ŽILINSKÁ UNIVERZITA V ŽILINE
FAKULTA RIADENIA A INFORMATIKY

AUTOREFERÁT DIZERTAČNEJ PRÁCE

Študijný odbor: **Aplikovaná informatika**

Žilina, Apríl, 2016

autor : Ing. Michal Chovanec
vedúci: prof. Ing. Juraj Miček, PhD

ŽILINSKÁ UNIVERZITA V ŽILINE
FAKULTA RIADENIA A INFORMATIKY

Ing. Michal Chovanec

Autoreferát dizertačnej práce
Aproximácia funkcie ohodnotení v algoritmoch Q-learning neurónovou sieťou

na získanie akademického titulu philosophiae doctor (v skratke PhD.)
v študijnom programe doktorandského štúdia
aplikovaná informatika
v študijnom odbore:
9.2.9 aplikovaná informatika

Dizertačná práca bola vypracovaná v dennej forme doktorandského štúdia forme doktorandského štúdia na katedre technickej kybernetiky, Fakulte riadenia a informatiky Žilinskej univerzity v Žiline

Predkladateľ:
Ing. Michal Chovanec
Katedra technickej kybernetiky
Fakulta riadenia a informatiky
Žilinská univerzita v Žiline

Školiteľ:
prof. Ing. Juraj Miček, PhD
Katedra technickej kybernetiky
Fakulta riadenia a informatiky
Žilinská univerzita v Žiline

Oponenti:

Titul, meno a priezvisko :
Názov pracoviska :

Titul, meno a priezvisko :
Názov pracoviska :

Titul, meno a priezvisko :
Názov pracoviska :

Autoreferát bol rozoslaný dňa: .....

Obhajoba dizertačnej práce sa koná dňa ..... o ..... h. pred komisiou pre obhajobu dizertačnej práce schválenou odborovou komisiou v študijnom odbore 9.2.9 aplikovaná informatika, v študijnom programe aplikovaná informatika, vymenovanou dekanom Fakulty riadenia a informatiky Žilinskej univerzity v Žiline.

prof. Ing.Martin Klimo, PhD. predseda odborovej komisie študijného programu aplikovaná informatika v študijnom odbore 9.2.9 aplikovaná informatika Fakulta riadenia a informatiky Žilinská univerzita Univerzitná 8215/1 010 26 Žilina

Abstrakt

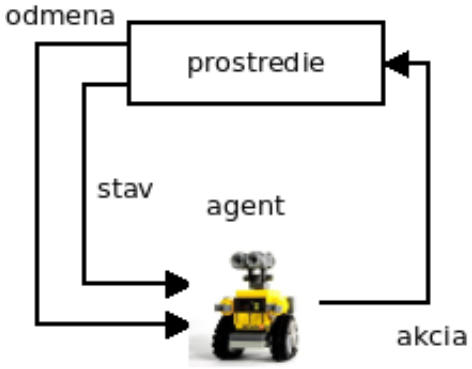
MICHAL CHOVANEC: *Aproximácia funkcie ohodnotení v algoritmoch Q-learning neurónovou sieťou*
Žilinská Univerzita v Žiline, Fakulta riadenia a informatiky, Katedra technickej kybernetiky.
Vedúci: prof. Ing. Juraj Miček, PhD
FRI ŽU v Žiline, 2016

Práca sa zaoberá aproximáciou funkcie ohodnotení konania agenta, v algoritmoch Q-learning. V priestoroch s malým počtom stavov predstavuje vhodné riešenie tabuľka. Pre prípady veľkého počtu stavov je tabuľkové riešenie ťažko vypočítateľné. Je tak nutné použiť aproximáciu. Vhodným kandidátom je neurónová sieť. Tradičné riešenie doprednej siete je však nepoužiteľné z dôvodov nemožnosti takúto sieť učiť. V práci je preto venovaný priestor neurónovej sieti bázických funkcií ktorú už je možné na daný problém trénovať iteračnými metódami.

Kapitola 1

Ciele práce

V učiacich sa systémoch založených na predkladnaní dvojíc vstup - požadovaný výstup je možné stanoviť chybu a tú vhodnými metódami minimalizovať. V prípade systému s odmeňovaním sa požaduje od výstupu výkonnej jednotky postupnosť akcií, ktoré maximalizujú celkovú odmenu. Príkladom môže byť rozhodovanie robota, ktorý má splniť cieľ pozostávajúci z niekoľkých elementárnych úkonov, ale postupnosť týchto elementárnych úkonov nie je známa - nie je teda definovaná požadovaná hodnota výstupu. Riešením tohto problému je zavedenie systému odmeňovania agenta (robota) 1.1.



Obr. 1.1: Učenie s odmeňovaním

Odmena je získaná z prostredia po vykonaní akcie. Ohodnotenie akcie v danom stave je tvorené predošlými skúsenosťami z vykonania predošlých akcií a získania odmien. Podstatou učenia je teda ohodnotenie vykonaných akcií v danom stave aby bolo možné v každom stave rozhodnúť, ktorá akcia je najlepšia - vyberá sa teda postupnosť akcií π pre ktorú je funkcia

$$\Lambda(\pi) = \sum_{n=0}^{L(\pi)} \gamma^n P_{\pi(n)}(s(n), s(n-1)) \tag{1.1}$$

Agent ako jednotka schopná konať rozhodnutia (akcie) v prostredí danom Markovovim [22] rozhodovacím procesom hľadá optimálnu stratégiu v zmysle rovnice 1.1. maximálna. Kde $\gamma \in \langle 0,1 \rangle$ je koeficient zabúdania, $P_{\pi(n)}(s(n), s(n-1)$ je odmeňovacia funkcia po prechode zo stavu $s(n-1)$ do stavu $s(n)$ vykonaním $\pi(n)$ a $L(\pi)$ je dĺžka postupnosti π

Cieľom agenta je teda nájsť optimálnu stratégiu a maximalizovať tak odmenu. Pre veľký počet stavov je hľadanie optima metódou počítania pravdepodobností prechodov medzi stavmi $P(s,s')$ ťažko vypočítateľné.

Východiskom sú napríklad algoritmy Q-learning, alebo SARSA. Tieto algoritmy počítajú ohodnotenie akcie v danom stave $Q(s(n), a(n))$, ktoré číselne vyjadruje vhodnosť danej akcie. Využitie môžu nájsť [38], [39], [40] napríklad pri plánovnaí rozhodnutí v

1. robotike
2. virtuálnych agentových systémoch

3. počítačové hry

Vo všobecnosti riešia uvedené algoritmy problémy umelej inteligencie, kedy nie je možné zostaviť trénovacie dáta v tvare vstup, požadovaný výstup a aplikácia je obmedzená na udeľovanie odmien agentovi za vykonanie zvolenej stratégie [43], [44]. Na rozdiel od evolučných algoritmov (genetické algoritmy, diferenciálna evolúcia, simulované žíhanie), kedy je daná kriteriálna funkcia, umožňujú algoritmy Q-learning, alebo SARSA postupne zlepšovať riešenie na princípe hľadania optimálnej stratégie z niekoľkých optimálnych podstratégií - už nájdené optimálne riešenie podstratégie sa nemení. V prípade evolučných algoritmov je typická zmena všetkých hľadaných parametrov. Nie sú teda vhodné na úlohy kde sa požaduje generovanie postupnosti akcií.

Pre algoritmus Q-learning je zaručená konvergencia k optimálnemu ohodnoteniu (v zmysle 1.1) [41] pre ľubovolnú metódu výberu akcií - postačuje aby každá akcia mala nenulovú pravdpodobnosť vykonania v prislúchajúcom stave. V prípade SARSA táto konvergencia nie je zaručená pre všetky metódy výberu akcií. Oba algoritmy pracujú v diskrétnom čase.

Pre problémy s rádovo stovkami stavov, ktoré sú diskrétne, môže byť fukcia $Q(s(n),a(n))$ realizovaná formou tabuľky. Konvergencia k optimálnemu riešeniu je v tomto prípade zaručená. Pre problémy kde je počet stavov veľmi veľký (tisíce a viac), alebo stavy nenadobúdajú diskrétne hodnoty je potrebné zvoliť aproximáciu tejto funkcie. Konvergencia v tomto prípade už nie je zaručená.

Prístupov ako aproximovať túto funkciu je niekoľko [31], [32], [33], [34]. Najčastejšie používané

1. Diskretizácia stavov spojitých hodnôt tabuľkou
2. Lineárna kombinácia príznakov
3. Dopredná neurónová sieť
4. Neurónová sieť bázických funkcií

Prvý spôsob predstavuje triviálne riešnie problému redukciou nekonečného počtu stavov na konečný.

Druhý spôsob spočíva v pevne definovaných príznakoch, ktoré závisia od typu problému. Tieto príznaky tvoria súbor funkcií $f_i(s(n),a(n))$. Hodnota $Q(s(n),a(n))$ je daná lineárnou kombináciou týchto príznakov. Hľadá sa teda vektor váh w pre ktorý $Q_b(s(n),a(n),w) = \sum_{i=0}^{I} w_i f_i(s(n),a(n))$ má minimálnu veľkosť chyby e, definovaná je ako $e(w) = \sum_{s,a}(Q(s(n),a(n)) - Q_b(s(n),a(n),w))^2$ Problematická zostáva voľba príznakových funkcií - ich tvar aj počet.

Tretí spôsob spočíva v použití doprednej neurónovej siete ako univerzálny aproximátor funkcie. Schopnosť aproximovať funkciu doprednou neurónovou sieťou je veľmi dobre známa aj preskúmaná. Pre úlohy Q-learning algoritmu je však nepoužiteľná [42], z dôvodov nemožnosti túto sieť naučiť doteraz dostupnými prostriedkami. Hoci existuje niekoľko prípadov kde sa učenie dá uskutočniť, vo všeobecnosti sú v protiklade dva požiadavky :

1. Učenie siete na požadovanú hodnotu
2. Generovanie požadovanej hodnoty

Sieť teda musí zároveň poskytovať správny výstup pre minulé stavy a zároveň sa učiť na súčastný stav bez toho, aby sa hodnoty z minulých stavov zmenili.

Štvrtý spôsob je využíva lineárnu kombináciu bázických funkcie. Bázické funkcie sú dané vopred, avšak ich parametre sa menia v priebehu učenia, podobne ako vektor váh lineárnej kombinácie w. Nech sú ich parametre označené ako v. Cieľom je nájsť také w a v pre ktoré chyba $e(v,w) = \sum_{s,a}(Q(s(n),a(n)) - Q_b(s(n),a(n),v,w))^2$ je minimálna. Kde $Q_b(s(n),a(n),v,w) = \sum_{i=0}^{I} w_i f_i(s(n),a(n),v_i)$.

Cieľom práce je overiť možnosti aproximácie funkcie $Q(s(n),a(n))$ uvedenými metódami. Vzľadom na už prebehnutý výskum a problémy dopredných neurónových sieti, sa problematika sústredí najmä na hľadanie vhodných bázických funkcií. Práve v tejto oblasti je venovaný výskumu najväčší priestor. Tieto funkcie by mali byť volené tak, aby zmena parametrov v_i jednej funkcie, neovplivnila výsledok inde ako pre žiadané $s(n)$ a $a(n)$. Použité riešnie je potom možné využiť vo veľkých stavových priestorov, kde možnosti použiť tabuľku zlyhávajú z dôvodov

1. Veľké pamäťové nároky

2. Nutnosť navštíviť a správne spočítať Q pre všetky $s(n)$, $a(n)$

Prvý problém nepredstavuje pre súčasné počítače až tak veľký nedostatok tabuľkového riešenia. Horšia je situácia v prípade vypĺňania korektných hodnôt v tabuľke. Práve rekurentnou povahou algoritmov Q-learning a SARSA je časovo veľmi náročné vyplniť tieto hodnoty - mnohonásobne treba navštíviť všetky stavy a vykonať v nich všetky akcie. Práve to je primárny dôvod aproximovať funkciu $Q(s(n),a(n))$.

