(s, a)$ konverguoja iki optimalios Q-funkcijos $q_*(s, a)$. Taip laikui bėgant užpildoma Q-lentelė. Kuo ypatinga ši lentelė? Kuomet lentelė tampa gerokai atnaujinta, agentas, būdamas tam tikroje būsenoje, sekantį veiksmą rinksis pagal tai, kurį veiksmą atitinkanti Q-vertė yra didžiausia, nes tai įspėja apie didžiausią tikėtiną

atlygį. Taigi iš esmės, **optimali strategija yra sukonvergavusių Q-verčių Q-lentelė**.

Bet kaip pasirinkti patį pirmą ėjimą, kuomet visos Q-lentelės vertės lygios 0? Į šį klausimą atsako **tyrinėjimo** ir **eksploatacijos** sąvokos. Tyrinėjimas apibūdina aplinkos analizavimą bandant įvairiausius veiksmus ir renkant informaciją. Eksploatacija yra susijusi su jau sukauptos informacijos apie aplinką panaudojimu ieškant maksimalaus atlygio. Kadangi agento tikslas yra maksimizuoti tikėtiną atlygį, iš pirmo žvilgsnio atrodo, kad tyrinėjimo aspektas nėra patrauklus, nes vietoje geresnių veiksmų kartais teks atlikti prastesnius. Tačiau iš kitos pusės, tyrinėjimas padeda surasti naujos informacijos apie aplinką, kurios dėka agentas gali rasti kelią į dar didesnes tikėtino atlygio vertes. Todėl galima pastebėti, kad tiek tyrinėjimas, tiek eksploatacija yra svarbūs procesai ieškant optimaliai strategijai.

Kyla klausimas, kaip apjungti šiuos du vienas kitam prieštaraujančius procesus? Vienas iš būdų yra naudoti ***epsilon-greedy*** **strategiją**. Joje yra apibrėžiamas tyrinėjimo greitis ϵ, kuris iš pradžių lygus 1. Šis dydis yra lygus tikimybei, kad agentas sekančio veiksmo pasirinkimo metu pasirinks atsitiktinį įvykį, dėl ko agentas užsiimtų aplinkos tyrinėjimu. Laikui bėgant, aplinka tampa vis labiau ištirta, ir ϵ vertė yra mažinama mūsų nustatytu greičiu, taip pabrėžiant, kad tyrinėjimą vis labiau pakeičia eksploatacija – agentas tampa vis labiau **godus**.

Kaip yra atnaujinamos Q-vertės? Agentui aplankant vis daugiau būsenų ir atliekant vis daugiau veiksmų, kažkada taip pasitaikys, kad agentas grįžta prie jau matytos būsenos-veiksmo poros. Tokiu atveju Q-vertė jau turima, tačiau dėl labiau ištirtos aplinkos jos vertė galėtų pasikeisti. Naudojant mokymosi spartą α yra nusprendžiama, kiek informacijos pasilikti apie jau turimą konkrečios būsenos-veiksmo Q-vertę ir kiek jos pasilikti iš tos pačios būsenos-veiksmo Q-vertės, bet suskaičiuotos vėlesniu laiko momentu. Kuo didesnis α, tuo stipriau agentas perims naujai suskaičiuotą Q-vertę.

Būsenos-veiksmo porai (s, a) laiko momentu t Q-vertė atnaujinama remiantis tokia išraiška:

$$q^{new}(s,a) = (1-\alpha)q(s,a) + \alpha\left(R_{t+1} + \gamma \max_{a'} q\left(s', a'\right)\right) \tag{7}$$

Šis skaičiavimas bus atliekamas kiekvienu laiko momentu iki kol agento veikla bus nutraukta. Konvergavus šioms Q-vertėms, bus gauta optimali strategija.

Verta pabrėžti, kad naudojant Q-mokymąsi sudėtingiems modeliams (turintiems daug būsenų ir daug veiksmų kiekvienoje būsenoje) optimali būsena bus randama labai lėtai (praktiškai nerandama), nes tikimybė aplankyti tą pačią būseną mažėja didėjant būsenų skaičiui.

Daug daugiau informacijos apie visa tai, kas buvo paminėta čia, galima rasti šaltiniuose [1] ir [2].

