Univerzitet u Novom Sadu
Fakultet tehničkih nauka
Departman za računarstvo i automatiku

Prošireni Taksi Problem-Učenje Potkrepljenjem

Projekat urađen iz predmeta:Samoobučavajući i adaptivni algoritmi

Irina Radanović (IN 5/2022)
Nikolina Brkljač(IN 2/2022)
Studijski program:Informacioni inženjering
Februar, 2026.

Sadržaj:

1. Uvod

Ovaj projekat se bavi rešavanjem problema optimalnog upravljanja agentom u okruženju korišćenjem reinforcement learning algoritama. Cilj je naučiti strategiju koja maksimizuje nagradu kroz interakciju sa okruženjem.

2. Opis problema i okruženja

U ovom projektu ćemo implementirati osnovni problem sa taksijem sa dodatnom benzinskom pumpom i klizavim poljima.
Taksi se kreće duž mreže 5x5. Njegov zadatak se sastoji od tri faze:
1. Pronalaženje putnika: odlazak na jednu od 4 stanice (R, G, Y, B) gde se putnik nalazi.
2. Izvršavanje akcije PREUZIMANJA
3. Kretanje ka odredištu i izvršavanje akcije ISPORUKE

Dodaci problemu:
1. Upravljanje resursima (gorivo): Svako kretanje troši 1 jedinicu goriva. Ako gorivo dostigne 0, taksi se zaustavlja i epizoda se završava neuspehom. Agent mora da nauči kada da skrene ka benzinskoj pumpi.
2. Stohastičnost (led): Na određenim poljima (ledenim delovima), taksi može da prokliza. To znači da agent daje komandu „Idi na sever“, ali završava npr. „na istoku“. Ovo uči agenta da izbegava ta polja ili da planira neuspeh.
3. Izbegavanje prepreka (zidovi): Polja koja potpuno blokiraju kretanje

Okruženje:

Naše prilagođeno okruženje će naslediti gymnasium.Env koja definiše strukturu koju sva okruženja moraju da prate.

Gorivo: maks. 10 jedinica (0-9)
Lokacija putnika: 0=R, 1=G, 2=Y, 3=B, 4=u taksiju.

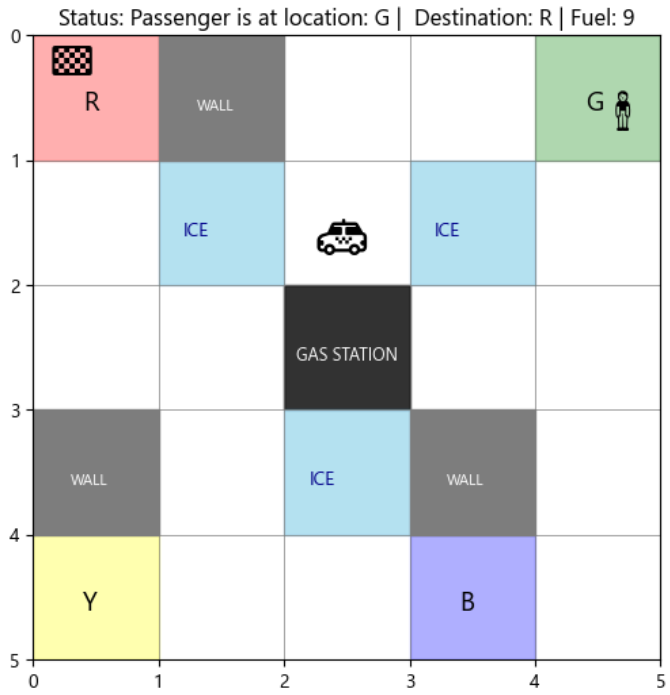
Odredište: 0=R, 1=G, 2=Y, 3=B
Stanje je definisano sa [red(pozicija taksija), kolona(pozicija taksija), lokacija putnika, odredište, gorivo]

Akcije:
0 - N
1 - S
2 - E
3 - W
4 - PREUZIMANJE
5 - ISPORUKA
6 - TOČENJE GORIVA

Sistem nagrađivanja:
-1 po koraku, osim ako se ne aktivira druga nagrada
-5 ako taksi udari u granicu ili zid unutar table (prepreka)
+10 uspešno preuzimanje
-10 neuspešno preuzimanje ili neuspešna isporuka
+50 za isporuku putnika
-50 ako nestane goriva pre kraja epizode

3. Test funkcije

Funkcija visualize(env) za izgled table i prikaz svih polja.
env-Taksi okruženje



Funkcija test_ice_logic(env) koja testira da li dobro rade klizava polja i gde će se taksi okliznuti.
Funkcija test_wall_logic(env) koja testira da li dobro rade zid polja i blokiraju taksi.