1.1 Q-learning algoritmus

Q-learning algoritmus je definovaný pre časovo diskrétne systémy. Agent ktorý prechádza stavový priestor vykonaním niektorej z vopred daných akcií získava za tieto prechody odmeny. Cieľom algoritmu je ohodnotiť všetky akcie v jednotlivých stavoch, tak aby bol dosiahnutý ustálený stav a v každom stave bolo možno vybrať akciu prinášajúcu najväčšiu odmenu, v zmysle s 1.1.

1.1.1 Definícia algoritmu

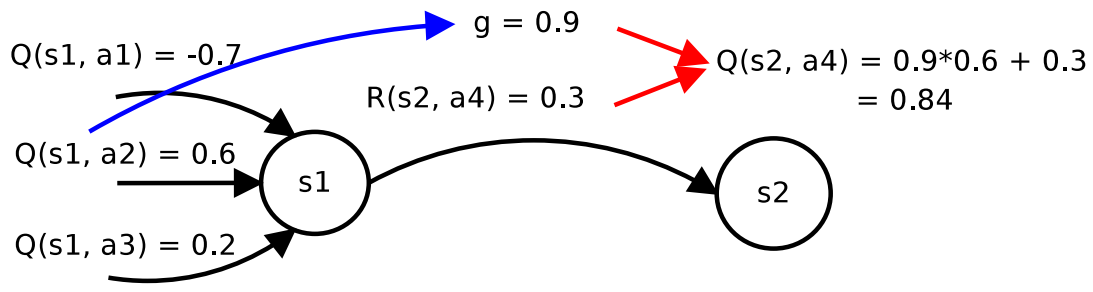
Autorom Q-learning algoritmu je Christopher J.C.H. Watkins, v roku 1992 publikoval článok kde tento algoritmus predstavil [10] a niekoľko ďalších vysvetlení tohto algoritmu je možné nájsť v [11] alebo [12]. Dôkazy o konvergencií k otimálnemu riešeniu (v zmysle s 1.1) sú k dispozícií [13], [14], [15], [16].
Je daná odmeňovacia funkcia $R(s(n),a(n))$, ktorá vyjadruje okamžité ohodnotenie konania agenta v stave $s(n)$ vykonaním akcie $a(n)$. V reálnych aplikáciach táto funkcia nadobúda takmer v každom $s(n)$ a $a(n)$ hodnotu 0. Pre správnu funkciu algoritmu, musí byť aspoň jedna hodnota nenulová - napr. ohodnotenie dosiahnutia cieľového stavu (samotná existencia cieľového stavu však pre algoritmus nie je potrebná).

Funkcia ohodnotení je definovaná ako

$$Q_n(s(n),a(n)) = R(s(n),a(n)) + \gamma \max_{a(n-1)\in\mathbb{A}} Q_{n-1}(s(n-1),a(n-1)) \tag{1.2}$$

- $R(s(n),a(n))$ je odmeňovacia funkcia
- $Q_{n-1}(s(n-1),a(n-1))$ je funkcia ohodnotení v stave $s(n-1)$ pre akciu $a(n-1)$
- γ je odmeňovacia konštanta a platí $\gamma \in (0,1)$.

Funkcia 1.2 definuje ohodnotenie akcií vo všetkých stavoch t.j. agent, ktorý sa dostal do stavu $s(n)$ vykonaním akcie $a(n)$ zo stavu $s(n-1)$ získal odmenu $R(s(n),a(n))$ a zlomok najväčšieho možného ohodnotenia, ktoré mohol získať dostaním sa do stavu $s(n-1)$, situáciu ilustruje obrázok 1.2.



Obr. 1.2: Ilustrácia funkcie ohodnotení, pre $\gamma = 0.9$

Je potrebné poznamenať, že práve časť $\max_{a(n-1)\in\mathbb{A}} Q_{n-1}(s(n-1),a(n-1)$ zabezpečuje nezávislosť konvergencie k optimu bez ohľadu voľby stratégie výberu akcie - postačuje, aby každá akcia, v každom stave mala nenulovú pravdepodobnosť vykonania. Určitým variantom je algoritmus SARSA [17]

$$\begin{aligned} Q_n(s(n),a(n)) = \\ (1-\alpha)Q_{n-1}(s(n),a(n)) + \\ \alpha(R(s(n),a(n)) + \gamma Q_{n-1}(s(n-1),a(n-1))) \end{aligned} \tag{1.3}$$

kde $\alpha \in (0,1)$, hodnota $Q_n(s(n),a(n))$ sa teda ustáli na strednej hodnote a závisí od stratégie výberu akcií. Q-learning teda vychádza z toho, čo najlepšie sa mohlo stať a SARSA z toho čo sa naozaj stalo.

Kapitola 2

Navrhnuté bázické funkcie neurónovej siete pre aproximáciu

Vhodnú funkciu je možné zmenou parametrov upraviť do tvaru, aby pre zvolený vstup $I_0(n)$ dosahovala požadovanú hodnotu a postupným zväčšovaním vzdialenosti $\mid I_0(n) - I_i(n) \mid$ klesala jej hodnota k nule.

Najjednoduhším príkladom takýchto funkcií je

$$f_j(X(n)) = \begin{cases} k_j & ak\ X(n) = X_0^j \\ 0 & inak \end{cases} \tag{2.1}$$

kde k_j je hodnota požadovaná v bode X_0^j. Výstupom siete potom je

$$y(X) = \sum_{j=1} f_j(X(n)) \tag{2.2}$$

Z charakteru Q-learning algoritmu majú hodnoty $Q(s(n),a(n))$ charakter postupne klesajúcich hodnôt. Je teda potrebné vybrať iné funkcie.

Nasledujú preto definície funkcií s ktorými boli urobené experimenty.

Dané sú bázické funkcie $f_j^x(s(n),a(n))$, kde x je typ bázickej funkcie. Požadovaná hodnota $Q^x(s(n),a(n))$ je potom lineárnou kombináciou týchto funkcií typu x.

Z charakteru Q-learning algoritmu 1.2 je možné určiť požiadavky na tieto funkcie :

1. predpis 1.2 je tvorený klesajúcou exponenciálou - podobný charakter by mala mať aj bázická funkcia
2. existencia jedného globálneho maxima a zmenou parametrov určovať polohu tohto bodu
3. možnosť ľubovolne meniť strmosť funkcie v okolí maxima
4. funkcia by mala byť zhora aj z dola ohraničená

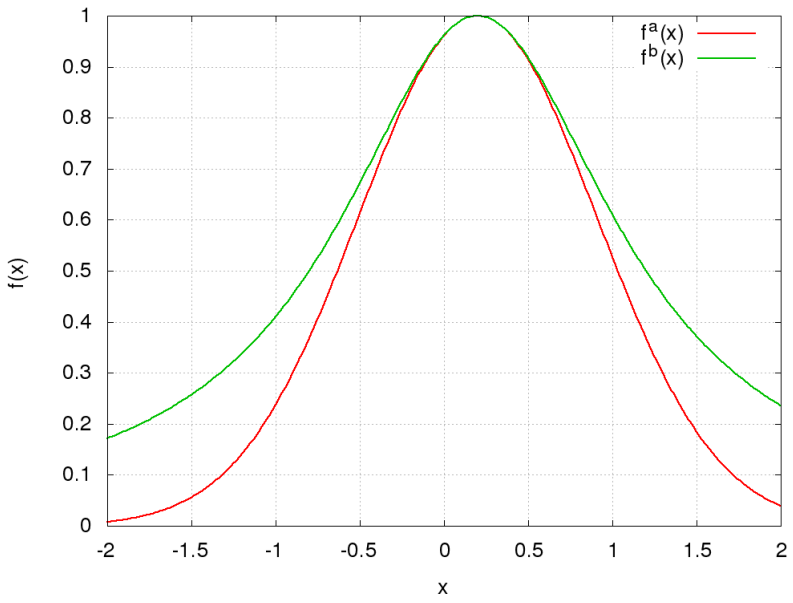
Cieľom je mať možnosť nezávisle nastaviť maximá funkcií do oblastí, ktoré zodpovedajú nenulovím hodnotám $R(s(n),a(n))$ - bod 2. Ak ohodnotenie spĺňa podmienku najlepšej možnej akcie v danom stave, dá sa očakávať že bude mať menšiu strmosť, naopak, ak funkcia popisuje bod kde $R(s(n),a(n))$ dosahuje malé hodnoty (obvykle záporné), bude požadovaná vysoká strmosť tejto funkcie - obe požiadavky sú zhrnuté v bode 3. Bod 4 umožňuje rozumne ohraničiť rozsah funkcie.

Niektoré tvary bázických funkcií ktoré možno uvažovať pre problém aproximácie

$$f_j^1(s(n),a(n)) = e^{-\sum_{i=1}^{n_s} \beta_{aji}(n)(s_i(n)-\alpha_{aji}(n))^2} \tag{2.3}$$

$$f_j^2(s(n),a(n)) = \frac{1}{1+\sum_{i=1}^{n_s} \beta_{aji}(n)(s_i(n)-\alpha_{aji}(n))^2} \tag{2.4}$$

$$f_j^3(s(n),a(n)) = e^{-\sum_{i=1}^{n_s} \beta_{aji}(n)|s_i(n)-\alpha_{aji}(n)|} \tag{2.5}$$



Obr. 2.1: Znázornenie priebehov bázických fukcií

kde
$\alpha_{aji}(n) \in \langle -1,1 \rangle$ určuje polohu maxima funkcie
$\beta_{aji}(n) \in (0,\infty)$ určuje strmosť funkcie.
Ich priebehy pre prvé dve uvedené sú na obrázku 2.1.
Pre symetrické prechody medzi stavmi ich možno zjednodušiť na

$$f_j^1(s(n),a(n)) = e^{-\beta_{aj}\sum_{i=1}^{n_s}(s_i(n)-\alpha_{aji})^2} \tag{2.6}$$

$$f_j^2(s(n),a(n)) = \frac{1}{1+\beta_{aj}\sum_{i=1}^{n_s}(s_i(n)-\alpha_{aji})^2} \tag{2.7}$$

$$f_j^3((n)s,a(n)) = e^{-\beta_{aj}\sum_{i=1}^{n_s}|s_i(n)-\alpha_{aji}(n)|} \tag{2.8}$$

Aproximovaná funkcia ohodnotení pre l bázických funkcií je potom

$$Q^x(s(n),a(n)) = \sum_{j=1}^{l} w(n)_j^x f_j^x(s(n),a(n)) \tag{2.9}$$

kde $w(n)_j^x$ sú váhy bázických funkcií.
Je teda potrebné stanoviť celkovo 3 sady parametrov : α β w.

2.0.2 Určenie parametrov α

Parameter α určuje posunutie maxima funkcie a postupuje sa podobne ako v prípade **??**. Treba zohľadniť fakt, že pre konečný výsledok je dôležité pokryť všetky oblasti s nenulovým $R(s(n),a(n))$, vrchol krivky bude ležať nad nad bodom $[s(n),a(n)]$.
Zmena parametrov α prebieha v piatich krokoch.

