# DISA ALGORITMA ITERATIVË BAZUAR NË DIFERENCËN E PËRKOHËSHME PËR TË MËSUAR NJË POLITIKË OPTIMALE TË VEPRUARI NË NJË PROCES MARKOVI

**JEZUINA KOROVESHI., ANA KTONA., DENADA ÇOLLAKU.,**

**LORETA LEKA.**

Universiteti i Tiranës, Fakulteti i Shkencave të Natyrës, Departamenti i Informatikës

e-mail: jezuina.koroveshi@fshn.edu.al

**Përmbledhje**

Të mësuarit me anë të përforcimit është një mënyrë të mësuari bazuar në prova dhe gabime, derisa agjenti të marrë një sinjal pozitiv nga mjedisi, që i tregon atij se veprimi i kryer është veprim i dëshiruar. Në ndryshim nga format e tjera të të mësuarit të makinës (të mësuarit e mbikëqyrur), agjentit nuk i tregohet se çfarë veprimi duhet të kryejë në një gjendje të caktuar, por ai duhet ta zbulojë vetë veprimin, i cili e dërgon nga një gjendje në një tjetër dhe shoqërohet me një shpërblim nga mjedisi. Mjedisi mund të paraqitet formalisht me anë të një procesi vendimi Markovi, i cili ka veçorinë që gjendjet e ardhshme varen vetëm nga gjendja aktuale dhe jo nga gjendjet e shkuara. Në këto kushte, agjenti duhet të mësojë një politikë optimale veprimi, e cila për çdo gjendje në të cilën ndodhet zgjedh veprimin që duhet të kryejë, në mënyrë që të maksimizojë shpërblimin total të grumbulluar si pasojë e veprimeve të kryera. Kjo formë të mësuari ka gjetur aplikime në drejtime të ndryshme, si psh: sistemi AlphaGo i cili mund të luajë lojën "GO" dhe të fitojë ndaj një lojtari ekspert; të kuptuarit dhe të procesuarit e gjuhëve natyrale; sisteme të adaptueshëm në varësi të përdoruesit etj. Punimi do të paraqesë dhe do të trajtojë në planin krahasues disa metoda iterative bazuar në diferencën e përkohshme, si Q-Learning dhe SARSA, për të mësuar nje politikë optimale veprimi në një mjedis Markovi. Do të trajtohen kushtet e konvergjencës, parametrat, dilema midis eksplorimit dhe shfrytëzimit, dhe si ndikojnë ato në përshpejtimin apo ngadalësimin e procesit të të mësuarit të agjentit.

**Fjalëkyçe:** Proces Markovi, SARSA, Q-Learning, diferencë e përkohshme.

**Abstract**

Reinforcement learning is a way of learning based on trial and error, until the agent receives a positive signal from the environment, which shows that the action taken was a desirable one. Unlike other forms of machine learning, such as supervised learning, the agent is not taught what action to take in a certain state. He must discover what action to take when he is in a certain state, which will move him in a new state, and will be accompanied by a reward from the environment. The environment can be formally specified as a Markov decision process, that has the property that future states depend only on the present state and not on past ones. In these conditions, the agent must learn an optimal policy, which maps every state with an action, with the purpose of maximizing the total reward gathered from actions taken. This form of learning has applications in different areas, such as:

AlphaGo system that plays the well-known game of GO, and beat a high-ranking GO player; understanding and processing of natural language; adaptive systems etc. This paper will describe and compare some of the iterative algorithms based on temporal difference, such as Q-learning and SARSA, that are used to learn an optimal policy in a Markov process. It will analyze convergence conditions, parameters, the exploration-exploitation dilemma, and how do they affect in slowing or speeding up the learning process.