2 Metodika

Zuikio klajonių uždavinio sprendimui buvo panaudotas Q-mokymasis. Šio metodo įgyvendinimui buvo parašyta programa panaudojant Python programavimo kalbą. Programos kūrimo metu buvo išskirti keli objektai, kuriems buvo sukurtos klasės: atvaizdavimo, pasaulio, vilko, obuolio ir zuikio. Atvaizdavimo klasė yra mažiausiai svarbi, kadangi ji atsakinga tik už pasaulio atvaizdavimą ekrane, todėl jos nenagrinėsime. Pasaulio klasė yra pagrindinė, ja pasinaudojant vykdoma programa. Pasinaudojant ja yra sugeneruojama pasirinkto dydžio stačiakampė plokštuma, kurioje vaikšto vilkai, zuikis ir guli obuoliai. Programos principas toks, kad joks agentas neveiks be inicijuoto pasaulio. Pačio uždavinio sprendimui pasaulio klasė taip pat nėra labai svarbi. Toliau plačiau aparsime likusias 3 klases – obuolio, vilko ir zuikio.

2.1 Obuolys

Ši klasė yra labai paprasta ir apie ją verta aptarti tik vienintelį dalyką – kaip parenkama kiekvieno naujo obuolio koordinatė pasaulyje.

Uždavinio sąlyga reikalauja, kad vidutinis atstumas tarp morkų būtų $0.7M$–$0.9M$, kur M yra skaičius, kiek energijos vienetų gauna zuikis suvalgęs obuolį. Vidutinis atstumas buvo randamas skaičiuojant euklidinius atstumus tarp visų obuolių. Pridedant naują obuolį, kai jis buvo suvalgytas arba inicializavimo metu, skaičiuojant vidutinį atstumą tarp obuolių yra atsitiktiniu būdų sugeneruojamos naujo obuolio koordinates ir žiūrima, ar tenkinama vidutinio atstumo tarp obuolių sąlyga, jeigu sąlyga netenkinama, koordinatės generuojamos iš naujo iki tol, kol tenkinama sąlyga.

Pastebėta, kad vidutinis atstumas apriboja, kiek energijos gali suteikti obuolys ir maksimalų obuolių skaičių, kadangi kai kuriais atvejais nepavyksta sugeneruoti tokių koordinačių, kad būtų tenkinama sąlyga. Praktiškai, kuo didesnis pasaulis, tuo daugiau energijos gali suteikti obuolys, nes didesniame pasaulyje galimi didesni atstumai. Matosi ir kada artėjama prie maksimalios M vertės tam tikro dydžio pasauliui, nes kuo didesnis M, tuo toliau vienas nuo kito yra obuoliai ir ribiniu atveju obuoliai būna prisispaudę prie sienų, kad tenkintų sąlyga.

2.2 Vilkas

Vilkas yra agentas kuris juda tam tikromis griežtomis taisyklėmis ir nesimoko. Pirminiu laiko momentu jo vieta ir žiūrėjimo kryptis pasaulyje yra parenkama atsitiktinai. Judėjimas paskirstytas į du rėžimus, kai jo matymo lauke nėra zuikio ir yra.

Pirmuoju rėžimu vilkas juda tik per vieną langelį žiūrėjimo kryptimi įstrižai į kairę arba į dešinę iki kol prieina siena. Žingsnio metu, kai sekanti jo koordinatė būtų sienos vietoje, jis atsimuša tarsi biliardo kamuolys, pakeičia kryptį ir juda toliau įstrižai.

Antrasis rėžimas įsijungia, jeigu vilko matymo lauke yra zuikis, tuomet vilko galimi žingsnių variantai pasikeičia ir sekantis žingsnis yra tas kuris Euklidiniu atstumu yra arčiausiai zuikio. Taip užtikrinama, kad vilkas gaudytų zuikį. Zuikiui pabėgus iš matymo lauko Vilko judėjimas vėl persijungia į pirmąjį ir sekantis žingsnis yra jo žiūrėjimo kryptimi įstrižai į dešinę.

Vilkas turi 4-ias žiūrėjimo kryptis – į viršų, apačią, kairę ir dešinę. Kiekvienai krypčiai sugeneruoti sąrašai, kurie leidžia iš pasaulio reliatyviai vilko koordinatei nustatyti pasaulio koordinates kurias mato vilkas.

Kiekvieno pasaulio ciklo gale vilko nauja ir buvusi koordinatės yra išsaugomos. Vilkų pasaulyje gali būti pasirinktinai kiek norime.

