UNIVERSITATEA TEHNICĂ "GHEORGHE ASACHI" DIN IAȘI

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

SPECIALIZAREA CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI

LUCRARE DE DIPLOMĂ – RAPORTUL 1

ÎNVĂȚAREA MEDIULUI DINAMIC AL UNUI JOC ORIGINAL FOLOSIND METODE DE ÎNVĂȚARE CU ÎNTĂRIRE IERARHICĂ

Profesor coordonator,

Florin Leon

Student,

Vlad Batalan

Iași, 2022

1. *Scopul temei alese*

Creierul uman reprezintă cel mai complicat și misterios organ, având în compoziția sa în medie 86 de miliarde de neuroni, capabil nu doar de execuția unor proceduri complexe de orientare în mediul înconjurător, cât și de procese psihice superioare oricărei specii. În acest context, au fost realizate numeroase studii cu scopul de a crea un model capabil să imite și chiar să elaboreze de la sine astfel de procese. Un exemplu de studiu este cel al funcției dopaminergice mezencefală, unde învățarea cu diferență temporală a constituit un framework propice ("*A model of how the basal ganglia generate and use neural signals that predic reinforcement*" – J. C. Houk, C.M. Adams, A.G. Barto). Dacă pentru o persoană activitatea de a găti niște ochiuri la aragaz reprezintă o acțiune simplu de realizat și banală, pentru un robot reprezintă o sarcină dificilă deoarece presupune aprinderea focului, plasarea tigăii, spargerea ouălor, aruncarea cojilor și lăsarea ochiurilor în tigaie un timp până se pregătesc. Toate aceste acțiuni prezintă în componența lor și mai multe subacțiuni precum mișcarea brațului pe o axă un număr de unități, colectarea datelor vizuale, analiza obiectelor și a distanței până la acestea și multe altele. În această direcție, niciun subiect din domeniul machine learning nu are un impact mai profund si mai susținut în psihologie și neuroștiință decât învățarea cu întărire computațională.

Principalul scop al temei alese este acela de a crea un model care se poate adapta la un mediu specific de tip simulare în care agentul are în fiecare moment de timp posibilitatea de a alege ce acțiune să realizeze dintr-un set bine definit. În mediul respectiv există diferite obstacole, baricade, obiective, zone care nu pot fi accesate fără realizarea unor pași intermediari, fapt ce creionează dificultatea ridicată de rezolvare a problemei propuse. Întregul proces de învățare și adaptare a modelului este realizat în contextul unei aplicații de tip platformer care prezintă o caracteristică specială a agentului: acesta se poate teleporta în timp și poate interacționa indirect cu instanțele sale din trecut. Scopul urmărit de agent este de a evada din camera în care se află prin intermediul unei porți care trebuie activată (Figura 1).

Deoarece mediul este dinamic, obiectele cu care interacționează agentul și obiectivele intermediare sunt în continuă schimbare, o abordare clasică a învățării cu întărire nu este suficientă. Astfel, lucrarea va avea în vedere o abordare ierarhică a învățării cu întărire prin faptul că alegerile acțiunilor nu vor fi făcute doar la nivel primar, ci și la nivel mai înalt prin intermediul subrutinelor, fiecare având un anumit scop intermediar prestabilit.

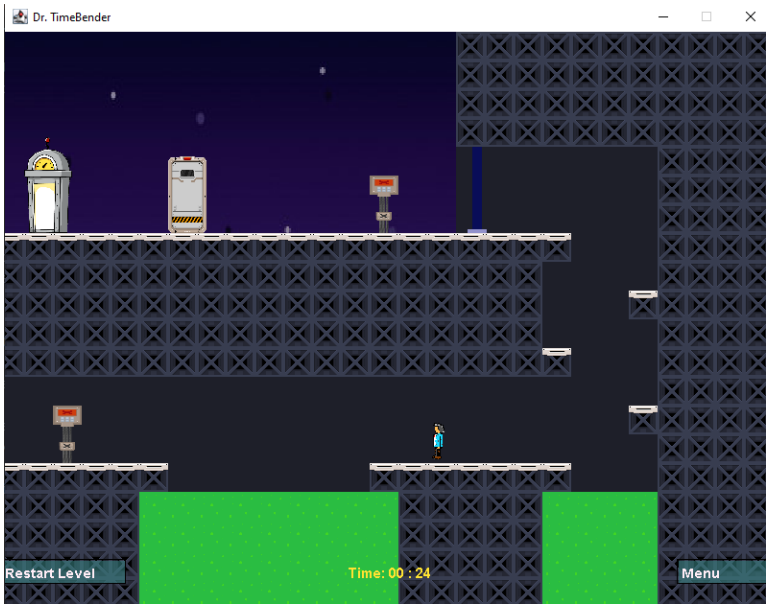

Figura 1

2. *Referințe la teme/subiecte similare*

[1] "*Hierarchically organized behaviour and its neural foundations: A reinforcement learning perspective*" – Matthew M. Botvinick, Yael Niv, Adrew G. Barto. (https://www.sciencedirect.com/science/article/pii/S0010027708002059)

[2] "*Hierarchical reinforcement learning and decision masking*" – Matthew Michael Botvinick (https://www.sciencedirect.com/science/article/pii/S0959438812000876).