**Keywords**: Temporal difference, Markov process, Q-learning, SARSA.

## 1. Hyrje

Të mësuarit me anë të përforcimit është një mënyrë të mësuari bazuar në ndërveprimin me mjedisin dhe eksperiencën e fituar si pasojë e këtij ndërveprimi. Në ndryshim nga format e tjera të të mësuarit të makinës, individit që mëson nuk i tregohet paraprakisht se çfarë veprimi duhet të kryejë në çdo gjendje ku mund të ndodhet. Në sajë të ndërveprimit me mjedisin, ai duhet të zbulojë vetë veprimet që i sjellin fitimin më të madh. Problemi që lind është se si të mësojmë çfarë veprimi duhet kryer në çdo situatë që ndodhemi, me qëllim që të kemi një përfitim sa më të madh si rezultat i veprimeve të kryera. Kjo formë të mësuari ka një fushë të gjerë aplikimesh si: robotika, lojra, procesim i gjuhëve natyrale, menaxhim biznesi (reklamim, rekomandim, menaxhim klientësh, marketing), shëndetesi etj (Li, 2017). Tesauro përdori diferencën e përkohshme të ndërthurur me rrjetet neurale për të luajtur lojën Backgamon (Tavëll), duke arritur nivele të krahasueshme me njeriun.(Tessauro, 1993). Të mësuarit me anë të përforcimit, rrjetat neurale dhe pemët e kërkimit janë përdorur të kombinuara duke krijuar programin AlphaGo për të luajtur lojën Go (Silver *et al*., 2016), e cila është një nga lojërat klasike më sfiduese për inteligjencën artificiale, për arsye të numrit shumë të madh të hapësirës së kërkimit si edhe të vështirësisë së vlerësimit të pozicionit dhe lëvizjeve. Në mars 2016, AlphaGo arriti të fitojë kundër Lee Sedol, i cili është 18 herë kampion bote për lojën Go.

Algoritmi Q-Learning dhe rrjetat neurale janë metodat e përdorura për të luajtur disa prej lojrave Atari 2600, më e njohura midis të cilave është loja Breakout (Mnih *et al*., 2013)

## 2. Përcaktime formale

Në të mësuarit e makinës, entiteti që mëson apo që merr vendimet është agjenti. Çdo gjë jashtë agjentit, me të cilën ai ndërvepron, quhet mjedis. Agjenti dhe mjedisi ndërveprojnë gjatë çdo sekuence diskrete të intervaleve kohore t=0,1,2,3….. Gjatë çdo intervali kohor t, agjenti percepton gjendjen e mjedisit në të cilën ndodhet, $S_t \in S$ ku S është bashkësia e të gjitha gjendjeve të mundshme të mjedisit. Më pas kryen veprimin $A_t \in A(S_t)$ ku $A(S_t)$ është bashkësia e veprimeve të mundshme që mund të kryhen në gjendjen $S_t$. Në intervalin pasardhës, si pasojë e veprimit të kryer, mjedisi i përgjigjet agjentit duke e cuar në një gjendje të re $S_{t+1}$ , si edhe duke i dhënë një shpërblim $R_{t+1} \in R \subset R$ (Sutton & Barto, 1998).

### 2.1 Proces vendimi Markovi

Në modelin e të mësuarit me anë të përforcimit, agjenti i kryen veprimet si funksion i gjendjes së mjedisit në të cilën ndodhet. Në mënyrë ideale, një gjendje e mjedisit do të jetë rezultat i gjendjeve paraardhëse, dhe njëkohësisht do të përmbajë të gjithë informacionin e nevojshëm për të vazhduar në gjendjet e tjera. Një gjendje e tillë thuhet se ka veçorinë e Markovit (Sutton & Barto, 1998). Nëse gjendja e mjedisit ka veçorinë e Markovit, atëherë përgjigja e mjedisit në kohën t+1 do të varet vetëm nga gjendja dhe veprimi i kryer në kohën t. Në këtë rast dinamika e mjedisit mund të përcaktohet vetëm me anë të probabilitetit (Sutton & Barto, 1998):

$$p(s', r|s, a) = Pr\{ S_{t+1} = s' , R_{t+1} = r \mid S_t = s, A_t = a \}$$

për çdo s, s',r dhe a. Një problem të mësuari, që ka veçorinë e Markovit, quhet proces vendimi Markovi (PVM) (Sutton & Barto, 1998). Nëse bashkësia e gjendjeve është e fundme, ateherë quhet proces vendimi Markovi i fundëm. Një PVM i fundëm, përcaktohet nga bashkësia e gjendjeve S, veprimeve A, probabilitet e kalimit nga njëra gjendje tek tjetra P (s'| s, a), si edhe nga funksioni i shpërblimit R(s) (Sutton & Barto, 1998). Për çdo gjendje s dhe veprim a, çifti pasardhës gjendje-shpërblim, përcaktohet si me poshte (Sutton & Barto, 1998):

$$p(s', r|s, a) = Pr\{ S_{t+1} = s' , R_{t+1} = r \mid S_t = s, A_t = a\}$$

Në këtë mënyrë mund të përcaktohet plotësisht një PVM i fundëm.

