LU3IN005-Probas, Stats et Info

19/10/2022

Rapport de TP -Bataille navale-

Présenter par :

HAMI ISLAM	21208634
DJERFAF ILYES	21215141

Groupe : 02-*binome 10*

1-*Modélisation et fonctions simples:*

Dans cette partie on définit notre liste de bateau en affectant à chaque bateau le nombre de case qu'il occupe et on donne la main à l'utilisateur pour choisir la taille de la grille (minimum 10x10).

Les fonctions à implémenter sont assez claires et bien detaillées dans le code.

2-*Combinatoire du jeu:*

1. Nombres de configurations possible:

En prenant le nombre maximum de configuration possible pour l'emplacement de chaque bateau de la liste [2, 3, 3, 4,5] sur une grille de taille 10 on aura N= 180 x 160 x 160 x 140 x 120 = 77414400000 avec le nombre de configuration pour placer un bateau de longueur i sur une grille vide de taille n est (n-i+1) × n × 2.

2. On constate que le résultat de la fonction renvoit exactement le même résultat que le théorique.

3. Nombres de configurations exactes:

Liste bateaux [5] =>180

Liste bateaux [5,3]=>27336

Liste bateaux[5,3,3]=> 3848040

Remarque: Plus la liste est grande , plus le temps d' éxecution augmente drastiquement.

4. Si il y'a un nombre N de grille équiprobables possible , alors la probabilité p de tirer une grille donnée à chaque tirage est de : p= 1/N.

5. Il suffit de calculer la probabilité p d'avoir une grille donnée et appliqué la formule N = 1 / p, et pour celà on calcule le nombre de grille générer pour avoir la même grille pour un nombre d'itération (dans notre cas 10

itération) et diviser le résultat sur le nombre d'itération pour avoir une approximation de la probabilité d'où le nombre de grille possible.

Pour une liste [5] : 169

Pour une liste [5,3] : 29214

Pour une liste [5, 3, 3] : 2405116

Comme on remarque ci-dessus l'approximation pour une liste de 1 ou 2 bateaux est proche du résultat exacte par contre plus la liste augmente plus l'approximation et le resultat exacte s'eloigne énormément , et ça peut être expliqué par le fait que le nombre de grille générées à chaque fois peut être très inférieur ou très supérieur au nombre exacte d'où la moyenne sera loin du résultat attendu, la solution est de tirer aléatoirement pour un grand nombre d'itération mais le temps d'exécution sera très long. Alors cette méthode n'est pas efficace pour calculer le nombre de configuration.

3-*Modelisation probabiliste du jeu:*

Dans cette partie on définit 2 classes:

Classe Bataille: représente la partie et contient les fonctions nécessaire au déroulement du jeu.

Classe Joueur: représente les différentes stratégies utilisables par un joueur.

1. Version aléatoire:

THEORIQUE:

On suppose tout d'abord que les tires sur les cases sont équiprobables , on ne peut pas tirer 2 fois sur la même case.

En se basant sur les hypothèses ci-dessus on calcule la probabilité Pk de gagner si on tire k fois :

-Soit (Ptouché)i = nombre de cases de bateau restantes/nombre de cases restantes dans la grille la probabilité de toucher un bateau au i-ème tire

-(PnonTouché)i = 1 - Ptouché = nombre de cases vides restantes/nombre de cases restantes dans la grille la probabilité de ne pas toucher un bateau au i-ème tire

-Pk = ((PnonTouché)1 ∩ (PnonTouché)2∩ …….. ∩ (Ptouché)k-1 ∩(Ptouché)k) ∪ ((PnonTouché)1 ∩ (PnonTouché)2∩ …….. ∩ (PnonTouché)k-1 ∩ (Ptouché)k) ∪ …………∪ ((Ptouché)1 ∩ (PnonTouché)2∩ …….. ∩ (PnonTouché)k-1 ∩ (Ptouché)k) ∪ ((Ptouché)1 ∩ (Ptouché)2∩ …….. ∩ (PnonTouché)k-1 ∩ (Ptouché)k)

- Soit X la variable aléatoire qui représente le gain en n = 1,…,100 fois :

P(X = n) = (PnonTouché)1 ∩ (PnonTouché)2∩ …….. ∩ (Ptouché)r

Comme les événements de toucher une case sont indépendants donc:

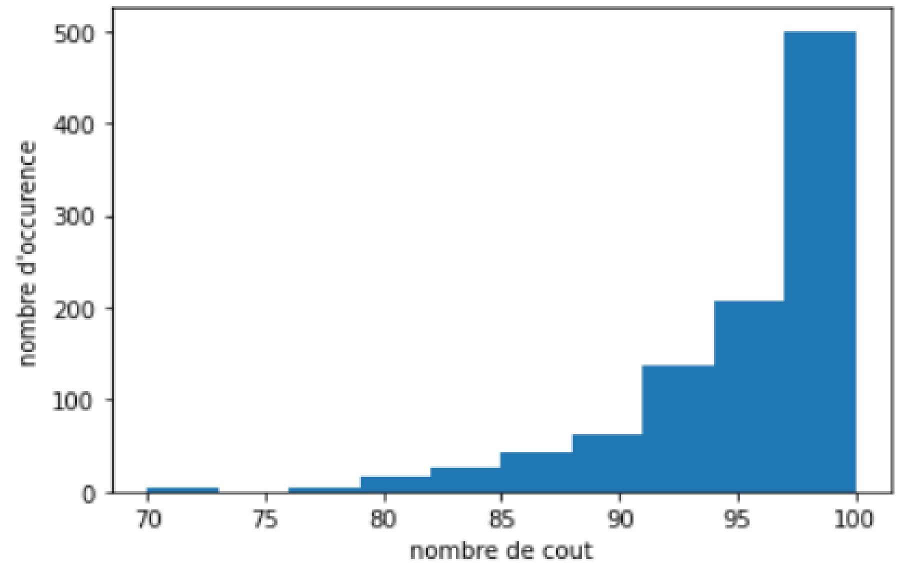
P(X = n) = (PnonTouché)1 × (PnonTouché) × ……. × (Ptouché)r

On a l'Espérance de X: E(X) =$\sum_{i=1}^{100} x_i \cdot P(X = x_i)$.

-On faisant les calculs on trouve E(X)=89.37.

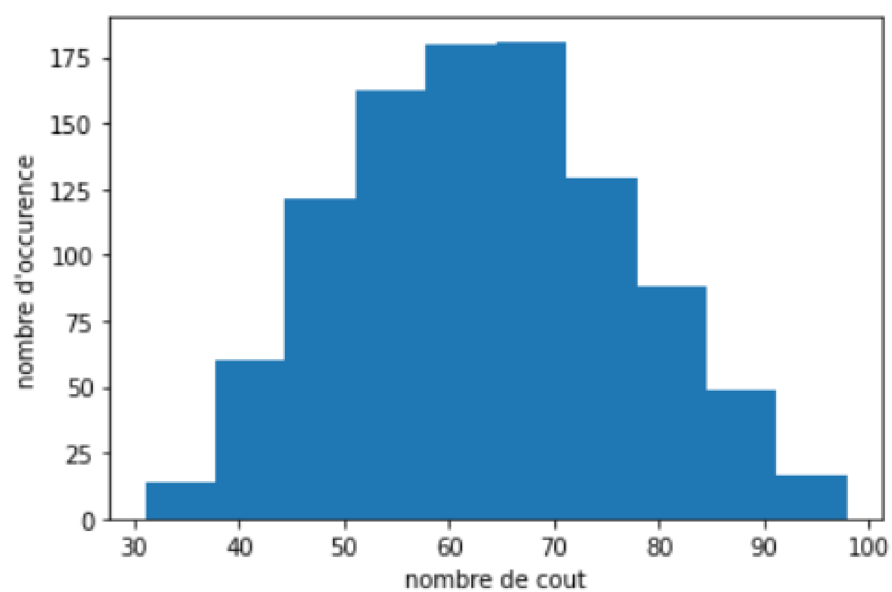
EXPERIMENTAL:

-Après avoir fait un grand nombre de test, on trouve E(X)=95.38. En comparant avec le résultat théorique, l'expérimental est légèrement plus grand car expérimentalement tout dépend de la fonction de random.



2-**Version heuristique:**

En calculant l'espérence , on trouve E(X)=78.65. Donc on déduit que cette nouvelle stratégie est meilleure et plus efficace que la première, on minimise les coups joués de casiment un tiers.



3-**Version probabiliste simplifiee:**

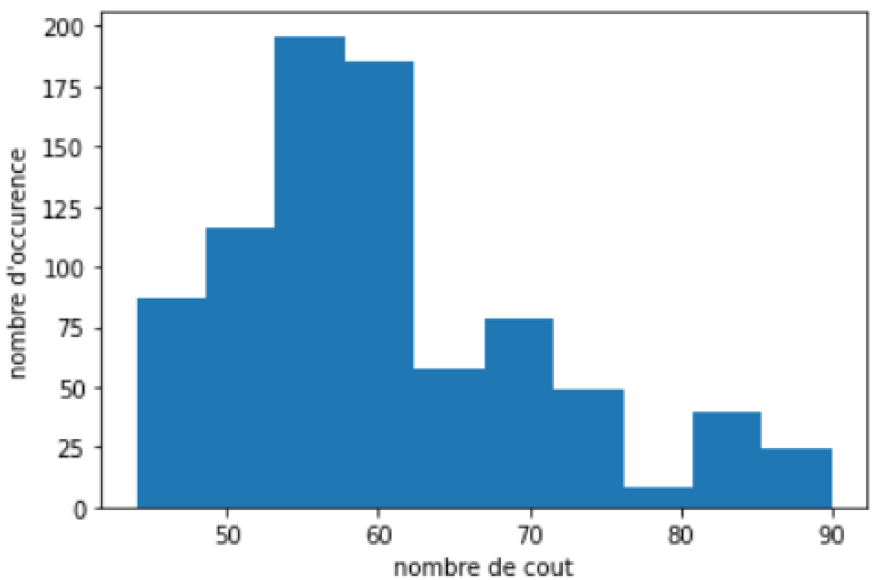
-Il est hors de question de calculer la distribution jointe sur tous les bateaux car ça sera compliqué d'énumérer toutes les grilles où un bateau se trouve dans une case donnée.

