Návrh a Evaluace UI Agentů pro Hru Farkle

Autor: Jakub Kučera
Předmět: BI-ZUM
Vyučující: Pavel Surynek

Abstrakt

Tato práce se zabývá návrhem, implementací a evaluací různých strategií umělé inteligence pro kostkovou hru Farkle. Podobně jako u karetních her je jedná o hru s neurčitostí . Cílem práce bylo vytvořit prostředí pro hru Farkle, implementovat různé typy agentů (naivní, heuristický, trénovaný pomocí posilovaného učení) a umožnit porovnání úspěšnosti ve vzájemných zápasech. Program dovoluje hru Hráč vs. Hráč a Hráč vs. AI a AI vs. AI. Klíčovou součástí práce bylo navržení zjednodušeného akčního prostoru pro RL agenta a definování vhodného observačního prostoru a systému odměn.

Úvod

Tento protokol popisuje implementaci hry Farkle, vytvoření standardizovaného prostředí pomocí knihovny PettingZoo, návrh různých agentů a metodiku jejich vzájemného porovnání. Důraz je kladen na popis redukce komplexního akčního prostoru hry na zvládnutelnou velikost pro trénování RL agenta a definici observačního prostoru a odměn.

1 Popis hry Farkle

Farkle je kostková hra pro dva a více hráčů, typicky hraná se šesti šestistěnnými kostkami. Cílem hry je jako první dosáhnout předem stanoveného počtu bodů (např. 5 000). Hráči se střídají v tazích. Tah hráče probíhá následovně:

1. **Hod kostkami:** Hráč hodí všemi dostupnými kostkami (na začátku tahu šesti).
2. **Výběr bodující kombinace:** Hráč musí vybrat alespoň jednu kostku nebo kombinaci kostek, která má bodovou hodnotu (např. 1 = 100 bodů, 5 = 50 bodů, trojice = 100 * číslo na kostce, atd.). Vybrané kostky se odloží stranou.
3. **Rozhodnutí (Pokračovat / Zapsat):**
    a. **Pokračovat (Roll again):** Pokud hráči po výběru bodujících kostek zbyly nějaké kostky k házení, může se rozhodnout znovu hodit zbývajícími kostkami a pokusit se navýšit skóre v daném tahu. Body z předchozích hodů v rámci tahu se sčítají.
    b. **Zapsat (Bank):** Hráč se může kdykoli po výběru bodující kombinace rozhodnout ukončit svůj tah a připsat si body získané v tomto tahu ke svému celkovému skóre.
4. **Farkle:** Pokud hod kostkami neobsahuje žádnou bodující kostku nebo kombinaci, hráč ztrácí všechny body získané v tomto *aktuálním* tahu a na řadě je další hráč. Tomuto se říká "Farkle".
5. **Hot Dice:** Pokud hráč použije všechny své kostky k vytvoření bodujících kombinací v rámci jednoho nebo více hodů, může si ponechat získané body a znovu hodit všemi šesti kostkami pro potenciální další navýšení skóre v rámci stejného tahu.

Hra končí, když jeden z hráčů dosáhne nebo překročí cílové skóre. Ostatní hráči mají ještě jeden tah na to, aby se pokusili překonat jeho výsledek.

2 Návrh řešení

Řešení sestává z několika klíčových komponent: implementace samotné hry, vytvoření standardizovaného prostředí pro interakci s agenty, a návrh a implementace samotných agentů.

2.1. Implementace hry a GUI

Byla vytvořena základní implementace pravidel hry Farkle v jazyce Python. Součástí je i jednoduché grafické uživatelské rozhraní (GUI), které umožňuje vizualizaci hry a interakci pro módy Hráč vs. Hráč a Hráč vs. AI a AI vs. AI.

2.2. PettingZoo prostředí

Pro umožnění trénování agentů pomocí standardních nástrojů pro posilované učení a pro snadnou interakci více agentů byla hra Farkle zapouzdřena do prostředí kompatibilního s knihovnou PettingZoo. PettingZoo je standardní API pro multi-agentní prostředí v Pythonu. Toto prostředí poskytuje agentům pozorování (observation), přijímá jejich akce (action) a vrací odměny (reward) a informaci o ukončení hry (termination/truncation).

