Turnaj

Zveme vás na turnaj v umělé inteligenci pro hry! Tento turnaj je inspirován slavným Axelrodovým turnajem ve vězňově dilematu ze 70. let.

Budeme hrát opakovanou symetrickou bimaticovou hru. V této hře jsou na výběr čtyři strategie (A, B, C, D). Níže uvedená bimatice definuje bodové zisky.

\	A	B	C	D
A	6 \ 6	3 \ 1	3 \ 1	3 \ 1
B	1 \ 3	2 \ 2	8 \ 0	8 \ 0
C	1 \ 3	0 \ 8	10 \ 10	1 \ 18
D	1 \ 3	0 \ 8	18 \ 1	4 \ 4

Například pokud si vyberete strategii A, soupeř si vybere strategii B, pak obdržíte 3 body a soupeř obdrží 1 bod. Všimněte si, že pokud se omezíme jen na strategie C, D (viz pravá dolní podmatice velikosti 2x2), pak je hra ekvivalentní vězňově dilematu.

V turnaji je vaším úkolem vytvoření skriptu v Pythonu[1] (potřebná verze Python 3.6+), který bude umět hrát výše popsanou bimaticovou hru.

Implementace

Pravidla pro psaní skriptů jsou snadná. Než si je popíšeme, stáhněte si prosím zdrojové kódy z repozitáře `https://github.com/sauermar/Turnaj-M-M` a nahlížejte do nich současně se čtením tohoto návodu. Nyní se můžeme podívat na jednotlivá pravidla.

Zaprvé skript musí implementovat rozhraní ze souboru `player.py`, to znamená dodržovat podmínky, co musí vracet a co mají k dispozici jednotlivé metody popsané v rozhraní. Přičemž metoda je v Pythonu normální funkce, která se volá s objektem jako s prvním parametrem. Pokud si chcete práci ulehčit, můžete si zkopírovat vzorového hráče `mirror.py` (ten již rozhraní implementuje) a pouze jeho metody upravovat podle své strategie.

Rozhraní obsahuje inicializátor `__init__`, což je funkce, která se zavolá při vytvoření objektu vaší třídy. Tento objekt se bude vytvářet na začátku každého souboje během turnaje, bude se tedy volat i váš inicializátor. Inicializátor se hodí na vytvoření atributů třídy, což jsou víceméně vlastní

[1] Pokud jste se s Pythonem nesetkali, doporučujeme si projít tuto stránku: https://naucse.python.cz/

lokální proměnné objektu, vhodné například na zapamatování předchozího stavu hry. Metodu `author_name` pouze upravte, aby vracela string s vaším (celým) jménem.

Metoda `next_move` vrací následující hráčův tah prostřednictvím typu `Move`. Jedná se o datový typ, pomocí kterého popisujeme možné tahy, tedy řádky (příp. sloupce) matice uvedené výše.

```
class Move(enum.Enum):
    a_safe_way = ("A", 0)
    betray = ("B", 1)
    cooperate = ("C", 2)
    deceive = ("D", 3)
```

Pro použití enumerace tahů je třeba typ `Move` z `player.py` naimportovat do vašeho skriptu. To jde udělat připojením `from player import Move` na začátek zdrojového kódu. Následně jde v kódu enumerace tahů použít dvěma způsoby. Buď se můžete odkázat na název pomocí `Move.cooperate`, nebo na hodnotu použitím `Move(("C",2))`.

A nakonec, skrz metodu `reward`, obdržíte instanci třídy `Result`, odkud je dostupný váš tah, oponentův tah a získané body. Jednotlivé metody použitelné na tomto objektu jsou definované v souboru `result.py`.

Jedná se o metodu `get_my_score`, která na základě zvolených strategií (tahů) vrátí vaše získané skóre, a `get_opp_score`, která stejným způsobem vrací soupeřovo získané skóre. Třída `Result` také obsahuje inicializátor, který k instanci objektu přiřadí atributy `my_move` a `opp_move`, dostupné přes tečku.

Nyní už znáte vše potřebné pro vytvoření vlastního skriptu, a jeho další vylepšení jsou tedy pouze na vás! Pokud by ale stále nebylo něco jasné, k dispozici jsou i jednoduché vzorové skripty hráčů, ze kterých můžete vycházet při implementaci rozhraní. Jedná se o soubory: `always_cooperate.py`, `always_deceive.py`, `unforgiving.py`, `score_counting.py`, `mirror.py`

Sami si proti svému dokončenému skriptu můžete zahrát pomocí souboru `testing.py`, kterému v konzoli zadáte název vašeho skriptu tečka název vaší třídy (která implementuje rozhraní `Player`) jako první argument a případně počet iterací (kol) hry jako druhý. Výchozí počet iterací je nastaven na 10. Nezapomeňte, že váš skript se musí nacházet ve stejné složce, kde je uložen soubor `testing.py`, jinak ho program nedokáže nalézt.

Nakonec, až budete se svým hráčem naprosto spokojeni, stačí nám zdrojový kód skriptu odeslat e-mailem.

Podrobnosti o turnaji

Všechny skripty budou mezi sebou hrát po dobu předem stanoveného počtu kol, jehož hodnotu vám nemůžeme předem prozradit, ale pro představu uvádíme, že to bude číslo mezi 10 a 1000.

V turnaji budou tři druhy hráčů

- soutěžící M&M
- ostatní účastníci turnaje
- 19 defaultních skriptů (které najdete ve složce `trivial`)

Lidští hráči budou mít v turnaji po třech instancích od svých skriptů. Defaultní skripty budou v turnaji každý jen jednou. Všechny instance budou hrát turnaj každý-s-každým a celkový součet bodů ze všech interakcí určí pořadí v tabulce. Do výsledné výsledkové listiny lidských hráčů bude použit medián ze skóre jejich tří instancí.

Z toho plyne, že každý váš skript bude interagovat 3x s každým skriptem napsaným ostatními účastníky, 2x sám se sebou, 1x s každým defaultním skriptem.

Zacházení s osobními údaji

Úplné a podrobné výsledky turnaje (včetně vašeho celého jména) budou sdíleny prostřednictvím našeho mailing listu `hry@mam.mff.cuni.cz`, který ovšem není veřejně přístupný. K vašim zdrojovým kódům budou mít přístup pouze organizátoři turnaje.

Na oficiální internetové stránce Korespondenčního semináře M&M budou uvedeny výsledky pouze se jmény účastníků M&M, kteří nám dali explicitní souhlas se zpracováním osobních údajů. Informace o ostatních soutěžících v turnaji nebudou veřejně dostupné.

Pokud máte jakékoliv dotazy, napište nám na e-mail!
martin.dvorak@matfyz.cz (organizátor)
market.sauer@gmail.com (autorka zdrojových kódů)