=>Oba testa su bila uspešna, zidovi blokiraju taksi, na ledu proklizava.

Funkcija test_performance(env, agent) testira performanse istreniranog agenta na 2000 nezavisnih epizoda i ispisuje test rezultate.
Osim ovih funkcija, računamo i test statistike i iscrtavamo grafik odnosa epizoda i nagrada.

4. Algoritmi

Implementirani algoritmi:

1. Q-Learning
   Formula ažuriranja:
   $$Q(s,a)=Q(s,a)+\alpha[r+\gamma \max Q(s',a)-Q(s,a)]$$
   α-learning rate
   γ-gama faktor obezvređivanja
2. Monte Karlo
   Formula ažuriranja:
   $$Q(s,a)=\text{Sum of Returns}(s,a)/\text{Visit Count}(s,a)$$
3. SARSA
   Formula ažuriranja:
   $$Q(s, a) = Q(s, a) + \alpha[r + \gamma\, Q(s', a') - Q(s, a)]$$

4.1. Rezultati Q-Learning algoritma

Akcije se biraju korišćenjem ε-pohlepne politike, tako da istraživanje i eksploatacija budu uravnoteženi.
Treniranje izvršeno nad 30000 epizoda, $\alpha = 1$, $\gamma = 0.99$.
Statistika nad istreniranim Q-Learning agentom:

Statistika treninga za 30000 epizoda:
Prosečna nagrada po epizodi: -8.457033
Maksimalna ostvarena nagrada: 55
Minimalna ostvarena nagrada: -352

Rezultati sa test_performance() koji simulira 2000 epizoda radi dodatne statistike:

Rezultati testova preko 2000 epizoda
Stopa uspeha: 99.20%
Stopa potrošnje goriva: 0.40%
Prosečan broj koraka: 17.26

Interpretacija:
Prosečna nagrada: -8.5, s obzirom na to da je okruženje dosta strogo i da ima puno penalizovanja, ovo je dosta dobar rezultat
Maksimalna nagrada: 55, skoro pa idealan scenario, nagrada od 50 za uspešnu isporuku uz minimalan trošak goriva
Minimalna nagrada: -352, rezultati sa početka treninga, agent luta, udara u zidove, granice i nestaje bez goriva

Stopa uspeha: 99.20% , s obzirom na strogu prirodu okruženja(led(stohastičnost), malo goriva, zidovi), ovaj rezultat je odličan i ukazuje nam na to da se agent dobro snalazi i uspeva da izvrši zadatak
Stopa potrošnje goriva (broj puta kada je taksi nestao bez goriva/broj epizoda): Stopa od 0.40% dokazuje da je agent uspešno naučio upravljanje resursima i da posetu benzinskoj stanici prioritizuje iznad direktnog puta.
Prosečan broj koraka: 17.26 , agent se kreće blizu optimalnog

4.2. Rezultati Monte Carlo algoritma

Akcije se biraju korišćenjem ε-pohlepne politike. Specifičnost ovog algoritma je što se ažuriranje vrši tek na kraju svake epizode (First-visit MC), koristeći kumulativnu nagradu. Treniranje izvršeno nad 30000 epizoda, $\gamma = 0.99$. Statistika nad istreniranim Monte Carlo agentom:

Prosečna nagrada po epizodi:-101.918933
Maksimalna ostvarena nagrada: -48
Minimalna ostvarena nagrada: -10930

Rezultati sa test_performance() koji simulira 2000 epizoda:

- Stopa uspeha: *0.0%*
- Stopa potrošnje goriva: *98.60%*
- Prosečan broj koraka: *11.70*

Interpretacija:

Monte Carlo algoritam u ovom okruženju pokazuje specifično ponašanje. S obzirom na to da uči iz celih sekvenci, on teže pronalazi optimalnu putanju kada je gorivo strogo ograničeno. Ako agent tokom nasumičnog istraživanja ne stigne do cilja, on dobija samo negativne povratne informacije, što može dovesti do niže stope uspeha u poređenju sa TD metodama.