[3] "*Model-based hierarchical reinforcement learning and humain action control*" – Matthew Botvinick, Ari Weinstein. (https://royalsocietypublishing.org/doi/full/10.1098/rstb.2013.0480)

[4] "*Recent Advances in Hierarchical Reinforcement Learning*" – Andrew G. Barto, Sridhar Mahadevan. (https://link.springer.com/content/pdf/10.1023/A:1022140919877.pdf)

[5] "*Train <<Undying>> Flappy bird Using Reinforcement Learning in Java*" – Quing Lan (https://towardsdatascience.com/train-undying-flappy-bird-using-reinforcement-learning-on-java-98ff68eb28bf)

[6] "*Make smarter agents with Hierarchical Reinforcement Learning*" – Mauricio Feder Argerich (https://towardsdatascience.com/hierarchical-reinforcement-learning-a2cca9b76097)

[7] "*The MAXQ Method for Hierarchical Reinforcement Learning"* - Thomas G. Dietterich (http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.9.313&rep=rep1&type=pdf)

3. *Resurse hardware/software utilizate*

În privința resurselor de tip hardware, se va utiliza o mașină de calcul care dispune de urmatoarele specificații:

- Minim 8GB memorie RAM,
- Placă grafică integrată NVIDIA GeForce GTX 950M
- Procesor I7 4800HQ.
- Sistem de operare de tip Windows 10+ sau Linux

Pentru partea software, se vor utiliza următoarele:

- IDE-uri: *Intelij Ultimate* pentru Java și *Pycharm* pentru Python
- Bază de date aplicație tip platformer: *SQLite*
- Interfață aplicație platformer: *Java awt* și *Java Swing*
- Framework pentru utilizare învățare cu întărire în Java: bilbioteca de tip open-source DJL (*Deep Java Library*: https://github.com/deepjavalibrary/djl), fiind un API construit pe bilbiotecile din Python inteligență artificială*: mxnet, PyTorch, ONNX RUNTIME, TensorFlow.*
- Framework pentru învățare cu întărire în Python: *PyTorch, TensorFlow*

4. Algoritmi sau metode alese

Din cauza mecanicii de teleportare în timp și interacțiunea agentului cu instanțele sale din trecut, problema nu poate fi rezolvată prin aplicarea algoritmului QLearning clasic, ci o variantă ierarhică a acestuia (Hierarhical QLearning [4]) care presupune un strat în plus de abstractizare. Acesta ia in considerare și subrutine numite opțiuni, fiecare având propria pilitică, condiție de terminare și un set de stări din care poate fi inițiată. O variantă alternativă apropiată poate fi și metoda MAXQ [7] care propune realizarea unor arbori de precedență a subrutinelor.

Metoda de împărțire a problemei in subprobleme utilizată va fi delimitarea prin acțiunea de teleportare în timp, astfel încât fiecare instanță a agentului va avea la un moment dat un scop intermediar diferit: de a ajunge a panoul de comandă care face posibilă trecerea în camera următoare pentru instanțele din viitor. În funcție de obstacolele prezente, se pot antrena separat diferite subrutine. Un exemplu ar putea fi subrutina de așteptare: odată ce este atins un panou de comandă, agentul trebuie să staționeze destul timp cât să permită instanțelor din viitor să treacă în încăperea următoare.

5. *Rezultate aşteptate*

Pentru început, va fi realizată o variantă minimală a aplicației de tip platformer ca să minimizeze timpul de antrenare a modelului. Aceasta va include posibilitatea de a alege dacă se afișează componentele grafice sau nu și va avea viteza jocului mărită. Odată ce va fi realizată aplicația minimalistă, vor fi realizate scripturi în Python de învățare prin întărire pentru realizarea modelului, utilizând biblioteci precum PyTorch și TensorFlow. În ultima parte a proiectului, modelul va fi importat din nou în aplicația principală și vor fi testate rezultatele a diferitor abordări în cadrul real.

Datele cu privire la modul în care a funcționat procesul de antrenare vor fi folosite ca suport în formularea unei concluzii cu privire la eficiența metodelor ierarhice ale învățării cu întărire.