Atlikus bandymus pastebėjome, kad galima sukurti tokį pasaulį pagal uždavinio sąlygą, kad zuikis ir visiškai atsitiktinai klaidžiojantis galėtų išgyventi daugiau ciklų nei pasaulyje yra langelių. Pavyzdžiui, kai $M = 8$, o pasaulio langelių skaičius yra $N = 169$, o obuolių skaičius 25 - zuikis išgyvena ilgiau. Taip pat galima sudaryti sąlygas, kad nuo pirmo ir paskutinio paleidimo zuikis išgyventų, tokiu atveju, net atsitiktinai klaidžiojant zuikio energija vis didėja. Tai atvejis, kai $M = 5$, $N = 49$ ir yra 25 obuoliai. Tokių kombinacijų galima būtų rasti ir daug daugiau.

2.3 Zuikis

Zuikio klasė yra esminė, kadangi ji aprašo besimokantį agentą. Zuikis optimalios strategijos paieškai naudoja Q-mokymąsi.

Kaip ir Obuolio bei Vilko klasės objektai, zuikis turi būsenos atnaujinimo funkciją, kurioje eiliškumo tvarka įvyksta šie procesai:

- suskaičiuojama zuikio būsena, kuri yra apibrėžta jo matymo lauko;
- įvertinamas susidūrimas su Vilko klasės objektais (jei lieka gyvas, pajudinamas centrinio langelio link per 4 langelius);
- jei vilkas, obuolys ir zuikis yra viename langelyje, zuikis obuolio nepaima, bet jei vilkas kitame langelyje, obuolys suvalgomas;
- kitu atveju, zuikis tiesiog pajuda;
- pabaigoje patikrinama, ar zuikis vis dar gyvas. Jei nebe, jis ”atgimsta” naujame langelyje.

Zuikio būsena apibrėžiama taip pat, kaip ir vilko atveju – matymo lauko langeliai užpildomi skaliariniais dydžiais, kurie nusako, kas tame langelyje yra (prioriteto mažėjimo tvarka: priešinis agentas, obuolys, siena, tuščias langelis). Pabrėžtina, kad zuikio matymo laukas yra dvigubai didesnis nei vilko ir jis naudojamas strategijos mokymuisi, kuomet vilko būsena naudojama tik nustatant, ar jo matymo lauke yra zuikis ar ne.

3 Rezultatai ir jų aptarimas

Atlikus bandymus pastebėjome, kad galima sukurti tokį pasaulį pagal uždavinio sąlygą, kad zuikis ir visiškai atsitiktinai klaidžiojantis galėtų išgyventi daugiau ciklų nei pasaulyje yra langelių. Pavyzdžiui, kai $M = 8$, o pasaulio langelių skaičius yra $N = 169$, o obuolių skaičius 25 – zuikis išgyvena ilgiau. Taip pat galima sudaryti sąlygas, kad nuo pirmo ir paskutinio paleidimo zuikis išgyventų, tokiu atveju, net atsitiktinai klaidžiojant zuikio energija vis didėja. Tai atvejis, kai $M = 5$, $N = 49$ ir yra 25 obuoliai. Tokių kombinacijų galima būtų rasti ir daug daugiau.

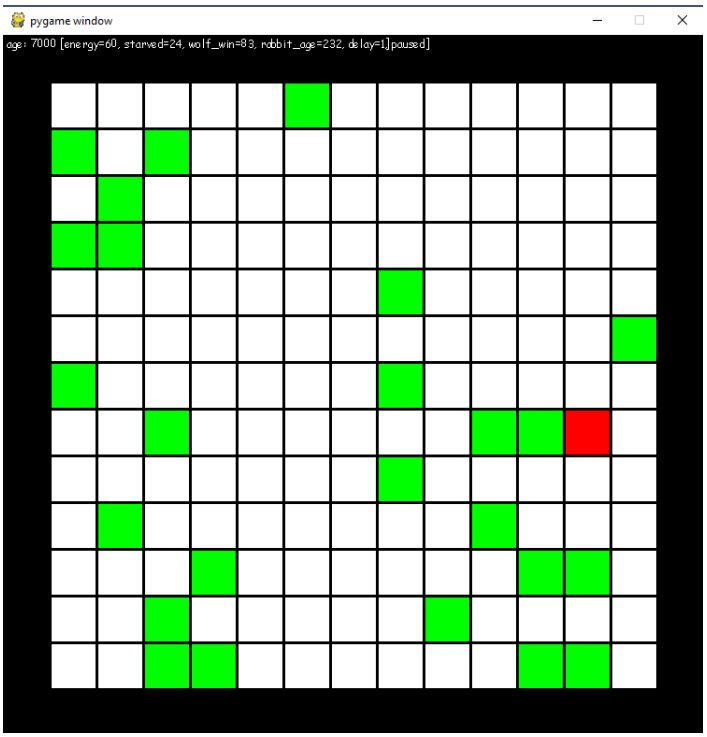
Dar vienas būdas, kaip patikrinti, ar metodika veikia, galėtų būti toks: kelis kartus palyginti atsitiktinio zuikio klaidžiojimo ir besimokančio zuikio rezultatus po pakankamai didelio skaičiaus iteracijų tame pačiame pasaulyje. Kadangi besimokančio zuikio pradžioje dominuoja atsitiktinumas, didelio skirtumo tarp jo ir tarp atsitiktinai klaidžiojančio neturėtų būti, bet eigoje jie turėtų išryškėti. Galimi parametrai, kuriuos galima tikrinti: zuikio mirčių nuo vilkų ir bado vertės, ilgiausias zuikio išgyventas laiko tarpas be mirčių.

