J'accuse! Prise 1

Les gens du CO des CSGames 2024 ont trimé dur pour créer les épreuves, les horaires, la logistique, etc. Afin de se détendre et de se récompenser, ils décident de jouer à un vieux jeu classique, CLUE.

Malheureusement, Colonel de Lachevrotière a oublié le boîtier chez lui. Déterminé à ne pas jouer à un autre jeu de société, Mrs Prevot a alors une idée: elle propose un nouveau jeu qui recrée l'esprit du jeu CLUE.

Configuration du jeu

C'est un jeu à un contre un, mais on a besoin d'une autre personne (le croupier) pour configurer l'état initial du jeu. Les autres membres du CO ne pourront qu'être spectateurs.

Le jeu se joue avec une petite enveloppe et un paquet de cartes réduit (plus précisément, on prend seulement les cartes de coeur d'un paquet normal, donc 13 cartes).

Le croupier effectue les opérations suivantes, hors de vue des deux joueurs:

- Il brasse aléatoirement le paquet
- Il choisit au hasard deux cartes du paquet. Il les met dans l'enveloppe. Ces cartes seront appelées les cartes coupables.
- Parmi les cartes restantes, il choisit aléatoirement 5 cartes qu'il donne au joueur 1.
- Il donne les 6 cartes restantes au joueur 2

Évidemment, comme dans le vrai jeu Clue, toutes les cartes ont été face cachée pendant ces opérations, donc aucun joueur ne sait ce qu'il y a dans l'enveloppe ou dans la main de l'autre joueur.

Règles du jeu

Les joueurs jouent chacun leur tour. Le joueur 1 est le premier à jouer. À son tour, un joueur choisit l'UNE des deux actions suivantes:

- Pige: **le joueur pige une carte au hasard dans la main de son adversaire.** Évidemment, l'adversaire peut brasser ses cartes derrière son dos avant la pige pour être certain que le joueur qui pige pige bel et bien au hasard. **Ensuite, le joueur ajoute la carte pigée à sa main.**
- Accusation: **le joueur dit "J'accuse" en nommant les deux cartes qu'il croit être dans l'enveloppe.** À ce moment, la partie se termine. On ouvre l'enveloppe. Le joueur l'emporte si son affirmation est exacte, sinon l'adversaire l'emporte.

Il n'y a pas de limite au nombre de tours que peut durer la partie. En effet, si vous avez bien compris, un joueur pige toujours dans une main contenant 6 cartes. Il peut être vraiment malchanceux et souvent piger une carte qu'il vient tout juste de se faire retirer (ce qui ne donne aucune information nouvelle).

Votre mission

Calculez la probabilité que le joueur 1 remporte la partie, en supposant que les deux joueurs ont une excellente mémoire et jouent de manière optimale (c'est-à-dire tenter l'accusation seulement si c'est le meilleur choix).

Flag

Le ficher 'flag-steps-fake.txt' sert d'exemple pour vous indiquer comment créer le FLAG. À une exception près, reproduisez exactement les étapes dans 'flag-steps-fake.txt'.

L'exception est bien sûr que vous devez utiliser la bonne probabilité pour faire ces étapes, au lieu de $\frac{175}{375}$ fournie en exemple.

J'accuse! prise 2

Il semble que la version précédente du jeu ne satisfait pas le joueur 1. En raison du calcul que vous avez fait précédemment, il se plaint que ses chances de victoire ne sont pas assez près de 50% à son goût.

Pour s'en approcher davantage il propose les changements de paramètres suivants:

- On joue avec tous les coeurs ET tous les piques, donc 26 cartes, 2 fois plus qu'avant!
- Le croupier met 3 cartes aléatoires dans l'enveloppe au lieu de 2. Ainsi, un joueur qui accuse doit deviner les 3 cartes.
- Les joueurs reçoivent respectivement 11 et 12 cartes au lieu de 5 et 6
- Toutes les autres règles sont inchangées.

Cependant, avec ces nouveaux paramètres, les autres membres du CO ont peur de s'endormir. Ils prétendent qu'ils vont s'endormir si les deux joueurs jouent plus de 25 tours chacun.