2.3. Prostor akcí a jeho redukce

Přirozený prostor akcí ve hře Farkle je poměrně komplexní. Po každém hodu kostkami (který obsahuje bodující kombinaci) má hráč dvě hlavní volby:

1. **Zapsat body (Bank):** Vybrat *specifickou* bodující kombinaci kostek a ukončit tah.
2. **Pokračovat (Continue):** Vybrat *specifickou* bodující kombinaci kostek a pokračovat hodem zbývajícími kostkami.

Problémem je, že možných *specifických* bodujících kombinací může být více (např. při hodu 1, 1, 2, 5, 5, 6 lze vybrat {1}, {5}, {1, 1}, {5, 5}, {1, 5}, {1, 1, 5}, {1, 5, 5}, {1, 1, 5, 5}). To vede k velkému a variabilnímu akčnímu prostoru, který je náročný pro trénování RL agentů.

Navrhli jsme **redukci akčního prostoru**. Místo výběru *specifické* kombinace kostek se agent rozhoduje mezi:

1. **Zapsat body pro k kostek (Bank k dice)**
2. **Pokračovat s k kostkami (Continue with k dice)**

kde k je počet kostek, které budou *odloženy* jako bodující. Implicitně se předpokládá, že agent **vždy vybere tu kombinaci k kostek, která dává nejvyšší skóre**. Pokud existuje více kombinací se stejným počtem k kostek a stejným nejvyšším skóre, výběr mezi nimi je pro účely dalšího hodu (počtu zbývajících kostek) ekvivalentní.

Validita redukce: Tato redukce je pro optimální strategii validní. Pokud se agent rozhodne pokračovat, měl by vždy maximalizovat své okamžité skóre z daného hodu pro zvolený počet odložených kostek k. Otázka, *které* k zvolit (pokud je více možností), nebo zda raději zapsat body, je pak strategickým rozhodnutím, které se agent učí. Agent se tedy rozhoduje mezi Bank a Continue_k1, Continue_k2, ..., Continue_k_max, kde ki odpovídá počtu kostek v nejlepší bodující kombinaci dané velikosti. Tato redukce signifikantně zmenšuje akční prostor. Převod z redukované akce (Continue_k) zpět na konkrétní tah ve hře je jednoznačný: najdi nejlepší bodující kombinaci k kostek, odlož je a hoď zbytkem.

2.4. Observation Space

Observační prostor (observation space) definuje informace, které agent dostává v každém kroku, aby se mohl rozhodnout. Musí obsahovat všechny relevantní informace pro strategické rozhodnutí. Pro hru Farkle byl navržen následující observační prostor:

- **Vaše zbývající skóre do vítězství**
- **Soupeřovo zbývající skóre do vítězství**
- **Aktuální skóre v tahu**
- **Počet kostek v aktuálním hodu**
- **Skóre za kombinaci 1 kostky pro aktuální hod**
- **...**
- **Skóre za kombinaci 6 kostek pro aktuální hod**

Tento prostor poskytuje agentovi dostatek informací pro rozhodnutí, zda riskovat další hod (s ohledem na zbývající kostky, potenciální zisk a riziko Farkle) nebo zda zapsat body.

Ve hře s více než 2 hráčy udává druhý bod zbývající skóre soupeře s největším počtem bodů. Pokud pro k kostek neexistuje skórující kombinace, je uvedené skóre rovno nule (přestože se nejedná o platný tah).

2.5. Funkce odměn

Pro trénování RL agenta byla zvolena jednoduchá a přímá funkce odměn:

- **+1** za vítězství ve hře.
- **-1** za prohru ve hře.
- **0** za všechny mezikroky během hry.

Tato tzv. řídká odměna (sparse reward) je standardní pro epizodické hry a nutí agenta optimalizovat svou strategii pro konečný cíl – výhru. Ačkoli existují alternativní metody (reward shaping), tato základní forma je pro začátek dostačující a zabraňuje agentovi "zneužívat" systém odměn nesprávným způsobem.