### 2.2 Funksioni i utilitetit

Funksioni i utilitetit do të varet nga sekuenca e gjendjeve të vizituara (historiku i mjedisit). Në çdo gjendje s, agjenti merr shpërblimin R(s), i cili është një vlerë numerike e kufizuar. Mënyrat për përcaktimin e funksionit të utilitetit janë dy (Russell & Norvig, 2010):

a. Mbledhja e shpërblimeve

$$U_h([s_0 , s_1 , s_2 , \ldots]) = R(s_0 ) + R(s_1 ) + R(s_2 ) + \cdots .$$

b. Shpërblimet e reduktuara

$$U_h([s_0 , s_1 , s_2 , \ldots]) = R(s_0 ) + \gamma R(s_1 ) + \gamma 2\, R(s_2 ) + \cdots$$

ku $\gamma$ është faktori i reduktimit, ka vlerë midis 1 dhe 0, dhe tregon preferencën e agjentit për shpërblimet e menjëherëshme ndaj shpërblimeve në të ardhmen.

### 2.3 Politika e veprimit

Politika specifikon se çfarë veprimi duhet të kryejë agjenti në çdo gjendje ku mund të ndodhet. Zakonisht shënohet me $\pi$, dhe $\pi(s)$ është veprimi i rekomanduar për t'u kryer në gjendjen s. Në këtë mënyrë, pavarësisht rezultatit të veprimit, agjenti e di gjithmonë çfarë veprimi duhet të kryejë (Russell & Norvig, 2010). Për arsye të natyrës stohastike të një PVM, historia e gjendjeve të vizituara sa herë që zbatohet një politikë e caktuar, do

të jetë e ndryshme. Cilësia e një politike do të matet me anë të utilitetit të pritshëm të gjendjeve të mundshme të gjeneruara nga kjo politikë (Russell & Norvig, 2010). Utiliteti i pritshëm, i përftuar si pasojë e zbatimit të politikës π, nisur nga gjendja s do të jetë (Russell & Norvig, 2010):

$$U\pi(s) = E\left[\sum_{t=0}^{\infty} \gamma^t * R(St)\right]$$

ku pritshmëria lidhet me shpërndarjen e probabiliteteve mbi sekuencën e gjendjeve të përcaktuara nga gjendja fillesetare s dhe politika π. Politika optimale, π*, është ajo që do të japë utilitetin e pritshëm më të madh.

### 2.4 Funksionet vlerë

Funksioni vlerë është një vlerësim që i bëhet gjendjes së mjedisit (ose çiftit gjendje-veprim), për të treguar se sa e favorshme është për agjentin të ndodhet në një gjëndje të caktuar (ose të kryeje një veprim në një gjendje të caktuar). Vlerësimi bëhet mbi bazën e utilitetit të pritshëm të një gjendjeje ose të një çifti gjendje-veprim (Sutton & Barto, 1998). Duke qënë se shpërblimet e marra varen nga veprimet e kryera, dhe veprimet varen nga politika e ndjekur, atëherë edhe funksionet vlerë do të përcaktohen mbi bazën e një politike të caktuar. Vlera e një gjendjeje s, në bazë të një politike π, për një PVM përcaktohet si më poshtë (Sutton & Barto, 1998):

$$v_\pi(s) = E\pi[Ut \mid St = s] = E\left[\sum_{k=0}^{\infty} \gamma^k * R_{t+k+1} \mid St = s\right]$$

Ky është funksioni i vlerës së gjendjes për një politikë π. Në mënyrë të ngjashme përcaktohet edhe vlera që ka kryerja e një veprimi a në gjendjen s, e cila shënohet qπ (s, a) dhe nënkupton utilitetin e pritshëm nëse fillojmë nga gjendja a, kryejmë veprimin s dhe më pas ndjekim politikën π (Sutton & Barto, 1998):

$$q_\pi(s,a) = E\pi[Ut \mid St = s, At = a] = E\left[\sum_{k=0}^{\infty} \gamma^k * R_{t+k+1} \mid St = s, At = a\right]$$

Ky është funksioni i vlerës së veprimit për një politike π.