- na začiatku sa zvolia $\alpha_{jia}(n)$ náhodne, ze $\langle -1,1 \rangle$
- spočítajú sa vzdialenosti od predloženého vstupu $d_{ja}(n) =\mid s(n) - \alpha_{ja}(n) \mid$

- nájde sa také ka kde pre $\forall j : d_{ka}(n) \leq d_{ja}(n)$
- spočíta sa krok učenia $\eta'_a(n) = \eta_1 \mid Q_r(s(n),a(n)) \mid$
- upravia sa parametre $\alpha_{aki}(n+1) = (1-\eta')\alpha_{aki}(n) + \eta' s_i(n)$

kde
$Q_r(s(n),a(n))$ je požadovaný výstup
η_1 je konštanta učenia

Krok učenia teda závisí od veľkosti požadovanej hodnoty, tým sa zabezpečí aby maximum krivky naozaj ležalo nad bodom $[s(n),a(n)]$.

2.0.3 Určenie parametrov β

Parameter β určuje strmosť krivky. Ak boli k dizpozicií naraz všetky požadované výstupy, bolo by možné spočítat tento parameter z rozptylu. Požadované hodnoty však prichádzajú postupne, strmosť krivky sa preto upravuje priebežne, podľa toho či požadovaná hodnota leží nad, alebo pod krivkou.

- stanoví sa chyba $e(n) = Q_r(s(n),a(n)) - Q(s(n),a(n))$
- pre každú bázickú funkciu $\beta_{ja}(n+1) = \beta_{ja}(n) + \eta_2 e(n) w_{ja}(n)$
- skontroluje sa $\beta_{ja}(n) \in (0,\infty)$

kde
$Q_r(s(n),a(n))$ je požadovaný výstup
η_2 je konštanta učenia

2.0.4 Určenie váhových parametrov w

Nakoniec sa gradientovou metódou určia váhové paramete. Pre presné riešenie by bolo možné použiť metódu nejmenších štvorcov, tá je však pre veľký počet bázcikých funkcií ťažko vypočítateľná. Zmena parametrov je potom daná nasledujúcim postupom

- stanoví sa chyba $e(n) = Q_r(s(n),a(n)) - Q(s(n),a(n))$
- pre každé $w_{ja} : w_{ja}(n+1) = w_{ja}(n) + \eta_3 e(n) y_j(n)$
- skontroluje sa $w_{ja}(n) \in (-r,r)$

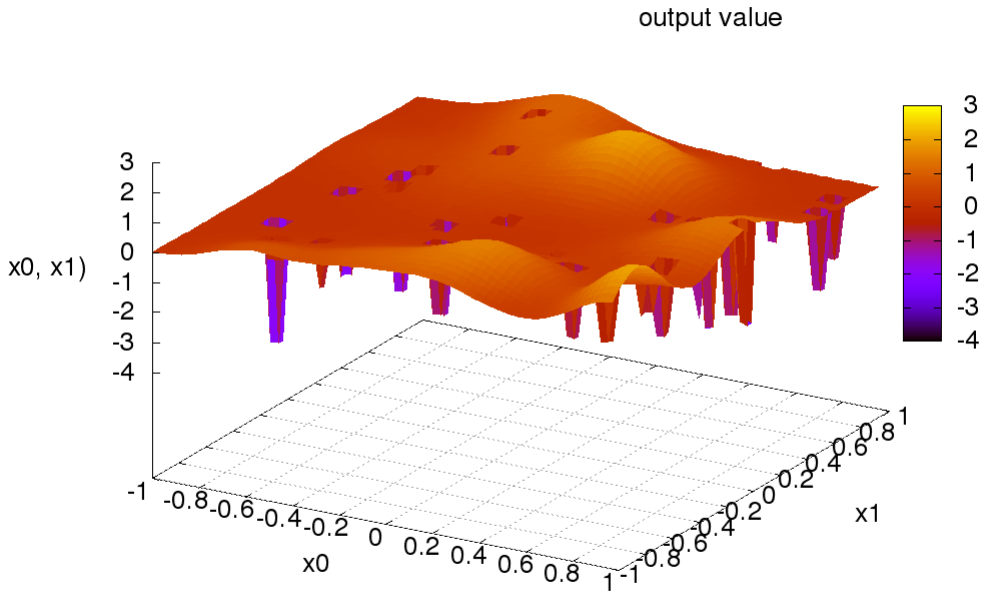
kde
η_3 je konštanta učenia
r je maximálny rozsah váh

2.0.5 Hybridný variant

Ak by funkcia $R(s(n),a(n))$ mala len jednu kladnú hodnotu a ostatné by boli nulové, aproximáciu $Q(s(n),a(n))$ by veľmi dobre popísala Gaussova krivka 2.8. Ak by funkcia $R(s(n),a(n))$ mala len záporné hodnoty a ostatné by boli nulové, funkcia $Q(s(n),a(n))$ by s ohliadnutím na 1.2 si boli rovné. Vo funkcí $Q(s(n),a(n))$ by sa tak objavilo niekoľko záporných hodnôt, ostro ohraničených.

Vyjduc z týchto úvah, je možné skombinovať výhody oboch : Gaussova krivka ktorá dokáže pokryť nenulovými hodnotami celý definyčný obor a má možnosť tak šíriť hodnoty Q na ďalšie stavy a funkcie 2.1. Funkcia 2.1 predstavuje vlastne tabuľku, ktorá nadobúda nenulové hodnoty vo vybraných bodoch - tvorí tak adaptívnu tabuľku.

Je teda možné skombinovať funkciu 2.1 s niektorou z 2.8, čo vedie na vzťahy



Obr. 2.2: Znázornenie predmetnej funkcie

$$P_i(s(n),a(n)) = \begin{cases} r_{ai} & if\ s(n) = \alpha_i^1 \\ 0 & inak \end{cases} \tag{2.10}$$

$$H_j(s(n),a(n)) = w_{aj} e^{-\beta_{aj} \sum_{i=1}^{ns} (s_i(n) - \alpha_{aji}^2)^2} \tag{2.11}$$

$$Q(s(n),a(n)) = \sum_{i=1}^{I} P_i(s(n),a(n)) + \sum_{j=1}^{J} H_j(s(n),a(n)) \tag{2.12}$$

kde
α_j^1 sú oblasti kde $H_j(s(n))$ nadobúda nenulové hodnoty
α_j^2 sú oblasti pre ktoré $f_j(s(n),a(n))$ nadobúda maximum
r_{ai} je hodnota zápornej odmeny $R(s(n),a(n))$
w_{aj} je váha a zobovedá veľkosti maxima resp. minima pre fukciu
β_{aj} je strmosť, a platí $\beta > 0$
I a J sú počty bázických funkcií

Označenia P a H vznikli z tvaru funkcií : peak a hill. Funkcia bude na ďalších grafoch označená ako Gauss + AT : kombinácia Gaussovej krivky a adaptívnej tabuľky. Mechanizmus učenia zostáva rovnaký ako pre bázické funkcie v predošlej časti. Ukážka priebehu funkcie pre dve premenné je na obrázku 2.2. Počet funkcií $P_i(s(n),a(n))$ bol zvolený 30 a počet funkcií $P_i(s(n),a(n))$ 20. Pre názornosť boli parametre r_{ai} zvolené záporné a parametre β_{aj} kladné.

Funkcia predstavuje nový tvar bázických funkcií pre aproximovanie funkcie ohodnotení $Q(s(n),a(n))$.

Kapitola 3

Experimentálna časť

3.1 Ciele experimentu

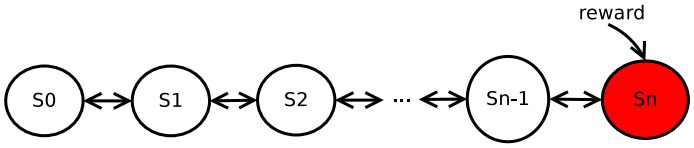
V oblasti Q-learning algoritmoch je možné pozorovať dva hlavné smery výskumu

- aproximácia funkcie ohodnotení [31] [32] [33] [34]
- spôsob výberu akcie [35] [36] [37]

Obe majú široké pole diskusií v snahe vyriešit niekoľko hlavných problémov Q-learning algoritmu a to najmä
:

- veľký počet prechodov medzi stavmi
- malá zmena vo výpočte $Q(s(n),a(n))$ môže spôsobiť veľké zmeny v stratégií.

Cieľom práce je na danej množine odmeňovacích funkcií $R(s(n),a(n))$ overiť možnosti aproximácie $Q(s(n),a(n))$. Prvým intuitivným spôsobom bola snaha aproximovať predmentnú funkciu doprednými neurónovými sieťami. Principiálne tomu nič nebráni, problém je ale nedokonalý algoritmus učenia, a to, že sa vplyvom rekurentnej povahy Q-learning algoritmu pokúša neurónová sieť zároveň predikovať správnu hodnotu a zároveň učiť na požadovanú hodnotu.



Obr. 3.1: Ilustrácia postupného nabaľovania chyby

Postupne sa tak v sieti nabaľuje chyba. Tento problém ilustruje 3.1. Je daná postupnosť stavov a každom okrem východzieho a cieľového sú dve akcie. Odmena $R(s,a)$ je všade nulová, len po dosiahnutí cieľového stavu je rovná kladnej hodnote.

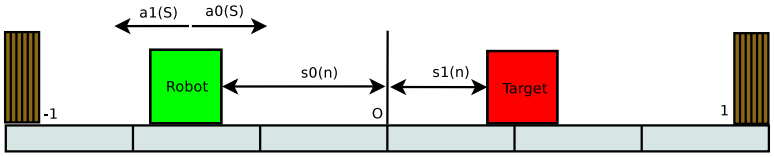
Pre korektné vyplnenie hodnôt v s_{n-1} sa vyžaduje korektá hodnota v s_n

$$Q(s(1),a(1)) = R(s(1),a(1)) + \gamma \max_{a(0)\in\mathbb{A}} Q(s(0),a(0))$$
$$Q(s(2),a(2)) = R(s(2),a(2)) + \gamma \max_{a(1)\in\mathbb{A}} Q(s(1),a(1))$$
$$\dots$$

V prípade doprednej siete učenej algoritmom Backpropagation, zmena hodnoty v jednom bode $Q(s(n),a(n))$ spôsobí zmenu vo všetkých ostatných hodnotách a nikde nie je zaručené, že k správnej hodnote - v určitom štádiu učenia sa tak môže zdať, že hodnoty koretkne kovergujú, a inom sa môžu vzďaľovať. Práve preto sa pre klasické úlohy rozpoznávania predkladajú sieti vzory v náhodonom poradí a v mnohých opakovaniach. Vzory a požadované výstupy sú však nezávislé.

3.1.1 Divergencia riešenia

Tento efekt divergencie bol pozrovaný nie len vyššie uvedenými autormi, ale aj experimentálne overený v tejto práci. Usporiadanie experimentu je na obrázku 3.2. Robot má dve akcie, pohyb o pevne zvolený krkok vľavo alebo vpravo. Úlohou je dostať sa do cieľa, ktorý môže byť umiestnený kdekoľvek. Pre jednoduchosť bol vybraný dvoj rozmerný stavový priestor z rozsahu $s \in \langle -1, 1 \rangle$. Stav systému charakterizovaný vektorom s je poloha robota voči počiatku a poloha cieľa voči počiatku, takýto systém je aj dobre graficky znázorniteľný.



Obr. 3.2: Schéma experimentu pre doprednú neurónovú sieť

Z ostatných parametrov ktoré boli použité pre beh experimentu :

- počet iterácií = 10000000
- delenie stavového priestoru = 1/8.0
- $\gamma = 0.7$
- neurónová sieť :
  - počet skrytých vrstiev = 2
  - počet neurónov v skrytých vrstvách = 10
  - rozsha váh = 4.0
  - krok učenia $\eta = 0.001$

Najskôr bolo určené riešenie použitím tabuľky (ktoré bolo pre malý počet stavov možné spočítať). Najdôležitejší výstup je výber korektnej akcie, kde $+1$ znamená jeden smer a -1 smer opačný. Veľmi ľahko sa dá očakávať ostré rozdelenie stavového priestoru po diagonále : ak je robot naľavo od cieľa musí sa pohybovať doprava a naopak. Výsledok je na obrázku 3.5. Pre úplnosť, obrázok 3.6 znázorňuje hodnoty $\max_{a(n-1)\in\mathbb{A}} Q(s,a)$. Opäť sa dá ľahko očakávať že pre najmenšiu vzdialenosť bude táto hodnota najväčšia - hodnoty na diagonále.