Na grafiku krive učenja za Monte Carlo uočava se sporiji trend rasta, jer je svakom ažuriranju potrebno da se epizoda završi, a uspeh u okruženju sa ledom i zidovima zahteva precizno planiranje koje MC teže generiše iz čiste stohastike na početku.

4.3. Rezultati SARSA algoritma

SARSA je implementirana kao *on-policy* algoritam, gde se ažuriranje Q-vrednosti vrši na osnovu akcije koja je zapravo izabrana za sledeće stanje. Treniranje izvršeno nad 30000 epizoda, $\alpha = 0.1$, $\gamma = 0.99$. Statistika nad istreniranim SARSA agentom:

Prosečna nagrada po epizodi:-15.967000
Maksimalna ostvarena nagrada: 56
Minimalna ostvarena nagrada: -444

Rezultati sa test_performance() koji simulira 2000 epizoda:

- Stopa uspeha: *93.70%*
- Stopa potrošnje goriva: *3.00 %*
- Prosečan broj koraka: *22.34*

Interpretacija:

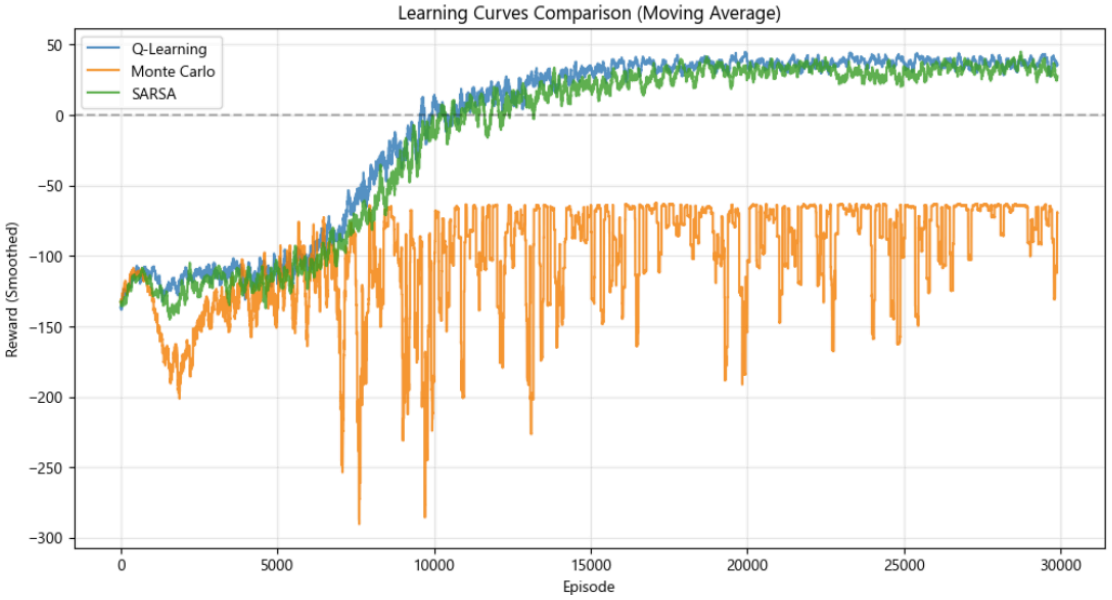
SARSA pokazuje visok stepen robusnosti. Zbog svoje *on-policy* prirode, ovaj algoritam je "svestan" opasnosti koje nose proklizavanja na ledu. Agent uči konzervativniju politiku kretanja kako bi izbegao kazne od udaraca u zid ili gubitka goriva usled neočekivanih skretanja.

Rezultati testa pokazuju visoku stopu uspeha, sličnu Q-Learningu, ali sa tendencijom ka nešto većem prosečnom broju koraka, što potvrđuje da SARSA radije bira "sigurniji" nego nužno najkraći put kroz klizava polja.