Panašų, bet kiek besikiriantį palyginimą atlikome ir mes. 1–3 pav. pateikti atsitiktinio klaidžiojimo ($\epsilon = 1$), klaidžiojimo su $\epsilon = 0{,}5$ ir godaus klaidžiojimo su $\epsilon = 0{,}1$ rezultatai (*epsilon* šiais atvejais nebekito, o buvo fiksuoti). Visais trimis bandymais agentams buvo suteikta prieiga prie tos pačios Q-lentelės, kurią suformavo besimokantis agentas, atlikęs 400 000 žingsnių. Kiekvieno pav. pavadinime pateikti keli parametrai, tačiau šiuo atveju verta atkreipti dėmesį į 2 iš jų: *starved* (nurodantį, kiek kartų zuikis mirė iš bado) ir *wolf_win* (nurodantį, kiek kartų zuikis mirė dėl susidūrimo su vilku). Galima pastebėti, kad atsitiktinio klaidžiojimo metu gauti rezultatai yra prasčiausi: $starved = 24$ ir $wolf\_win = 83$. Geriausiai pasirodė godaus klaidžiojimo agentas: $starved = 17$ ir $wolf\_win = 17$.

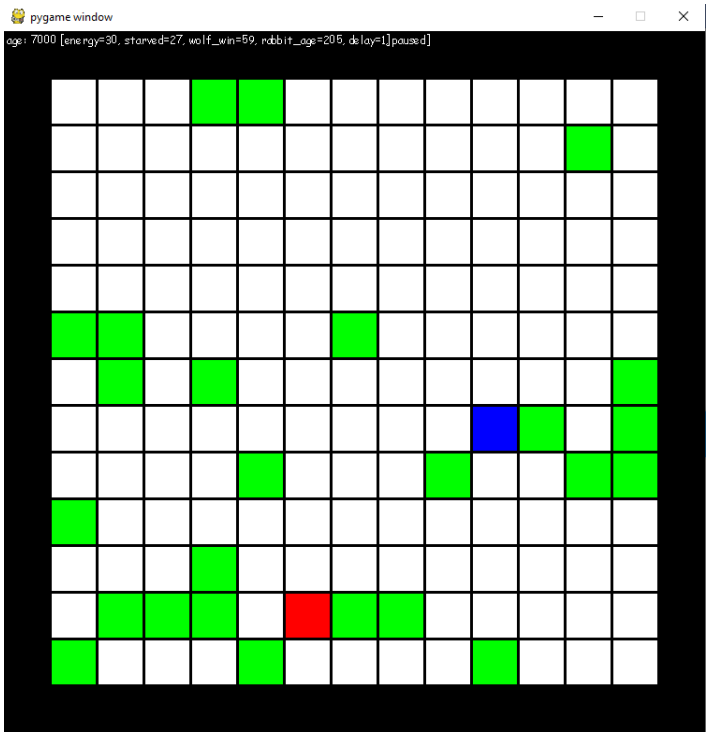
Paskutiniame 4 pav. pateiktas mažesnis pasaulis, kur zuikis buvo apmokytas 800 000 ciklų ir jo gebėjimas išgyventi godžiu režimu. Šiam atvejui reikalaujama išgyveno sąlyga yra 49 žingsniai. Pagal gautus rezultatus iš 5000 žingsnių zuikis išgyvena vidutiniškai 161 žingsnį.