## 3. Metodat e bazuara ne diferencën e përkohshme

Metodat e bazuara në diferencën e përkohshme (DT) janë një kombinim midis metodave Monte Carlo dhe programimit dinamik. Këto metoda mund të mësojnë direkt nga eksperienca, pa patur të nevojshme të kenë paraprakisht një model të dinamikave të mjedisit (Sutton & Barto, 1998). Metodat e DT klasifikohen në dy grupe:

1. Metodat për parashikim:

Këto metoda bëjnë një vlerësim në mënyrë iterative të funksionit vlerë të çdo gjendjeje të mjedisit, sipas një politike të caktuar π (Sutton & Barto, 1998).

2. Metodat për kontroll:

Metodat për kontroll bëjnë një vlerësim në mënyrë iterative të funksionit të vlerës së një veprimi të caktuar për cdo gjëndje të mjedisit. Këto metoda ndahen në dy grupe (Sutton & Barto, 1998):

1. Metoda e bazuar në një politikë veprimi (SARSA)

2. Metoda që nuk është e bazuar në një politikë veprimi (Q-Learning)

### 3.1 SARSA

SARSA është një metodë e bazuar në diferencën e përkohshme për të mësuar vlerën e çdo çifti gjendje-veprim ($q\pi$ (s, a)) sipas një politike të caktuar $\pi$ (Sutton & Barto, 1998). Në këtë rast, kemi një politikë veprimi $\pi$, sipas së cilës përcaktohet veprimi që do të kryhet në secilën gjendje. Duke ndjekur këtë politikë, kalohet nga njëra gjendje në tjetrën, derisa të arrihet në një gjendje fundore. Duke qënë se secili veprim shoqërohet me një gjendje të re dhe një shpërblim, algoritmi mëson se cila është vlera e kryerjes së veprimit a kur jemi në gjendjen s, duke bërë të mundur që të zgjedhim gjithmonë veprimin me vlerë më të madhe. Për të realizuar këtë gjë, algoritmi merr në konsideratë kalimin nga çifti gjendje-veprim, tek çifti pasardhës gjendje-veprim, duke përdorur rregullin e mëposhtëm për të bërë përditësimin e vlerave të çdo çifti (Sutton & Barto, 1998):

$$Q(St , At ) \leftarrow Q(St , At ) + \alpha [ Rt+1 + \gamma Q(St+1 , At+1 ) - Q(St , At ) ].$$

Ky rregull merr në konsideratë çdo element nga pesëshja (St , At , Rt+1 , St+1 , At+1), nga merr edhe emrin SARSA (Sutton & Barto, 1998). Forma e përgjithshme e algoritmit SARSA është dhënë në figurën 1. (Sutton & Barto, 1998). Algoritmi SARSA konvergjon në një politikë optimale, nën kushtet e mëposhtme (Singh *et al*., 2000):

a. Të gjitha çiftet gjendje-veprim vizitohen një numër të pafundëm herësh.

b. Politika e zgjedhur konvergjon në limit te politika greedy.

Inicializo Q(s,a), $\forall s \in S$, $a \in A(s)$, në mënyrë arbitrare, dhe Q(gjëndje fundore)= 0

Përsërit (për çdo episod)

    Inicializo gjendjen S

    Zgjidh një veprim A për gjendjen S, duke përdorur një politikë të caktuar (psh, Ɛ-greedy)

    Përsërit (për çdo hap të episodit)

        Kryej veprimin A, observo shpërblimin R dhe gjendjen pasardhëse S'

        Zgjidh veprimin A' për gjendjen S' bazuar në një plotikë të caktuar (psh, Ɛ-greedy)

        $Q(St , At ) \leftarrow Q(St , At ) + \alpha [ Rt+1 + \gamma Q(S', A' ) - Q(St , At )$

S ← S', A ← A'

Gjersa S të jetë gjendje fundore

**Figura 1**. Algoritmi SARSA

### 3.2. Q-Learning

Algoritmi Q-Learning jep mundësinë që të përafrohet vlera optimale për çdo çift gjendje-veprim, pavarësisht politikës që jemi duke ndjekur. Kjo gjë bëhet në sajë të mënyrës së përditësimit të vlerës së q, e cila merr në konsideratë vlerën maksimale të veprimeve në gjendjen pasardhëse (maxa Q(S', a)) sipas formulës së mëposhtme (Sutton & Barto, 1998):

$Q(St , At ) \leftarrow Q(St , At ) + \alpha (Rt+1 + \gamma \text{ maxa } Q(St+1 , a)) - Q(St , At )$ .