Jedno z najlepších riešení dosiahnuté doprednou neurónovou sieťou učenou Backpropagation algoritmom je na obrázkoch 3.5 a 3.6.

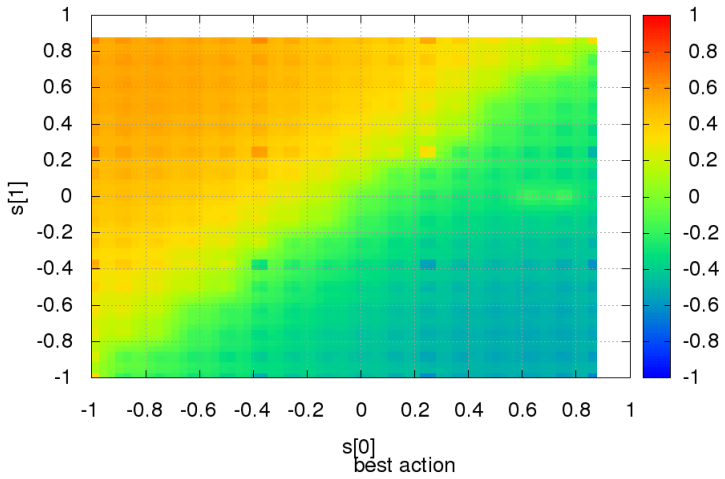
Napriek jednoduchej úlohe, nie je možné povedať že sieť úspešne aproximuje tento problém. Porovnaním výstupov najlepších akcií je možné vidieť určitý náznak podobnosti, ktorý je však vzhľadom na irelevantnosť úlohy bezpredmentný a dosahuje priveľkú chybu, najmä ak sa robot už blížil k cieľu.

Najlepšie výsledky dosahovala dopredná neurónová sieť s novo zavedeným modelom neurónu v tvare

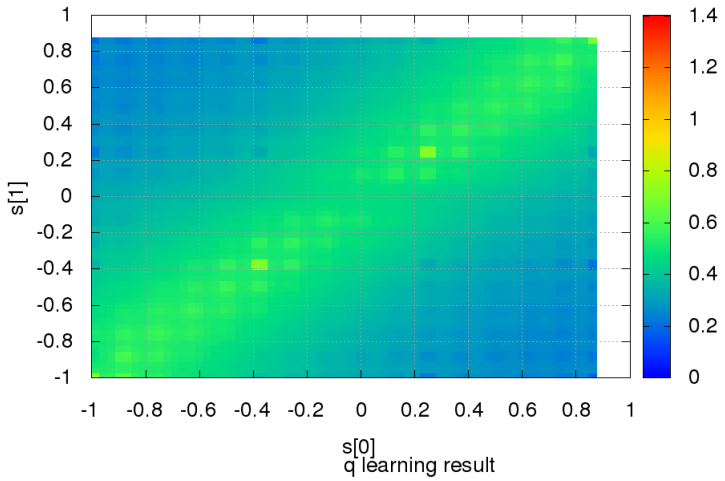
$$y(x(n)) = tanh(\sum_{i=0}^{N-1} w_i x_i(n) + \sum_{j=0}^{N-1}\sum_{i=0}^{j} v_{ij} x_i(n) x_j(n)) \tag{3.1}$$

kde x(n) je vstup do neurónu
w je vektor váh
v je matica váh

Takto definovaný model neurónu umožňuje okrem bežných funkcí McCuloh-Pittsovho neurónu aj násobiť prvky vstupného vektora (časť $v_{ij}x_i(n)x_j(n)$). Dôsledkom toho je realizácia zložitých funkcií len s použitím jednej skrytej vrstvy - výrazne sa tak zjednoduší učenie. Medzi typické funkcie ktoré sa s McCuloh-Pittsovim neurónom a jendou skytou vrstvou ťažko realizujú, možno uviesť napr : Fourierova transformácia, zmiešavanie signálov, riadenie toku dát na základe inej časti dát. Najmä posledne uvedená zvyšuje stupeň abstrakcie, kde neurónová sieť neaproximuje len jeden naučený druh funkcie, ale môže aproximovať viac, úplne rozdielných a medzi nimi vyberať. Uvedený model bol doteraz nepublikovaný v inej literatúre.



Obr. 3.3: Najlepšia akcia pre riešenie s tabuľkou



Obr. 3.4: Hodnoty $\max_{a(n-1)\in\mathbb{A}} Q(s,a)$ pre riešenie s tabuľkou

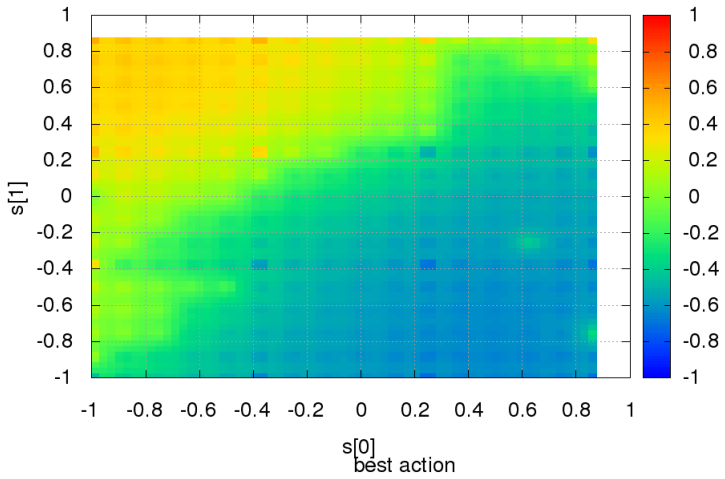
3.2 Riešenie aproximácie

Uvedení autori najčastejšie používajú tzv. príznaky (features) na aproximovanie $Q(s(n),a(n))$

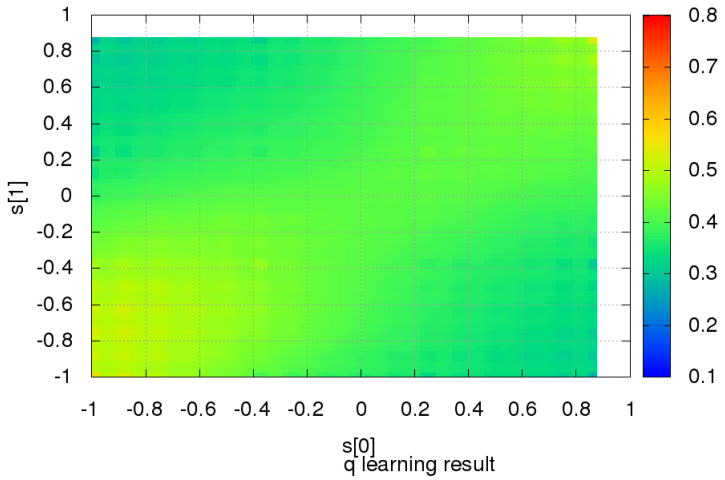
$$Q(s(n),a(n)) = \sum_{j=1} w_j g_j(s(n),a(n)) \tag{3.2}$$

kde $g_j(s(n),a(n))$ sú funkcie príznakov, ktorých je konečný počet a w_j predstavuje váhy ich lineárnej kombinácie.

Príznaky sú funkcie, ktoré sú pevne zvolené a závisia od typu úlohy. Práve to predstavuje najväčší nedostatok. Cieľom navrhovaného experimentu je využiť príznaky ktorých parametre sa menia - bázické funkcie. Vzniká tak akýsi hybrid medzi neurónovou sieťou a lineárnou kombináciou pevne zvolených príznakov.



Obr. 3.5: Najlepšia akcia pre riešenie s neurónovou sieťou



Obr. 3.6: Hodnoty $\max_{a(n-1)\in\mathbb{A}} Q(s,a)$ pre riešenie s neurónovou sieťou

Z ohľadom na minimalizovanie vplyvu zmeny parametrov j-teho príznaku alebo váhy w_j na ostatné príznaky a váhy, je potrebné, aby ich bolo možné nastavovať nezávislé - aby vhodná séria príznakov pokryla svoju podmnožinu stavového priestoru. Toto je možné dosiahnuť ortogonalitou príznakou, stráca sa však možnosť generovať funkciu ako je lineárna kombinácia týchto ortogonálnych funkcií. Vhodným kompromisom sú preto funkcie uvedené v 2.5, alebo funkcia 2.0.5.

3.3 Návrh experimentu

V niekoľkých bodoch je možné postup určiť ako

- výber funkcií $R(s(n),a(n))$

- určenie presného riešenia, použitím tabuľky s veľkým počtom prvkov
- voľba aproximačnej metódy
- pre každú $R(s(n),a(n))$ spočítať niekoľko nezávislych behov
- výsledky porovnať s presným riešením, overiť a zosumarizovať

Funkcie $R(s(n),a(n))$ budu vybrané tak aby boli riedke a plne sa využil Q-learing - okamžité odmeny sú známe len v malom počte prípadov. Postupne sa obmenia pre rôzne počty nenulových prvkov.

Presné riešenie, aby bolo možné spočítať bude mať niekoľko tisíc diskrétnych stavov. Pre jednoduchosť, bude v každom stave rovnaká a presne definovaná množina akcií.

Vyberie sa niekoľko aproximačných metód, ktoré sa použijí na spočítanie $Q(s(n),a(n))$. Tu je nevyhnutné upozorniť na častú metodickú chybu : aj keď je možné $Q(s(n),a(n))$ spočítať presne, nesmie byť toto presne riešenie použité na stanovenie približného riešenia. Príkladom je dopredná neurónová sieť, ktorá sa dá veľmi ľahko natrénovať ak je množina požadovaných výstupov vopred známa. V prípade Q-learning algoritmu sa ale požadované hodnoty spočítavajú rekuretne, až počas behu.

Kedže voľba niektorých počiatočných parametrov aproximačných metód je náhodná, je nevyhnutné spočítať niekoľko nezávislých behov a overť tak rozptyl, minimálnu, maximálnu a priemernu chybu.

Aby sa dalo kvalitatívne ohodnotiť použité riešenie, je nutné urobiť veľký počet experimentov. Aby bolo možné ľahko graficky znázorniť výsledok, bude stavový priestor dvojrozmerný a platí $s(n) \in \langle -1,1 \rangle$. Agent si bude vyberať z pevne danej množiny akcií a bude sa tak v tomto priestore môcť pohybovať a to :
$\mathbb{A} = [[0,1],[0,-1],[1,0],[-1,0],[1,-1],[1,1],[-1,-1],[-1,1]]$
prostredie umožní zmenu stavu vykonaním akcie $a(n) \in \mathbb{A}$, a to podľa

$$s(n+1) = s(n) + a(n)dt \tag{3.3}$$

Jednotlivé funkcie $R^k(s(n),a(n))$ predstavujú mapy odmien v ktorých sa agent pohybuje. Pre zjednodušenie bude platiť, že nezáleží ktorou akciou sa agent dostal do daného stavu - funkcia bude mať teda tvar $R^k(s(n))$ a predstavuje teda odmenu za to, že sa agent dostal na nejaké miesto.

Ako metódy aprximácie je zvolených 6 rôznych funkcií.