5. Komparativna analiza rezultata

U ovom delu rada vršimo direktno poređenje performansi tri primenjena algoritma: **Q-Learning**, **Monte Carlo** (MC) i **SARSA**. Analiza se zasniva na stabilnosti učenja, brzini konvergencije i krajnjoj efikasnosti agenta u rešavanju zadatka.

Grafik ispod prikazuje kretanje prosečne nagrade kroz epizode za sva tri algoritma. Da bi se lakše uočili trendovi, primenjen je pokretni prosek (moving average) na prozoru od 100 epizoda.



Interpretacija grafika:

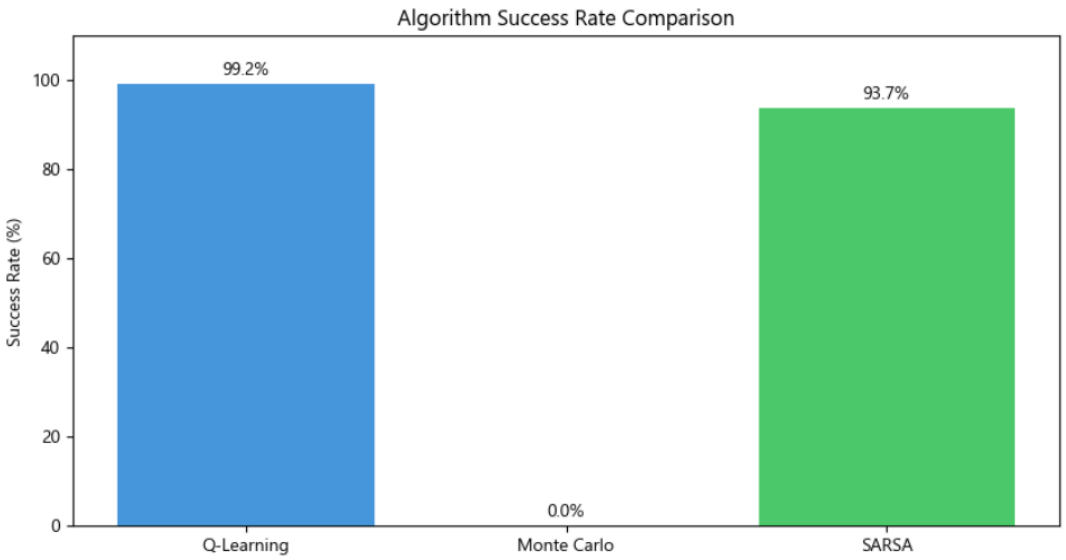
Q-Learning pokazuje najbržu konvergenciju. Veoma rano dostiže zonu pozitivnih nagrada, što ukazuje na efikasnost *off-policy* učenja u pronalaženju optimalne putanje uprkos stohastičnosti leda.

SARSA prati sličan trend kao Q-Learning, ali sa nešto sporijim usponom i manjom krajnjom prosečnom nagradom. Ovo je očekivano jer SARSA uči "sigurniju" politiku (on-policy), uzimajući u obzir i kazne koje bi mogla dobiti tokom faze istraživanja (ϵ-greedy).

Monte Carlo ostaje na dnu grafika sa nagradama koje se kreću oko -100 i niže. U okruženju sa strogim ograničenjem goriva, ovaj algoritam ne uspeva da konvergira jer retko dolazi do terminalnog stanja uspešnom isporukom putnika, te nema dovoljno pozitivnih primera za uspešno ažuriranje Q-vrednosti.

6. Zaključak

Kroz projekat smo pokazali da prošireni taksi problem zahteva sofisticirano upravljanje resursima. **Q-Learning** se pokazao kao najefikasniji u pronalaženju najkraćeg puta, dok je **SARSA** pokazala stabilnost u radu sa stohastičnim ledom. **Monte Carlo** metoda je najviše ispaštala zbog strogog limita goriva, što potvrđuje da su TD metode (Temporal Difference) pogodnije za okruženja sa oskudnim pozitivnim nagradama i visokim penalima.


Algorithm Success Rate Comparison
Success Rate (%)
0
20
40
60
80
100
99.2%
0.0%
93.7%
Q-Learning
Monte Carlo
SARSA