Forma e përgjithshme e algoritmit Q-Learning është dhënë më poshtë:

Inicializo Q(s,a), ∀s ∈ S, a ∈ A(s), në mënyrë arbitrare, dhe Q(gjendje fundore)= 0

Përsërit (për çdo episod)

Inicializo gjendjen S

Përsërit (për çdo hap të episodit)

Zgjidh një veprim A për gjendjen S, duke përdorur një politikë të caktuar (psh. Ɛ-greedy)

Kryej veprimin A, observo shpërblimin R dhe gjendjen pasardhëse S'

$Q(S , A ) \leftarrow Q(S , A ) + \alpha [ R + \gamma \text{ maxa } Q(S', a) - Q(St , At ) ]$

S ← S'

Gjersa S të jetë gjendje fundore

**Figura 2.** Algoritmi Q-Learning

Algoritmi Q-Learning do të konvergjojë me probabilitet 1 në një politikë optimale nën kushtet e mëposhtme (Watkin & Dayan, 1992):

1. Shpërblimet në çdo gjendje kanë vlera të kufizuara |rn| <R.
2. Norma e të mësuarit ka vlerën $0 \leq \alpha < 1$.
3. Çdo çift gjendje-veprim vizitohet një numër te pakufizuar herësh.

### 3.4 Dilema midis eksplorimit dhe shfrytëzimit

Algoritmet e mësipërm bëjnë një vlerësim në mënyrë iterative të vlerës që ka çdo veprim që mund të kryhet në çdo gjëndje të mjedisit. Në bazë të këtij vlerësimi, në çdo moment, për një gjendje të caktuar, ekziston një veprim që e ka vlerën më të lartë krahasuar me veprimet e tjera. Nqse zgjedhim të kryejmë veprimin me vlerë më të lartë, atëherë jemi duke shfrytëzuar njohurinë e fituar deri në këtë moment. Nga ana tjetër, nqse zgjedhim një prej veprimeve që e kanë vlerën më të vogël, jemi duke eksploruar veprimet, sepse kjo gjë na lejon që të përmirësojmë vleresimin që kemi për veprimet e tjera. Dilema qëndron në faktin nëse duhet të shfrytëzojmë njohurinë e fituar deri në një moment, apo duhet të eksplorojmë me veprimet e tjera. Psh, në një moment të caktuar dimë me siguri vlerën e nje veprimi, ndërsa për veprimet e tjera kemi një shkallë pasigurie. Pasiguria ka të bëjë me faktin që të paktën njëri nga këto veprime mund të këtë vlerë më të lartë se veprimi më i mirë. Në këtë rast, do të ishte mirë të eksploronim, pasi kështu do të mësonim se cili është vërtet veprimi më i mirë. Kjo gjë mund të na japë një shpërblim të vogël gjatë kohës që jemi duke eksploruar, por në terma afatgjatë shpërblimi do të jetë më i madh, pasi do të shfrytëzojmë veprimin më të mirë, të cilin e gjetëm falë eksplorimit të kryer.

## 4. Aplikimi i algoritmeve bazuar në DT në një mjedis Markovi

### 4.1. Përshkrimi i mjedisit

Për realizimin e testeve është përdorur mjedisi “Frozen Lake” nga platforma OpenAI GYM (Brockman *et al.*, 2016).Mjedisi është një gridë me përmasa 4x4 si në figurën 3. Secili pozicion në gridë ka një prej vlerave: S që tregon pozicioni i fillimit, F që tregon se sipërfaqja është e ngrirë dhe eshte e sigurt për të kaluar, H tregon që përmban gropë dhe nuk mund të kalohet, G është pozicioni final ku duam të shkojmë

| | | | |
|---|---|---|---|
| S | F | F | F |
| F | H | F | H |
| F | F | F | H |
| H | F | F | G |

**Figura 3**. Mjedisi

Agjenti fillon nga pozicioni i fillimit dhe duhet të mësojë të shkojë në pozicionin final. Nëse kalon në një prej pozicioneve të shenuara 'H', atëherë episodi përfundon dhe rifillon nga e para tek pozicioni i fillimit. Veprimet e lejuara që agjenti mund të kryejë janë: majtas, djathtas, poshtë dhe lart. Mjedisi është stohastik, që do të thotë se veprimi i zgjedhur nuk na çon tek pozicioni i pritshëm, por me një probabilitet të caktuar devijon nga ai

