Jackpot - koliko denarja priigra tvoj algoritem?

Repozitorij

Vsa izvorna koda je dostopna na GitHub repozitoriju:
https://github.com/andrazhribernik/jackpot

Zagon algoritma

1. Namestite python 2.7 in pip
2. git clone https://github.com/andrazhribernik/jackpot.git
3. cd jackpot
4. python jackpot.py http://<host_name>/<example_number>

Opis ciljev, pristopov in rezultatov

Cilj jacpot problema je bil, da izdelamo algoritem, ki bo sposoben priigrati kar se da čim večji zaslužek. Poiskati smo morali torej optimalno razmerje med raziskovanjem najboljšega igralnega avtomata (exploration) in izkoriščanjem najboljšega avtomata (exploitation). Takšne probleme lahko rešujemo z MAB algoritmi. V sklopu rešitve smo preizkusili pet različnih MAB algoritmom (ε-greedy, UCB1, UCB2, exp3 ter RPM).

Za potrebe evaluacije smo najprej preiskali kakšni so CTR-ji igralnih avtomatov pri vseh 10ih problemih, ki so nam bili na voljo za testiranje naše rešitve. Potem pa smo izdelali lasten lokalni strežnik, ki je simuliral Celtrin API. S tem smo povečali hitrost testiranja in hkrati imeli vpogled v ciljne CTR-je avtomatov. Na ta način smo lahko preverili kolikšen odstotek vsega možnega dobička smo priigrali. Oglejmo si naš postopek testiranja na primeru številka 6, kjer imamo 2 avtomata, ki imata CTR, ki je prikazan na Figure 1. Vidimo lahko, da je maksimalno dobiček, ki ga lahko priigramo pri tem primeru 600, saj ima najboljši avtomat v danem trenutku verjetnost dobička 0.6 (0.6 * 1000 potegov = 600). Naš najboljši algoritem je na tem primeru v povprečju (simulacijo smo ponovili 20-krat) priigral dobiček 572, kar znaša več kot **95% maksimalne nagrade**. Tak postopek smo ponovili za vse primere in za vse algoritme.

Na primerih s konstantnimi CTR-ji (primeri 1-5) se je kot najboljši izkazal RPM pristop, ki pa je bil zelo slab na primerih, kjer se CTR-ji spreminjajo. Zaradi tega smo ta algoritem prilagodili tako, da hrani zgodovino odzivov za omejeno časovno obdobje v preteklosti. Takšen pristop se je izkazal za dobrega tudi na primerih, kjer se CTR-avtomatov spreminja skozi čas. S tem pristopom smo na vseh 10ih primerih skupaj priigrali cca. **85% maksimalne nagrade** na naših primerih**.** Na Celtrinih 10ih primerih pa smo priigrali dobiček **3475** (sedaj bomo pa videli, kako dobro je to v primerjavi z algoritmi ostalih kolegov ☺).

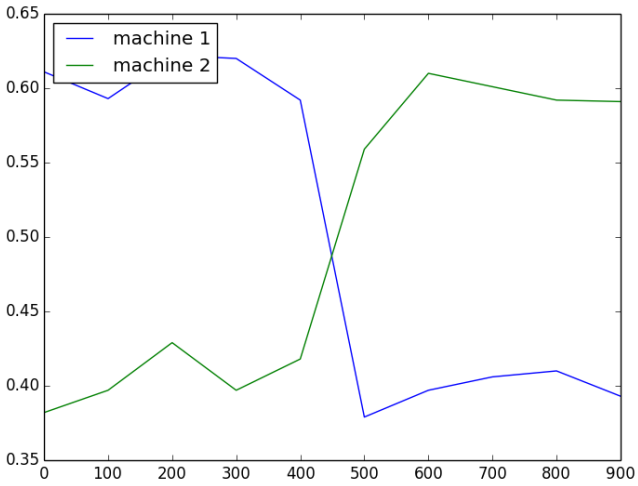

Figure 1 CTR igralnih avtomatov za primer št. 6

Vsi rezultati do katerih smo pri testiranju prišli so na voljo v mapi *results*.

Porabljen čas

1. 20-30 ur programiranja
2. 10-15 ur testiranja in izbor najboljšega pristopa