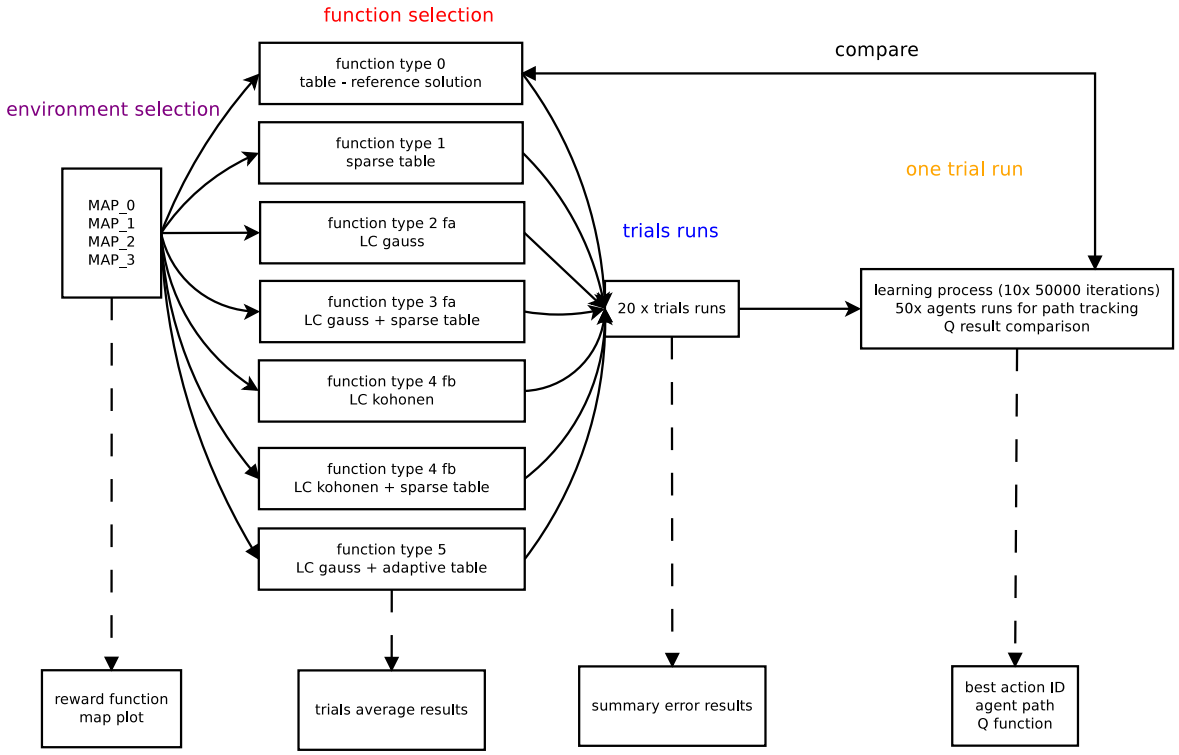
1. riedka tabuľka
2. Gaussova krivka $f_j^1(s(n),a(n))$ 2.8
3. Gaussova krivka $f_j^1(s(n),a(n))$ kombinovaná s riedkou tabuľkou
4. Modifikácia Kohonenovej neurónovej siete $f_j^2(s(n),a(n))$
5. Modifikácia Kohonenovej neurónovej siete $f_j^2(s(n),a(n))$ s riedkou tabuľkou
6. Guassova krivka a adaptívna tabuľka 2.0.5

Pre každú z nich prebehne 20 trialov aby bolo možné urobiť štatistické vyhodnotenie. V každom trialy prebehne 10*50000 učiacich interácií aby bolo možné v 10 tich krokoch sledovať priebeh učenia. Na konci prebehne 50 behov agentov z náhodných východzich stavov aby bolo možné sledovať ich cestu stavovým priestorom. Spolu teda prebehne 560 nezávislých experimentov a celkovo 280mil. behu algortimu.

Súhrnná schéma behu experimentov je na obrázku 3.7. Plné šípky predstavujú prepojenie úrovni metodológie. Čiarkované šípky znázorňujú výstupy v jednotlivých úrovniach. Presné riešenie je použité na porovnanie výslednej chyby.

- 50000 iterácií učenia
- rozmer s je $n_s = 2$, rozmer a je $n_a = 2$
- predpis funkcie ohodnotení

$$\begin{aligned} &Q(s(n),a(n)) = \\ &\alpha Q(s(n-1),a(n-1)) \\ &(1-\alpha)(R(s(n),a(n)) + \gamma \max_{a(n-1)\in\mathbb{A}} Q(s(n-1),a(n-1)) \end{aligned}$$



Obr. 3.7: Schéma experimentu

- $R(s(n),a(n)) \in \langle -1,1 \rangle$ náhodná mapa s 1 cieľovým stavom
- $\gamma = 0.98$ a $\alpha = 0.7$
- hustota referenčného riešenia = 1/32 (4096 stavov)
- počet akcií v každom stave = 8
- hustota riedkej tabuľky = 1/8 (1:16 pomer)
- počet bázických funkcií $l = 64$
- rozsah parametrov
  - $\alpha_{ja}(n) \in \langle -1,1 \rangle$
  - $\beta_{ja}(n) \in \langle 0,200 \rangle$
  - $w_{ja}(n) \in \langle -4,4 \rangle$

$Q_{rt}(s(n),a(n))$ referenčná funkcia Q (funkcia 0), kde $t \in \langle 0,19 \rangle$ je číslo trialu $Q_{jt}(s(n),a(n))$ testované funkcie Q a $j \in \langle 1,5 \rangle$.

Celková chyba behu trialu t je

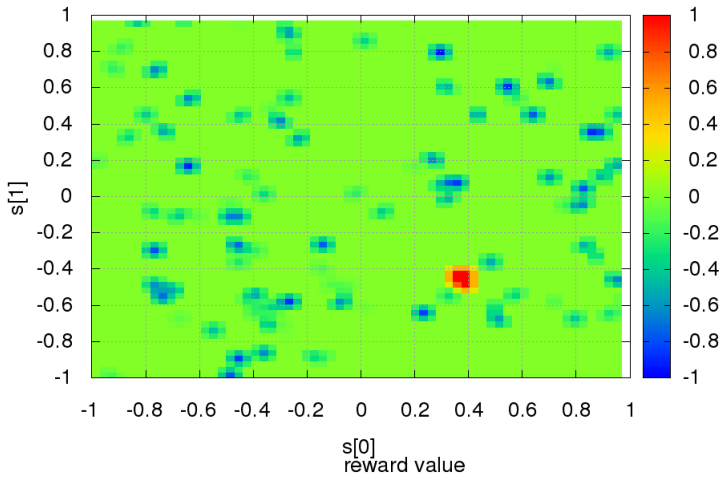
$$e_{jt} = \sum_{s,a} (Q_{rt}(s,a) - Q_{jt}(s,a))^2$$

priemerná, minimálna, maximálna chyba a smerodatná odchylka

$$\bar{a}_j = \frac{1}{20}\sum_t e_{jt}$$
$$e_j^{min} = \min_t e_{jt}$$
$$e_j^{max} = \max_t e_{jt}$$
$$\sigma_j{}^2 = \frac{1}{20}\sum_t (\bar{a}_j - e_{jt})^2$$

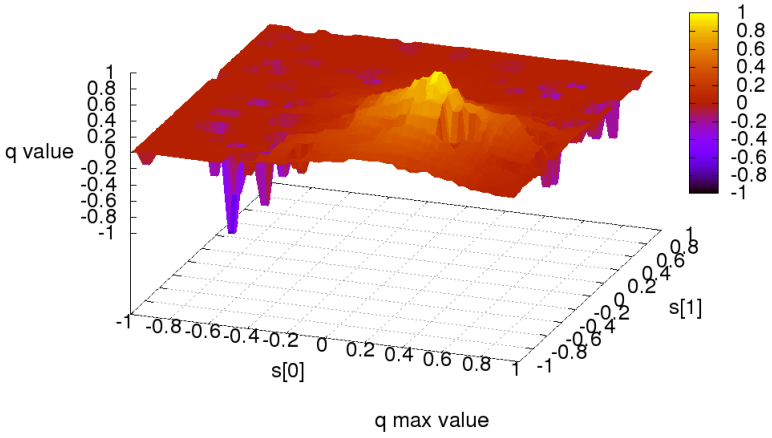
3.4 Výsledky experimentu

Experiment bol spočítaný pre 4 rôzne mapy - funkcie $R(s(n),a(n))$. Je potrebné poznamenať, že takto navrhnuté prostredie umožňuje agentovi aby nastala každý možný stav - to komplikuje možnosť redukcie počtu stavov. Ukážka mapy č. 2 je na obrázku 3.8. Pričom ako bolo v predošlej časti povedané, platí $R^k(s(n),a(n)) = R^k(s(n))$, t.j. odmena je rovnaká v každom prechode vedúcom do rovnakého stavu.

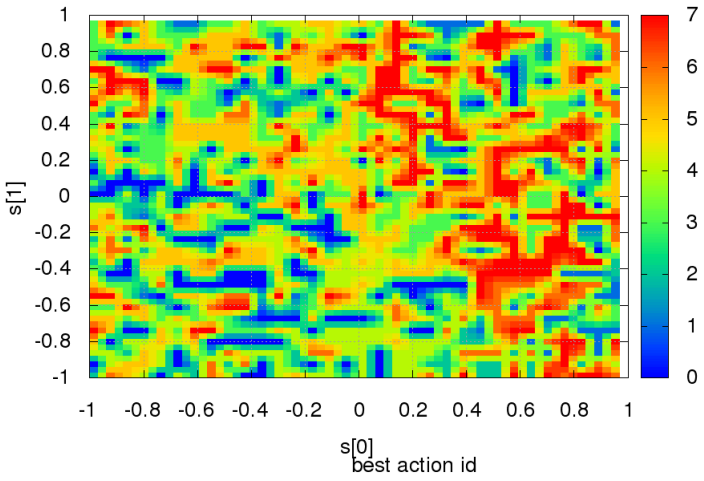


Obr. 3.8: Odmeňovacia funkcia R(s(n), a(n)), mapa 2

Pre riešenie Q funkcie s použitím tabuľky, ktoré bude vzhľadom na podmienky experimentu presným riešim je graf funkcie $\max_{a(n)\in\mathbb{A}} Q_n(s(n),a(n))$ na obrázku 3.9. Je možné ľahko pozorovať maximum v oblasti jediného kladného $R(s(n),a(n))$. Od tohto maxima sa šíria hodnoty na celý definičný obor podľa vzťahu 1.2. Ďalej je možné pozrovať záporné hodnoty, ktoré sa nešíria ďalej - predstavujú oblasti kde $R(s(n),a(n))$ nadobúda tiež záporné hodnoty. Na základe známeho $Q(s(n),a(n))$ je možné zostaviť mapu ktorú akciu číslovanú 0 až 7 má agent zvoliť - mapa najlepších akcií v danom stave je znázornená na obrázku 3.10.