pozicion. Për çdo veprim, me probabilitet 1/3 zhvendosemi tek pozicioni i pritshëm, ndërsa me probabilitet 1/3 zhvendosemi sipas dy drejtimeve pingul me drejtimin e zgjedhur. Shpërblimi që mjedisi jep është 0 për çdo pozicion të ndërmjetem, 1 për pozicionin final dhe -1 për pozicionet e shënuara me 'H'. Qëllimi i agjentit është të mësojë një politikë veprimi, që do ta çoje nga pozicioni i fillimit tek pozicioni final, duke shmangur gropat, në mënyrë që të maksimizojë shpërblimin, duke mësuar se cili është veprimi më i mirë për të kryer në çdo gjendje të këtij mjedisi.

### 4.2. Zbatimi

Për të mësuar politikën optimale të veprimit janë implementuar dy algoritmet SARSA dhe Q-Learning. Janë realizuar disa teste, me parametra të ndryshëm të koefiçentëve $\gamma$ dhe $\alpha$, për të parë sjelljen dhe konvergjencën e secilit prej algoritmeve.

#### 4.2.1 SARSA

Është përdorur politika Ɛ-greedy për zgjedhjen e veprimeve, e tillë që me probabilitet Ɛ zgjidhet një veprim i rastësishëm, ndërsa me probabilitet 1– Ɛ zgjidhet veprimi me vlerë më të madhe. Fillimisht Ɛ ka vlerë 0.5, dhe vjen duke u zvogëluar me rritjen e numrit të episodeve, për të ulur shkallën e eksplorimit. Grafikisht janë paraqitur rezultatet e ekzekutimit, për vlera të ndryshme të parametrave $\gamma$ dhe $\alpha$. Figurat 4 dhe 5 tregojnë shpërblimin total në varësi te numrit të episodeve të ekzekutuara.

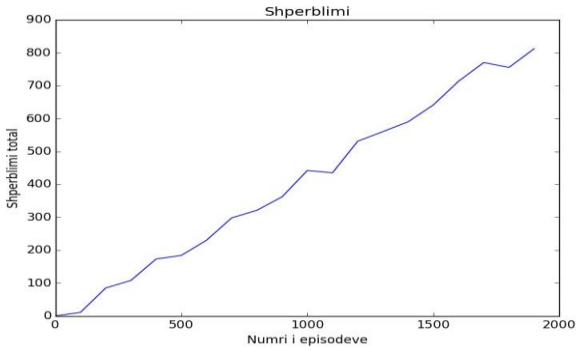


**Figura 4.** SARSA me $\alpha$= 0.1, $\gamma$ =0.9

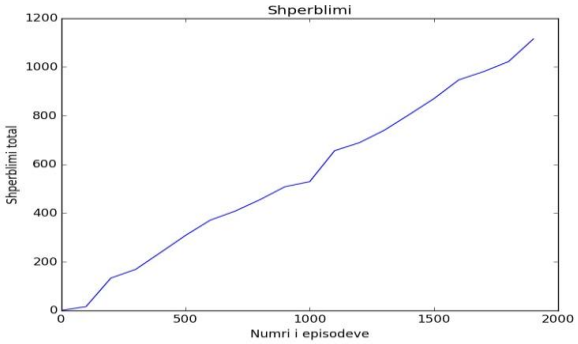


**Figura 5.** SARSA me α= 0.5, γ =0.1

### 4.2.2 Q-Learning

Është përdorur politika Ɛ-greedy për zgjedhjen e veprimeve, e tillë që me probabilitet Ɛ zgjidhet një veprim i rastësishëm, ndërsa me probabilitet 1– Ɛ zgjidhet veprimi me vlerë më të madhe. Fillimisht Ɛ ka vlerë 0.5, dhe zvogëlohet gradualisht me rritjen e numrit të episodeve, për të ulur shkallën e eksplorimit. Grafikisht janë paraqitur rezultatet e ekzekutimit, për vlera të ndryshme të parametrave γ dhe α. Figurat 6 dhe 7 tregojnë shperblimin total në varësi të numrit të episodeve të ekzekutuara.

