Bataille navale

Étude statistique du jeu Bataille Navale

Boudrouss Réda n°28712638

Zhenyao Lin n°28708274

Sorbonne Université
France
20 octobre 2022

Contents

Introduction

L'objectif de ce projet et de faire une étude statistique du jeu "bataille navale". Le jeu consiste en une grille de 10 x 10 cases sur laquelle sont placés 5 bateaux de taille respective 5,4,3,3 et 2. Pour chaque coup, le jouer doit tirer sur une case révélant son état : Raté, touché, ou coulé.

Quelle est donc la meilleur statégie que vas nous permettre de gagner avec un minimum de nombre de coup ?

Description du code

Le code de notre proje s'organise comme suit :

```
Bataille/
├── rapport/
├── src/
│   ├── data/
│   ├── Game/
│   │   ├── Bateau.py
│   │   ├── Engine.py
│   │   └── EngineStats.py
│   ├── Player/
│   │   ├── AbstractPlayer.py
│   │   ├── HeuristicPlayer.py
│   │   ├── HumanPlayer.py
│   │   ├── MCPlayer.py
│   │   ├── ProbPlayer.py
│   │   └── RandomPlayer.py
│   └── utils/
│       └── constants.py
└── main.py
```

Le dossier `rapport/` est le dossier qui contient tout les fichiers qui ont été nécessaires à la création du présent rapport.

Le dossier `src/` contient l'essentiel du code et de no différentes simulation.

Le fichier `main.py` est le fichier qui permet d'executer le code. Pour lancer le programe il faut `cd src/` et ensuite `python main.py`.

Le dossier `data/` contient toutes sortes d'informations sur les différences de données telque les résultats des joueurs, des images ou des données nécessaire à la bonne excécution du programme.

Le dossier `utils/` contient toutes sorte de fonctions qui ont été utiles au projet. Mais il contient aussi et surtout le fichier `constants.py` qui possède toutes les configurations du jeu et le convention de programmation. Hésitez pas à changer des paramètres dessus.

Le dossier `Game` contient les codes de la partie logique du jeu. C'est un sorte de "game engine" (d'où le nom `Engine.py`). `Engine.py` pour le jeu en lui même et `Bateau.py` pour le code de l'objet "Bateau" qui nous permet de facilité et centralisé notre code pour le bateau. `EngineStats.py` a les mêmes caractéritiques que `Engine.py` mais avec quelques fonctions assez techniques en plus. C'est ici que vous trouverais quelques une des fonctions demandés dans le sujet.

Combinatoire du jeu

Nous avons calculé à la main le nombre de façon possible de disposer chacun des bateaux sur une grille de 10x10.

Bateau	Taille	nb
Porte avion	5	120
Croiseur	4	140
Contre-torpilleurs	3	160
Sous-marin	3	160
torpilleur	2	180

Ce qui nous donne alors comme nombre de configuration maximal théorique : 7.74×10^{10}.

Les fonctions qui traitent du combinatoire se trouvent dans le dossier `src/utils/comb.py`. L'utilisation du bout de code suivant nous permet de trouver la même valeurs que nous résultats théoriques.

```
nb_tot: int = 1
for i in range(1,len(LEN_B)+1):
    nb_tot *= nb_placer(i)
```

Cette valeur cepandant inclue les tableaux où les bateaux peuvent se superposer ce qui n'est pas possible dans les règles du jeu. Pour avoir une valeur plus précise, nous devons ignorer tout ces plateaux.

La fonction `nb_placerL` fonction (*naïve toujours*) marche très bien pour des petites listes de bateaux ou pour des petites grilles, mais pour notre cas le temps d'excecution est énorme, même en optimisant avec les symètries cela prendrais toujours trop de temps, notre fonction se rapproche d'une complexité $O((3n)^m)$ avec n la taille de la dimension du plateau (supposé carré) et m le nombre de bateau.

Python n'est pas le langage pour ce genre de calcul intensif, on sait que notre résultat doit être aux alentours de 10^{10}. Le bout de code simplite suivant prend déjà trop longtemps à s'excécuter :

```
i = 0
while i < 1e10:
    i+=1
```

Essayons une autre approche, en supposons toutes les grilles équiprobable, on a donc : $P(\text{table}_n) = \frac{1}{N}$ avec $P(\text{table}_n)$ la chance de tirer une table aléatoire et N le nombre de table au totale.

Si on prend une fonction qui prend en paramètre une grille et génère des grilles aléatoirement jusqu'à ce que la grille générée soit la même que celle passé en paramètre. Alors dans ce cas le nombre de grille généré sera une approximation de nombre totale de tableau