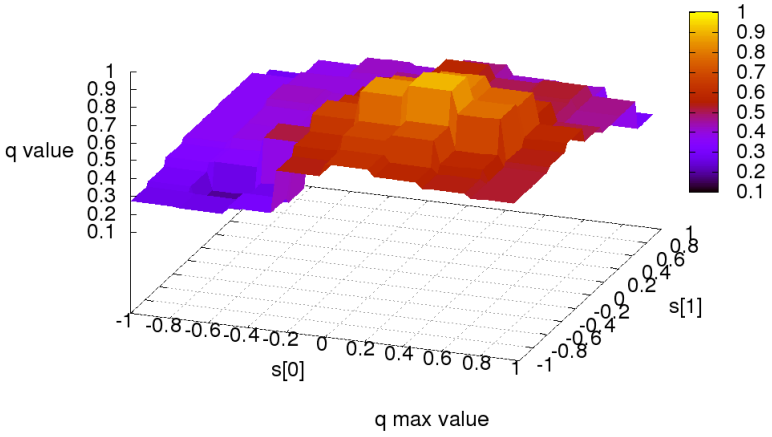


Obr. 3.9: Referenčné riešenie

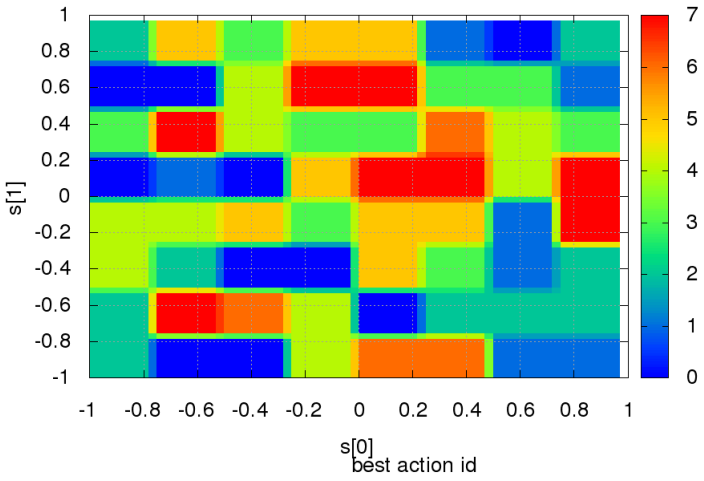


Obr. 3.10: Číslo najlepšej akcie použitím referenčného riešenia

Riešenie pre použitie riedkej tabuľky na aproximáciu je viditeľné na obrázku 3.11. Je vidieť nespojité zmeny, a absenciu schopnosti aproximovať náhle záporné hodnoty požadované zápornou $R(s(n),a(n))$. Zo známeho $Q(s(n),a(n))$ je ďalej možné zostaviť mapu najlepších akcií (očíslované od 0..7). Závislosť čísla najlepšej akcie od stavu je na obrázku 3.12.

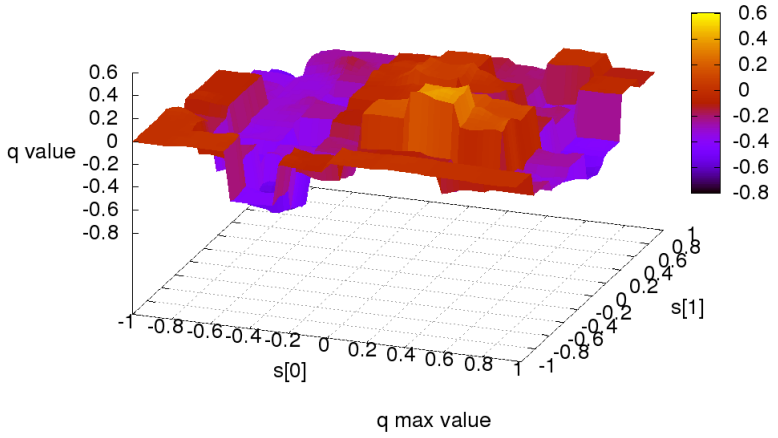


Obr. 3.11: Riešenie aproximácie použitím riedkej tabuľky

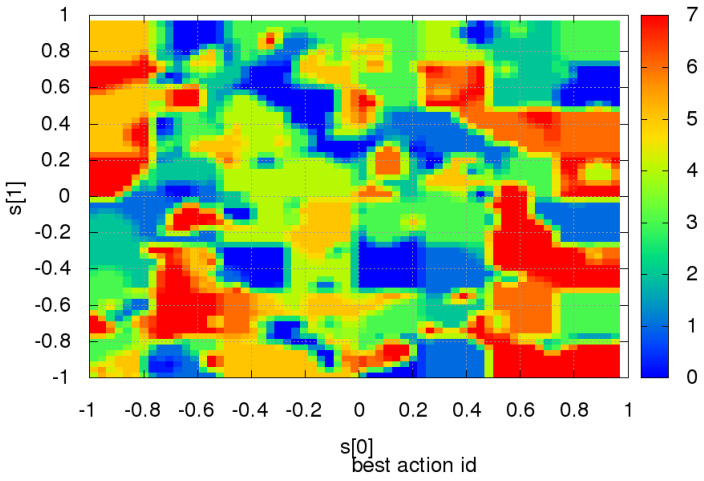


Obr. 3.12: Číslo Najlepšej akcie použitím riedkej tabuľky

Riešenie pre bázickú funkciu typu Gaussova krivka kombinovaná s riedkou tabuľkou (funkcia typu 3, obr. 3.7) je na obrázku 3.13. Je možné vidieť nepojité zmeny spôsobené riedkou tabuľkou aj vyhladené oblasti vďaka Gaussovým krivkám. Podobne ako v predošlom prípade je možné znázorniť závislosť najlepšej akcie od stavu na obrázku 3.14. Oproti riešeniu s riedkou tabuľkou je možné pozorovať zjemnenie prechodov.

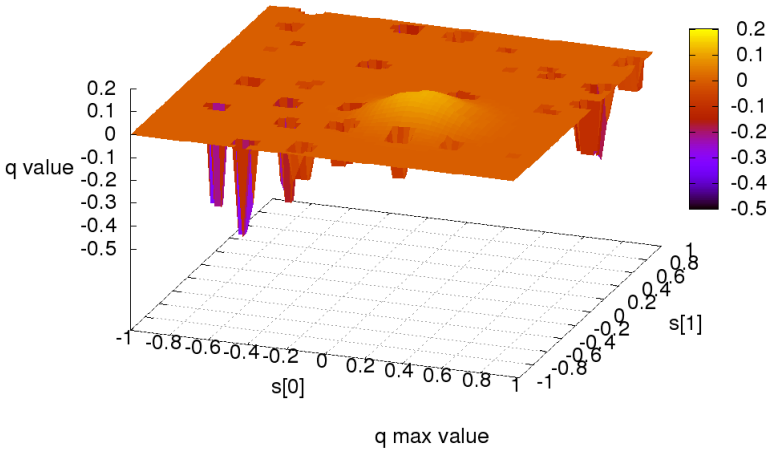


Obr. 3.13: Riešenie aproximácie použitím Gaussovej krivky kombinovanej s riedkou tabuľkou

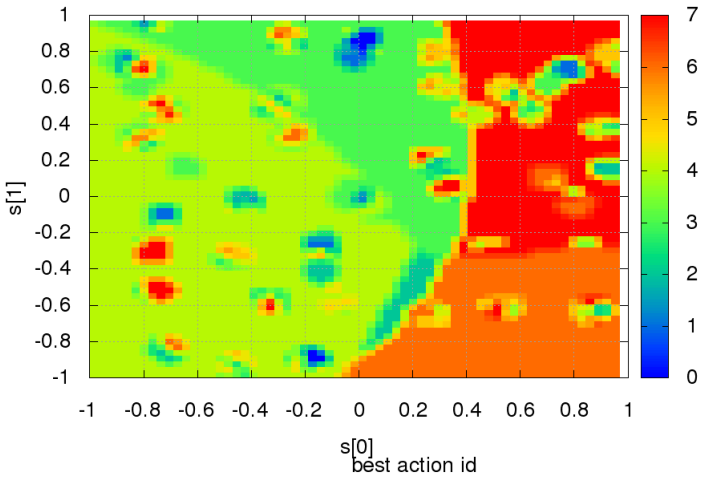


Obr. 3.14: Číslo najlepšej akcie použitím Gaussovej krivky kombinovanej s riedkou tabuľkou

Pre úplnosť, znázornenie riešenia pre novo zavedenú funkciu 2.0.5 - kombinujúc výhody hladkej Gaussovej krivky s tabuľkou, v bodoch kde sú záporné hodnoty $R(s(n),a(n))$. Výsledok pre $\max_{a(n)\in\mathbb{A}} Q_n(s(n),a(n))$ je možné pozrovať na obrázku 3.15 a mapa najlepších akcií je na obrázku 3.16.



Obr. 3.15: Riešenie aproximácie použitím Gaussovej krivky kombinovanej s adaptívnou tabuľkou

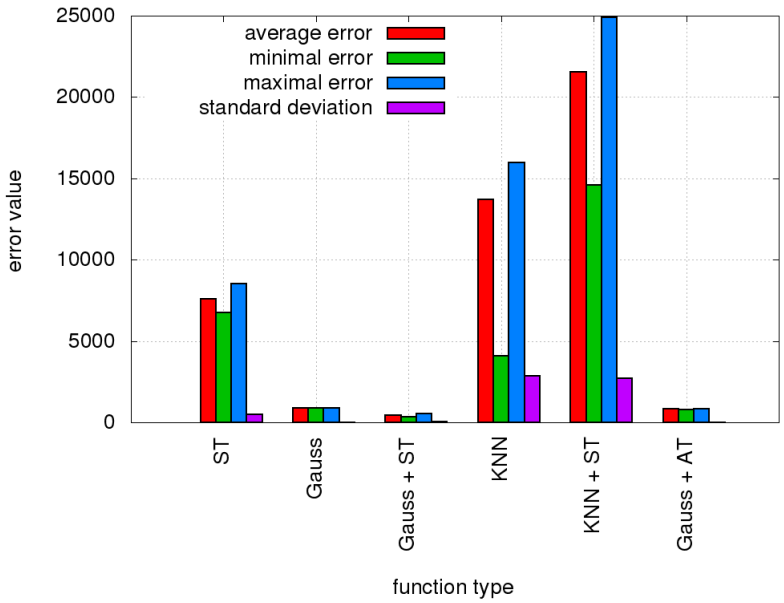


Obr. 3.16: Číslo najlepšej akcie použitím Gaussovej krivky kombinovanej s adaptívnou tabuľkou

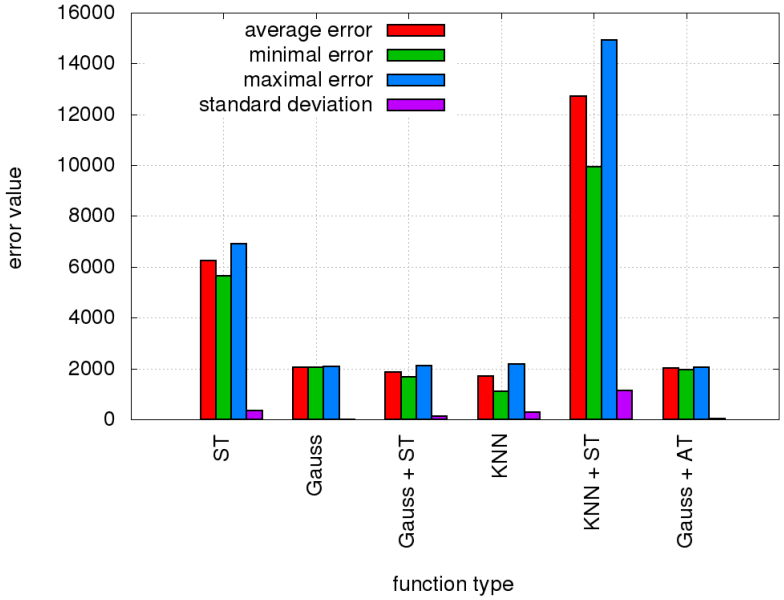
3.5 Priemerné výsledky experimentu

V predošlej časti uvedené výsledky zobrazujú výsledne riešenia pre 3 zvolené aproximačné metódy a jednu mapu - mapa 2. Celkové vyhodnotenie behu všetkých trialov a pre všetky štyri mapy je na nasledujúcich obrázkoch. Sledovali sa veličiny priemerná, maximálna, minimálna chyba a rozptyl chyby. Znázornenie výsledkov je na obrázkoch 3.17, 3.18, 3.19 a 3.20.