- Dans la méthode proposée pour chaque case on obtient le nombre de fois où le bateau bi apparaît potentiellement, en divisant par le nombre de configurations possible pour ce bateau dans une grille vide on obtient P(bi) la probabilité que bi soit dans la case (i,j).

Par contre cette hypothèse est fausse, car les événements de la présence d'un bateau dans une case donnée ne sont pas indépendants.

-La solution proposée est detaillée dans le code avec des commentaires.

-En calculant l'espérence on obtient E(X)=56.88, ce qui représente la meilleure stratégie parmis les 3.



4- *Senseur imparfait : à la recherche de l'USS Scorpion:*

1-On essaye de formaliser les diffèrents cas de probabilités présente :

.La probabilité de détecter un objet sachant qu'il se trouve dans la case

=> $P(Z_i = 1|Y_i = 1|)$= p_s (dans l'enonce)

.La probabilité de ne pas détecter un objet sachant qu'il se trouve dans la case

=> $P(Z_i = 0|Y_i = 1|)$= $1 - p_s$ (dans l'enonce)

.La probabilité de ne pas détecter un objet sachant qu'il ne se trouve pas dans la case => $P(Z_i = 0|Y_i = 0|)$=1 (car si l'objet ne se trouve pas dans la case, dans tout les cas il ne sera pas détecté)

.La probabilité de détecter un objet sachant qu'il ne se trouve pas dans la case

=> $P(Z_i = 1|Y_i = 0|)$=0 (cas impossible car si l'objet ne se trouve pas dans la case le senseur ne peut pas le détecter)

2- La Loi de Y_i: Suit une loi de bernouli B(1,π_i) tel que =>

.$P(Y_i = 0) = 1 - \pi_i$.$P(Y_i = 1) = \pi_i$

Loi de Z_i/Y_i: Les probabilités qui définient cette loi sont (cf.1) =>

.$P(Z_i = 1|Y_i = 1|)$=p_s .$P(Z_i = 1|Y_i = 0|)$=0

.$P(Z_i = 0|Y_i = 0|)$= 1 .$P(Z_i = 0|Y_i = 1|)$=$1 - p_s$

3- La probabilité qu'un sous marin se trouve en case k et un sondage est effectué mais ne détecte pas le sous marin.

Cette probabilité doit être exprimer par une intersection =>

$$P(Z_k = 0 \cap Y_k = 1)$$

. En utilisant la loi de la probabilité conditionnelle on a :

$$P(Z_k = 0 \cap Y_k = 1) = P(Z_k = 0|Y_k = 1|) * P(Y_k = 1)$$

$$=(1 - p_s)*\pi_k$$

4- Pour faire la mise à jour de π_k :

a- cas i=k:

$$\pi_{k'}=\left(\pi_{k'} = \pi_k \cdot (1 - p_s)\right)/(1 - \pi_k \cdot p_s)$$

b- cas i≠k:

$$\pi_{k'} = \pi_k/(1 - p_s \cdot \pi_k)$$

5- **Implémentation:**

.On a 2 cas possible dans cette implémentation :

A-Le cas où on ne connait pas l'objet perdu , la probabilité π_i est identique dans toute les cases, donc il suffit de parcourir la grille autant de fois qu'il faut jusqu'à ce que le senseur détecte l'objet.

B-Dans ce cas, on suit la logique du sous-marin qui diffuse la probabilité beaucoup plus dans les cases profondes. On a décidé de suivre les étapes suivante pour l'implémentation du code:

-On crée une grille qu'on remplira avec les probabilités π_i.

-On cherche la case avec la probabilité la plus élevé.

-On vérifie si le senseur détecte l'objet si il est présent.

-On fait la mise à jour de la probabilité de la case avec la plus haute probabilité en la diminuant et on augmente les probabilités des autres cases.

-On incrémente le nombre d'essaie et on refait les mêmes test avec les nouvelles probabilités jusqu'à ce que le senseur détecte l'objet.