Pabrėžtina, kad Q-mokymosi metodika šiam uždaviniui spręsti galimai nėra pats efektyviausias variantas. Zuikio regėjimo lauką sudaro 40 langelių ir kiekvienas jų gali turėti bent vieną iš keturių skaliarinių verčių (žinoma, ne visi langeliai gali būti sienomis, nes jais yra tik kraštiniai langeliai), kas reiškia, kad vienai zuikio būsenai apibūdinti reikia 40 kintamųjų, kurių kiekvieno apibrėžimo srities dydis yra 1–3 elementai.

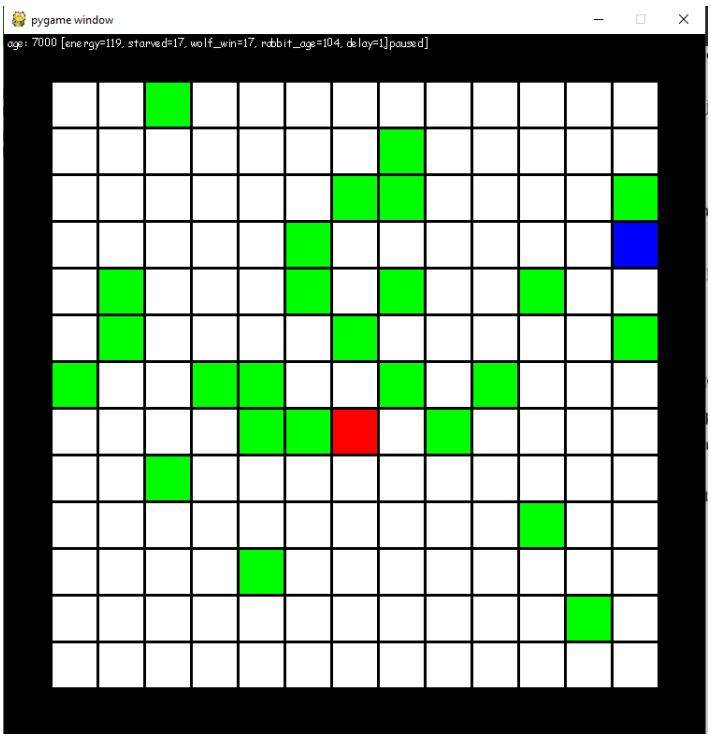
Vieni iš skaičiavimų, kurių neatlikome, bet pasirodė įdomus: Q-lentelės verčių skaičiaus priklausomybė nuo laiko. Natūralu tikėtis, kad bėgant laikui agentas būna aplankęs vis daugiau būsenų ir tikimybė surasti naują būseną mažėja. Matant Q-lentelės verčių skaičiaus konvergavimą galbūt būtų galima daryti išvada apie tai, kad agentas artėja prie optimalios strategijos radimo.



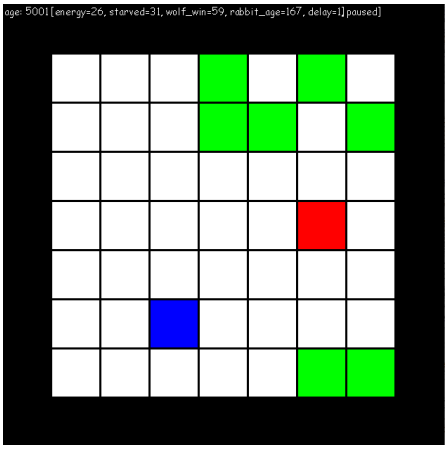
1 pav. Atsitiktinis: $starved = 24$, $wolf\_win = 83$, $age = 7000$, $\epsilon = 1$



2 pav. $starved = 27$, $wolf\_win = 59$, $age = 7000$, $\epsilon = 0.5$



3 pav. Godus: $starved = 17$, $wolf\_win = 17$, $age = 7000$, $\epsilon = 0.1$



4 pav. Godus: $starved = 31$, $wolf\_win = 59$, $age = 7000$, $\epsilon = 0.1$

Išvados

- Parašyta programa, simuliuojanti zuikio klajonių uždavinį.
- Zuikio vaikščiojimui ir mokymuisi panaudotas Q-mokymosi metodas.
- Dėl didelio būsenų skaičiaus, norint gauti optimalią strategiją reikia daug resursų, kadangi progresas matomas lėtai.
- Surasti pasaulio parametrai, kur zuikis išgyvena daugiau žingsnių nei yra pasaulyje langelių.
- Surasti pasaulio parametrai, kur zuikis visada išgyvena ir energija pastoviai didėja.

Literatūra

[1] R. S. Sutton and G. B. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 2018.

[2] S. Russel and P. Norvig. *Artificial Intelligence: A Modern Approach*. 3rd ed. Pearson, 2009.