Z výsledkov je možné vybrať troch najvhodnejších kandidátov na aproximáciu : Gaussova krivka, Gaussova krivka kombinovaná s riedkou tabuľkou, Gaussova krivka kombinovaná s adaptívnou tabuľkou. Výsledky sa môžu zdať vyrovnané, dôležité je však znázroniť pohyby jednotlivých virtuálnych robotov a podľa toho urobiť záver. Pohyby robotov pre mapu 2 a jednotlivé aproximačné metódy je možné sledovať na obrázkoch 3.21 3.22 3.23 a 3.24. Je zrejmé, že jedina vyhovujúca aproximačná metóda pre uvedené parametre experimentu je novo zavedená

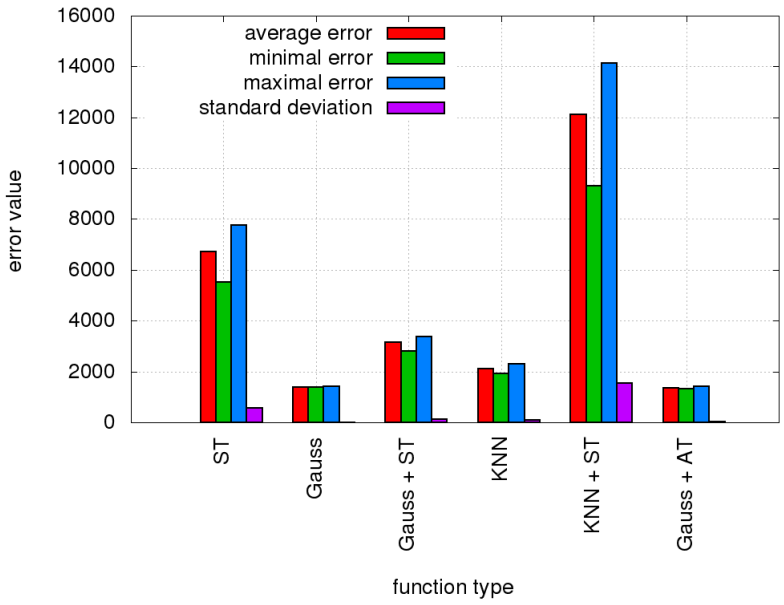


Obr. 3.17: Súhrnné výsledky pre všetky testovacie funkcie a mapu 0

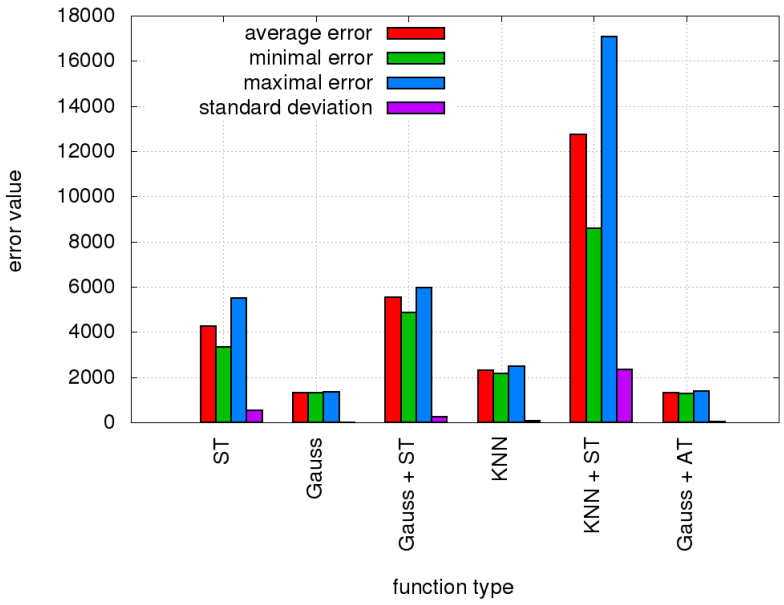


Obr. 3.18: Súhrnné výsledky pre všetky testovacie funkcie a mapu 1

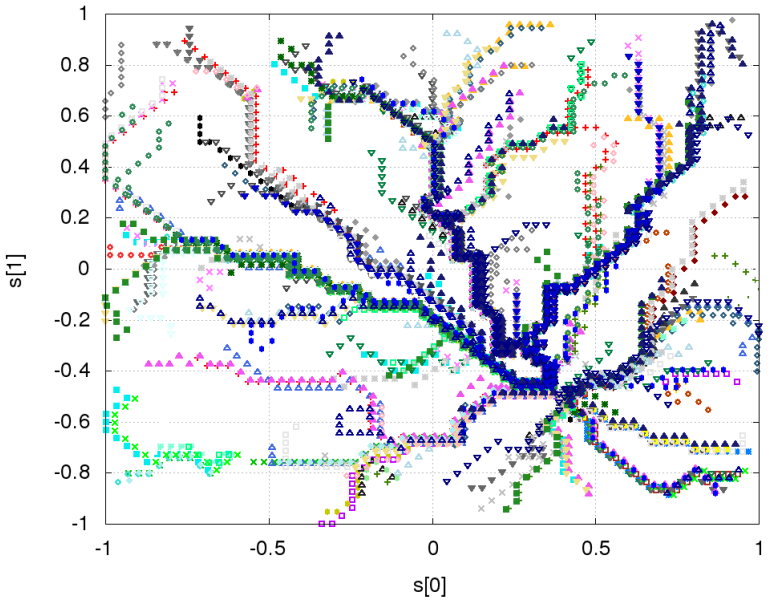
funkcia 2.0.5. Príčinou zlyhania ostatných je neschopnosť zebezpečiť potrebnú strmosť v oblastiach zápornej hodnoty $R(s(n),a(n))$. Samotná rekurentná povaha Q-learning algoritmu spôsobuje, že s rastúcou vzdialenosťou od jediného kladného $R(s(n),a(n))$ sa zmenšujú rozdiely $Q(s(n),a(n))$ pre jednotlivé akcie ktoré je možné v danom stave vykonať. To je obzvlášť nepríjemne pre experiment tak ako bol navrhnutý - malá zmena pohybu robota znamená aj malú zmenu stavu a aproximačná funkcia tak veľmi ťažko zachytí zmenu s požadovanou presnosťou.



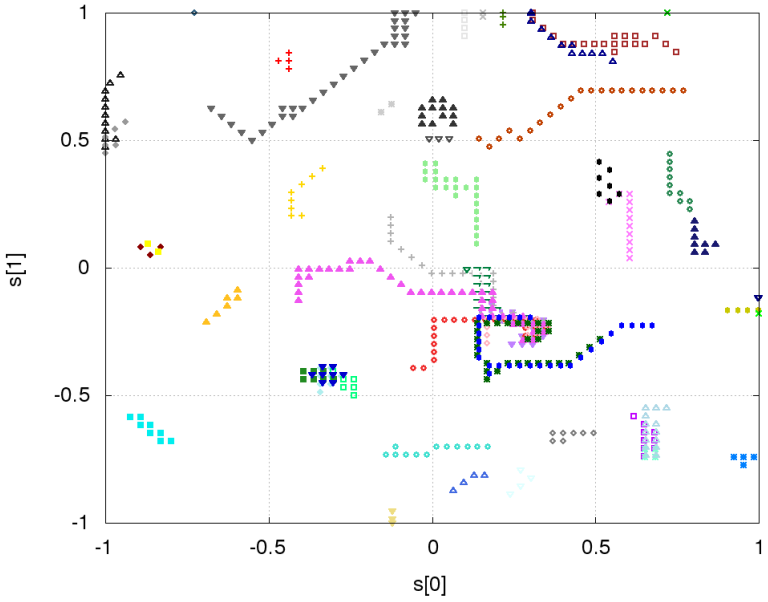
Obr. 3.19: Súhrnné výsledky pre všetky testovacie funkcie a mapu 2



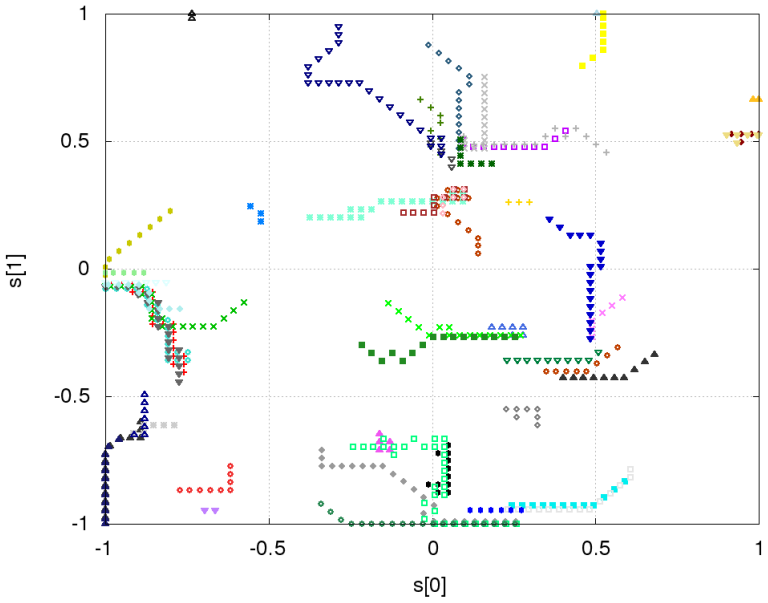
Obr. 3.20: Súhrnné výsledky pre všetky testovacie funkcie a mapu 3



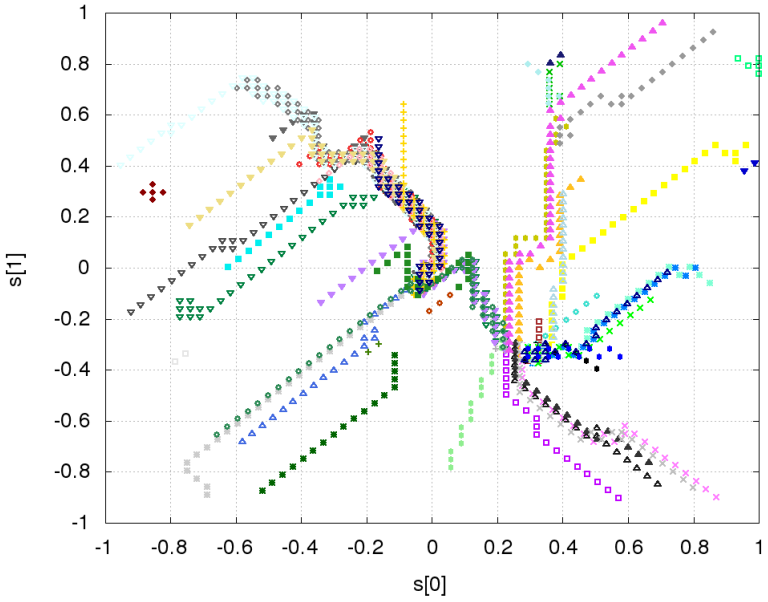
Obr. 3.21: Dráha robotov, referenčné riešenie



Obr. 3.22: Dráha robotov, aproximacia Gaussovou krivkou



Obr. 3.23: Dráha robotov, aproximacia Gaussovou krivkou kombinovanou s riedkou tabuľkou



Obr. 3.24: Dráha robotov, aproximacia Gaussovou krivkou kombinovanou s adaptívnou tabuľkou

Kapitola 4

Záver

Práca rieši problematiku aproximovania funkcie ohodnotení v algoritmoch Q-learning. S pomedzi najčastejšie používaných prístupov bola zvolená aproximácia neurónovou sieťou pomocou bázických funkcií. Oproti bežne používanému prístupu lineárnej kombinácií príznakov (features) sa líši tým, že samotné tvary príznaky si algoritmus stanovuje sám, počas učenia. Zmenšuje sa teda potrebná znalosť programátora.