## 5. Konkluzione

Në këtë punim u trajtuan dy algoritme të bazuar në diferencën e përkohshme, SARSA dhe Q-learning, për të mësuar një politikë optimale veprimi në një proces Markovi. Të dy këto algoritma nuk kanë nevojë të njohin paraprakisht modelin e mjedisit (probabilitetet e kalimit nga njëra gjendje tek tjera). Nga testet e kryera, rezulton se të dy algoritmet arrijnë të gjejnë një politikë optimale veprimi për mjedisin ku u testuan. Algoritmi SARSA sillet më mirë kur përdor një normë të mësuari të vogël, midis 0.1 dhe 0.5.

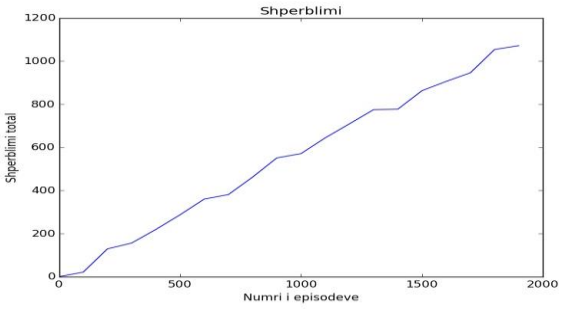


**Figura 6**. Q-Learning me α= 0.5, γ =0.9

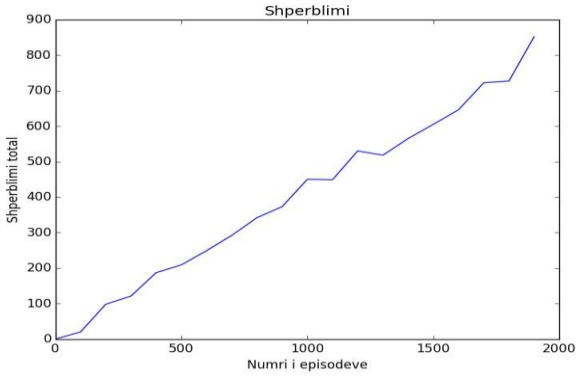


**Figura 7**. Q-Learning me $\alpha$= 0.1, $\gamma$ =0.9

Duke marrë parasysh natyrën stohastike të mjedisit, mund të mos arrihet gjendja fundore, por vihet re që arrihen të shmangen gjendjet me vlerë negative. Edhe algoritmi Q-Learning sillet më mirë për vlera të vogla të normës së të mësuarit. Në ndryshim nga SARSA, ai lejon që të mësohet politika optimale, pavaresisht politikës që jemi duke ndjekur për të zgjedhur veprimet që do të kryhen në çdo gjendje mjedisi.

**Literatura**

Richard S. Sutton., Andrew G. Barto., (1998): Reinforcement Learning: An introduction. MIT Press: 43-147

Stuart Russell., Peter Norvig., (2010): Artificial Intelligence A modern approcah. Prentice Hall: 646-852

Yuxi Li., (2017): Deep Reinforcment Learning: An Overview. ArXiv:1701.07274

Gerald Tesauro., (1993): TD-Gammon, A Self-Teaching Backgammon Program, Achieves Master-Level Play. Proceedings of the AAAI Fall Symposium on Intelligent Games: Planing and Learning: 19-23

David Silver et al., (2016): Mastering the game of Go with deep neural networks and tree search. Nature, vol 529: 484-503

Volodymyr Mnih., Koray Kavukcuoglu., David Silver., Alex Graves., Ioannis Antonoglou., Daan Wierstra., Martin Riedmiller., (2013): Playing Atari with Deep Reinforcement Learning. ArXiv:1312. 5602

Christopher Watkin., Peter Dayan., (1992): Technical Note, Q-Learning. Machine Learning, vol 8: 279-292

Satinder Singh., Tomi Jaakkola., Michael L. Littman., Csaba Szepevari., (2000): Convergence Results for Single-Step On Policy Reinforcment Learning Algorithms. Journal Machine Learning, vol 38 (3): 287-308

Greg Brockman., Vicki Cheung., Ludwig Pettersson., Jonas Schneider., John Schulman., Jie Tang., Wojciech Zaremba., (2016): OpenAI GYM. Ar-Xiv:1606. 01540