3 Návrh a implementace agentů

Byly implementovány a porovnány následující typy agentů:

3.1. Naivní agent

Tento agent se řídí velmi jednoduchými pravidly:

Výběr kombinace: Vždy vybere kombinaci kostek, která dává nejvyšší okamžité skóre.

Rozhodnutí pokračovat/zapsat: Pokud má k dispozici alespoň 3 kostky k dalšímu hodu, vždy pokračuje (Roll again). V opačném případě zapíše body (Bank)

3.2. Heuristický agent

Tento agent se snaží hrát "chytřeji" na základě pravděpodobnosti a očekávaného zisku:

- **Výběr kombinace:** Vybere kombinaci maximalizující očekávaný bodový zisk.
- **Rozhodnutí pokračovat/zapsat:** Na základě tabulky odhadne očekávanou hodnotu dalšího hodu se zbývajícími kostkami. Porovná tuto očekávanou hodnotu se skóre, které by získal okamžitým zapsáním.

Výpočet očekávané hodnoty může být zjednodušený nebo založený na předpočítaných tabulkách pravděpodobností Farkle pro různý počet kostek.

3.3. Agent trénovaný pomocí Reinforcement Learning (RL)

Tento agent byl trénován v připraveném PettingZoo prostředí s využitím definovaného **redukovaného akčního prostoru** a **observačního prostoru**.

- **Algoritmus:** Byl použit jeden ze standardních RL algoritmů vhodných pro diskrétní akční prostory (PPO).
- **Trénink:** Agent byl trénován hraním velkého počtu her proti sobě samému. Cílem bylo naučit se politiku (strategii), která maximalizuje kumulativní odměnu (tj. maximalizuje počet vyhraných her).
- **Strategie:** Naučená strategie by měla reflektovat kompromis mezi riskováním pro vyšší zisky a bezpečným zapisováním bodů, přizpůsobený aktuálnímu stavu hry (skóre, zbývající kostky, skóre soupeře).

4 Experimenty a výsledky

4.1. Metodika testování

Pro porovnání výkonnosti navržených agentů je vytvořena jednoduchá testovací funkce, kde každý typ agenta hraje proti každému jinému typu agenta (včetně sebe sama pro kontrolu). Bylo odehráno velké množství her, aby byly výsledky statisticky významné.

Primární metrika: Win Rate (procento vyhraných her).

Testované páry:

- Naivní vs. Heuristický
- Naivní vs. RL
- Heuristický vs. RL
- Naivní vs. Naivní (očekávaný win rate ~50%)
- Heuristický vs. Heuristický (očekávaný win rate ~50%)
- RL vs. RL (očekávaný win rate ~50%)

4.2. Výsledky

Naivní agent je nejslabší. Jeho strategie je místy příliš riskantní a místy přiliš konzervativní jelikož nijak nezohledňuje bodové ohodnocení.

Heuristický agent naivního agenta výrazně překonal, protože je jeho rozhodování založeno na kalkulovaném riziku. Základní heuristika mu však dovoluje uvažovat pouze jeden dopředu.

RL agent má potenciál dosáhnout nejlepších výsledků. Naivního agenta jednoznačně poráží. Kvůli omezenému tréninku má s heuristickým agentem problém. S důkladnějším trénováním může demonstrovat komplexnější strategie než pevně daná heuristika. Výkon RL agenta silně závisí na kvalitě tréninku, hyperparametrech a dostatečném počtu trénovacích epizod.

5 Závěr

Tato semestrální práce úspěšně demonstrovala návrh a implementaci UI agentů pro hru Farkle. Byla vytvořena funkční implementace hry, standardizované PettingZoo prostředí a tři různé typy agentů: naivní, heuristický a agent trénovaný pomocí posilovaného učení. Byl navržen a zdůvodněn mechanismus redukce akčního prostoru, který umožnil efektivní trénink RL agenta.

Experimentální evaluace ukázala relativní sílu jednotlivých přístupů. Jak se očekávalo, heuristický agent a RL agent výrazně překonali naivního agenta. Porovnání heuristického a RL agenta velmi záleží na zvolené heuristice a kvalitě trénování RL agenta.