Vedecký prínos je možné nájsť v

- Ukážka nevhodnosti použitia doprednej siete v predloženom probléme učenou gradientovými metódami. Riešenie nekonvergovalo ani po miliónoch iteráciach v triviálnom experimente s dvoma akciami. Príčinou je nelokálnosť učenia siete - zmena hodnoty v jednom bode, zmení hodnoty v každom bode, a nie nutne k lepšiemu. Od siete sa súčastne požaduje generovanie správnej hodnoty aj učenie v nejakom inom bode.

- Uvedenie algoritmu nanoQ, ktorý vyšetruje systém s jedným stavom. Nepodarilo sa nájsť publikáciu ktorá by tento princíp využívala. Algoritmus môže nájsť uplatnenie v riešení pohybu jednoduchého robota.

- Uvedenie novej bázickej funkcie, ktorá z testovaných najlepšie aproximuje funkciu ohodnotení. Táto funkcia môže byť učená lokálne, a vďaka časti $P(s(n),a(n))$ umožňuje zabezpečiť potrebnú strmosť, bez nutnosti širokého rozsahu parametrov β v časti $H(s(n),a(n))$ - ten môže zostať malý, a riešiť tak šírenie kladnej odmeny na ďalšie stavy v súlade s parametrom γ.

- Testovanie Q-learing algoritmu na reálnom robotovi, kde predstavuje druhú úroveň riadenia. Na spodnej vrstve sa pravuje s PID regulátormi, na druhej sa pomocou Q-learning algoritmu stanovujú žiadané hodnoty.

Napriek uvedeným skutočnostiam a súčastnému stavu komerčnej sféry, autor práce nepredpokladá využitie Q-learning algoritmov v priemyselnej praxi. Medzi hlavné dôvody možno zaradiť konzervatívny prístup riadenia v priemysle, kde väčšinu úloh plnohodnotne vyrieši PID regulátor a nad ním postavená logika vetvenia (napr. rôzne stavové automaty). Práca tak predstavuje nepatrný prínos v teoretickej oblasti reinforcement learning algoritmov. Jediné možné využitie v blízkej dobe je možné nájsť v počítačových hrách. Všetky zdrojové súbory a podrobné výsledky experimentov (vrátane dát na ďalšie smerovanie) sú k dispizícií pod GNU GPL licenciou. Práca tak spadá do kategórie otvorenej vedy. Zdrojové súbory pre Q-learning experiment sú k dispozícií na autorovom gite [46]. Spolu je to cca 55648 súborov, z toho cca. 17000 pripadá na výsledky experimentov a cca 36000 na zdrojové súbory. Zdrojové súbory (vrátane podkladov na výrobu) pre robota Motoko sú k dispozícií na [47].

Literatúra

[1] Nhan Nguyen, NASA Ames Research Center, Moffett Field, CA 94035 : Predictor-Model-Based Least-Squares Model-Reference Adaptive Control with Chebyshev Orthogonal Polynomial Approximation

[2] Girish Chowdhary and Eric Johnson, Least Squares Based Modification for Adaptive Control http://web.mit.edu/girishc/www/publications/files/Chow_Joh_CDC_10_ls.pdf

[3] Sun Pei, Noise Resistant Least Squares Based Adaptive Control, March 27, 2012, Stockholm, Sweden http://www.diva-portal.org/smash/get/diva2:514116/FULLTEXT01.pdf

[4] Prof. Nathan L. Gibson Department of Mathematics, Gradient-based Methods for Optimization. Part I., 2011 http://math.oregonstate.edu/~gibsonn/optpart1.pdf

[5] Antony Jameson, Department of Aeronautics and Astronautics Stanford University, Stanford, CA 94305-4035 Gradient Based Optimization Methods, http://aero-comlab.stanford.edu/Papers/jameson.gbom.pdf

[6] L. Hasdorff, Gradient optimization and nonlinear control, ISBN 0471358703, https://books.google.cz/books?id=o_ZQAAAAMAAJ

[7] Kevin L. Moore, Iterative Learning Control, http://inside.mines.edu/~kmoore/survey.pdf

[8] Kevin L. Moore, An Introduction to Iterative Learning Control Theory, http://inside.mines.edu/~kmoore/504_ILC_Seminar-Save.pdf

[9] Jeff Heaton, Introduction to Neural Networks with Java, Heaton Research, Inc., 2008, ISBN 1604390085

[10] CHRISTOPHER J.C.H. WATKINS, PETER DAYAN : Technical Note Q-Learning, Machine Learning, 8,279-292 (1992) http://www.gatsby.ucl.ac.uk/~dayan/papers/cjch.pdf

[11] Q-learning 1 https://www-s.acm.illinois.edu/sigart/docs/QLearning.pdf

[12] Q-learning 2 http://mnemstudio.org/path-finding-q-learning-tutorial.htm

[13] Francisco S. Melo Institute for Systems and Robotics, Instituto Superior Técnico, Lisboa, PORTUGAL : Convergence of Q-learning: a simple proof http://users.isr.ist.utl.pt/~mtjspaan/readingGroup/ProofQlearning.pdf

[14] Eyal Even-Dar, Yishay Mansour : Convergence of optimistic and incremental Q-learning, http://web.cs.iastate.edu/~honavar/rl-optimistic.pdf

[15] Carden, Stephen, "Convergence of a Reinforcement Learning Algorithm in Continuous Domains"(2014). All Dissertations. Paper 1325. http://tigerprints.clemson.edu/cgi/viewcontent.cgi?article=2326&context=all_dissertations

[16] Francisco S. Melo and M. Isabel Ribeiro, Convergence of Q-learning with linear function approximation, Proceedings of the European Control Conference 2007 Kos, Greece, July 2-5, 2007, http://gaips.inesc-id.pt/~fmelo/pub/melo07ecc.pdf

[17] Karan M. Gupta Department of Computer Science Texas TechUniversity Lubbock, TX 79409-3104 : Performance Comparison of Sarsa(λ) and Watkin's Q(λ) Algorithms, http://www.karanmg.net/Computers/reinforcementLearning/finalProject/KaranComparisonOfSarsaWatkins.pdf

[18] R. Rojas: Neural Networks, Springer-Verlag, Berlin, 1996, Kohonen Networks https://page.mi.fu-berlin.de/rojas/neural/chapter/K15.pdf

[19] Steven K. Rogers, Matthew Kabrisky SPIE Press, 1991, ISBN 0819405345 : An Introduction to Biological and Artificial Neural Networks for Pattern Recognition https://books.google.cz/books?id=uo4Smk6QnTgC

[20] Teuvo Kohonen and Timo Honkela (2007), Scholarpedia, 2(1):1568 : Kohonen network http://www.scholarpedia.org/article/Kohonen_network

[21] Markovove rozhodovacie procesy, stručne : Pieter Abbeel UC Berkeley EECS : Markov Decision Processes and Exact Solution Methods http://www.cs.berkeley.edu/~pabbeel/cs287-fa12/slides/mdps-exact-methods.pdf

[22] Martin L. Puterman : Markov Decision Processes: Discrete Stochastic Dynamic Programming , isbn 9781118625873, rok 2014, https://books.google.sk/books?id=VvBjBAAAQBAJ

[23] NanoQ learning zdrojové súbory https://github.com/michalnand/q_learning/tree/master/src/nano_q_learning

[24] Fundamentals of Artificial Neural Networks Mohamad H. Hassoun, MIT Press, 1995

[25] B. Irie Auditory & Visual Perception Res. Lab., ATR, Osaka, Japan, S. Miyake : Neural Networks, 1988., IEEE International Conference on, INSPEC 3350063

[26] Kolomongorov teorém, stručne https://en.wikipedia.org/wiki/Universal_approximation_theorem

[27] R. Rojas: Neural Networks, Springer-Verlag, Berlin, 1996, chap 7

[28] Martin Riedmiller, Computer Standards & Interfaces Volume 16, Issue 3, July 1994, Pages 265-278 : Advanced supervised learning in multi-layer perceptrons — From backpropagation to adaptive learning algorithms

[29] J. Leonard, M.A. Kramer, Computers & Chemical Engineering Volume 14, Issue 3, March 1990, Pages 337–341 Improvement of the backpropagation algorithm for training neural networks

[30] Jonathan Engel, Norman Bridge Laboratoryof Plly sics 161-33, California Institute of Technology, Pasadena, CA 91125, USA : Teaching Feed-Forward Neural Networks by Simulated Annealing

[31] Francisco S. Melo, Sean P. Meyn, M. Isabel Ribeiro An Analysis of Reinforcement Learning with Function Approximation, Appearing in Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland, 2008 http://www.machinelearning.org/archive/icml2008/papers/652.pdf

[32] David Silver : Lecture 6: Value Function Approximation http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/FA.pdf

[33] Francisco S. Melo M. Isabel Ribeiro : Q-learning with linear function approximation http://gaips.inesc-id.pt/~fmelo/pub/melo07tr-b.pdf

[34] Marina Irodova and Robert H. Sloan : Reinforcement Learning and Function Approximation, 2005, American Association for Artificial Intelli-gence (www.aaai.org) http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.81.7833&rep=rep1&type=pdf

[35] Punit Pandey, Dr. Shishir Kumar, DeepshikhaPandey, Reinforcement Learning by Comparing Immediate Reward, (IJCSIS) International Journal of Computer Science and Information Security, Vol. 8 , No. 5, August 2010 http://arxiv.org/pdf/1009.2566.pdf

[36] Melanie Coggan, CRA-W DMP Project at McGill University (2004) : Exploration and Exploitation in-Reinforcement Learning http://ftp.bstu.by/ai/To-dom/My_research/Papers-2.1-done/RL/0/FinalReport.pdf

[37] Mark Humphrys Trinity Hall, University of Cambridge August 1996, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.73.8309&rep=rep1&type=pdf

[38] Jinyi Yao Dept. of Comput. Sci. & Technol., Tsinghua Univ., Beijing, China Jiang Chen ; Zengqi Sun An application in RoboCup combining Q-learning with adversarial planning, 2002 http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1022159&abstractAccess=no&userType=inst

[39] Asma Al-Tamimi, Frank L. Lewis , Murad Abu-Khalaf Model-free Q-learning designs for linear discrete-time zero-sum games with application to H-infinity control, 2006 http://www.sciencedirect.com/science/article/pii/S0005109806004249

[40] Asma Al-Tamimi, Frank L. Lewis , Murad Abu-Khalaf Model-free Q-learning designs for linear discrete-time zero-sum games with application to H-infinity control, 2006 http://www.sciencedirect.com/science/article/pii/S0005109806004249

[41] Christopher J. C. H. Watkins, Peter Dayan Q-learning, 1992 http://link.springer.com/article/10.1007/BF00992698

[42] Mae L. Seto Springer Science & Business Media, 9. 12. 2012, Marine Robot Autonomy, ISBN 1461456592, chap 7.3.3.2

[43] Peter Dayan, Christopher J.C.H. Watkins, Reinforcement Learning http://www.gatsby.ucl.ac.uk/~dayan/papers/dw01.pdf

[44] Daniel Dewey, Oxford Martin Programme on the Impacts of Future Technology, Future of Humanity Institute : Reinforcement Learning and the Reward Engineering Principle http://www.danieldewey.net/reward-engineering-principle.pdf

[45] video robota Motoko Aftermath Michal Chovanec, youtube https://www.youtube.com/watch?v=8sskJN_zuko

[46] Michal Chovanec, Q-learning zdrojové súbory https://github.com/michalnand/q_learning

[47] Michal Chovanec, Motoko robot zdrojové súbory https://github.com/michalnand/motoko_after_math_linefollower