Votre mission

À l'instar du numéro précédent, les joueurs jouent encore de manière optimale. Hors de question d'accuser seulement pour faire plaisir aux spectateurs qui vont s'endormir!

Calculez la probabilité que le nombre de piges atteigne au moins 50. Une pige est le fait de prendre une carte au hasard dans la main de l'adversaire. Par exemple, si les 2 joueurs font 25 tours chacun sans s'accuser, cela fait 50 piges.

Flag

Même format que le flag précédent

Solutionnaire

Pour se comprendre, nommons le joueur courant le joueur à qui est le le tour en ce moment. L'autre joueur est l'adversaire.

En cours de partie, on peut représenter l'état actuel par la paire ordonnée (i, j) où:

- i: Le nombre de cartes de la main de l'adversaire qui sont inconnues du joueur courant
- j: Le nombre de cartes de la main du joueur courant qui sont inconnues de l'adversaire

D'abord, les piges sont aléatoires, Donc, côté tactique, le seul élément stratégique du jeu est de décider quand tenter une accusation.

Le joueur courant, s'il tente une accusation tout de suite, doit deviner les deux cartes coupables parmi $2+i$ cartes, car il y a $2+i$ cartes qu'il n'a pas vues. De son point de vue, il y a $C(2, i+2)$ paires possibles de cartes coupables, donc s'il accuse, il a seulement $\frac{1}{C(2,i+2)}$ de gagner. Ce qui vaut au mieux $\frac{1}{3}$ si $i > 0$.

L'adversaire est dans la même situation. Si le joueur courent n'accuse pas, l'adversaire aura $\frac{1}{C(2,j+2)}$ chances de gagner s'il accuse à son tour. Conclusion, le joueur courant n'a pas intérêt à tenter une accusation si $i > 0$ et $j > 0$, car l'adversaire aurait moins de $\frac{1}{2}$ de réussir la sienne à son tour.

Le seul cas où le joueur courant, dans l'état (i, j), a intérêt à faire une accusation, est si $i = 0$ ou $j = 0$. Si $j = 0$, ça veut dire que l'adversaire vient tout juste de voir la dernière carte non-coupable qu'il n'avait pas encore vue; puisqu'il a 100% de chances de réussir une accusation à son prochain tour, il faut accuser tout de suite avant qu'il puisse jouer son tour. De plus, selon ce raisonnement, vu que l'adversaire pense pareil, le cas $i = 0$ ne survient jamais puisque l'adversaire va accuser avant que ce soit notre tour.

Équations

Nommons $P_{i,j}$ la probabilité que le joueur courant remporte la partie si l'état actuel est (i, j), respectivement.

Comme expliqué plus tôt, on a:

$$P_{i,0} = \frac{1}{C(2, i+2)}$$

On a aussi, trivialement:

$$P_{0,j} = 1$$

Mais cette dernière équation est inutile car l'adversaire ne nous laissera jamais jouer notre tour si on connaît les coupables.

Les équations se complexifient lorsque $i > 0$ et $j > 0$. Soit $N = 6$, le nombre de cartes dans la main de l'adversaire avant que le joueur courant pige dedans. Dans ce cas, le joueur courant pige une carte aléatoire dans la main de son adversaire. Il a $\frac{i}{N}$ de piger une carte qu'il n'avait jamais vue auparavant. Dans ce cas, l'adversaire, à son tour, sera dans l'état $(j, i-1)$ et aura $P_{j,i-1}$ chances de l'emporter. En contrepartie, le joueur courant a $\frac{N-i}{N}$ de piger une carte qu'il avait déjà vue auparavant. Dans ce cas, l'adversaire, à son tour, sera dans l'état $(j, i-1)$ et aura $P_{j,i-1}$ chances de l'emporter. On a donc:

$$P_{i,j} = 1 - (\frac{i}{N} P_{j,i-1} + \frac{N-i}{N} P_{j,i})$$

Super! Allons coder un programme qui calcule récursivement tous les $P_{i,j}$! Pas si vite... pour appeler une fonction récursive sans boucler, il faut éviter les appels circulaires. Le problème, c'est que $P_{j,i}$ cause problème dans l'équation, car, si on le développe:

$$P_{j,i} = 1 - (\frac{j}{N} P_{i,j-1} + \frac{N-j}{N} P_{i,j})$$

Bref, le calcul de $P_{i,j}$ fait appel à $P_{j,i}$ qui fait appel à $P_{i,j}$... on boucle, il faut donc développer et réisoler $P_{i,j}$:

$$\begin{aligned}
P_{i,j} =& 1 - (\frac{i}{N} P_{j,i-1} + \frac{N-i}{N} P_{j,i}) \\
=& 1 - (\frac{i}{N} P_{j,i-1} + \frac{N-i}{N}(1 - (\frac{j}{N} P_{i,j-1} + \frac{N-j}{N} P_{i,j})) \\
=& 1 - \frac{i}{N} P_{j,i-1} - \frac{N-i}{N}(1 - (\frac{j}{N} P_{i,j-1} + \frac{N-j}{N} P_{i,j})) \\
=& 1 - \frac{i}{N} P_{j,i-1} - \frac{N-i}{N} + \frac{j(N-i)}{N^2} P_{i,j-1} + \frac{(N-j)(N-i)}{N^2} P_{i,j} \\
N^2 P_{i,j} =& N^2 - N P_{j,i-1} - N(N-i) + j(N-i) P_{i,j-1} + (N-j)(N-i) P_{i,j} \\
=& Ni - N P_{j,i-1} + j(N-i) P_{i,j-1} + (N^2 - Ni - Nj + ij) P_{i,j} \\
(Ni + Nj - ij) P_{i,j} =& = Ni - N P_{j,i-1} + j(N-i) P_{i,j-1}
\end{aligned}$$

Et ainsi, on a:

$$P_{i,j} = \frac{Ni - N P_{j,i-1} + j(N-i) P_{i,j-1}}{(Ni + Nj - ij)}$$

Cette équation est mûre pour de la programmation récursive, car il n'y a aucun risque de d'appels circulaires. En effet, le calcul de $P_{j,i-1}$ ou $P_{i,j-1}$ ne peut pas éventuellement faire appel à $P_{i,j}$, car le nombre total de cartes non-vues par les deux joueurs ne peut pas augmenter en cours de route.

La valeur que nous cherchons, au final, est $P_{N,N-1} = P_{6,5}$.

On obtient $P_{6,5} = \frac{6557099}{13634544}$

Code

Il suffit d'implémenter l'équation récursive ci-dessus, soit par une fonction récursive compute_p(i,j), soit en utilisant la programmation dynamique pour remplir progressivement un tableau 2D de tous les $P_{i,j}$. La programmation dynamique est nettement plus efficace, car des appels récursifs vont répéter très très très souvent les mêmes calculs. Cependant, les paramètres sont assez petits que l'utilisation d'une fonction récursive sans programmation dynamique fonctionne en moins de 5 secondes.

Voir le code fourni avec ce document. Pour la programmation dynamique, un tableau en 2D est utilisé pour sauvegarder $P_{i,j}$ pour chaque paire (i, j) possible. Le remplissage du tableau itère des plus petites valeurs de $i + j$ vers les plus grosses.

Flag

FLAG{6557099/13634544}

Prise 2

Cette question est conceptuellement plus facile, car il ne faut pas faire d'acrobaties algébriques pour éliminer un risque de références circulaires.

On rappelle que pour ce problème, $N = 12$, car les paramètres ont changé. Chaque joueur pige dans une main qui contient 12 cartes, et non 6.

Appelons $P_{k,i,j}$ la probabilité que, si on est dans l'état (i, j), il y ait exactement k piges dans le reste de la partie. La partie est dans l'état $(N, N-1)$ au début. Ainsi, la probabilité que la partie compte au moins 50 piges est

$$1 - \sum_{k=0}^{49} P_{k,N,N-1}$$

Il faudra encore établir des équations de récurrence pour trouver la valeur de chacun de ces termes, comme au numéro précédent.

Tout d'abord, pour des raisons tactiques expliquées au premier numéro, il n'y aura plus de piges si $i = 0$ ou $j = 0$ donc, si $i = 0$ ou $j = 0$, $P_{k,i,j} = 1$ pour $k = 0$ et $P_{k,i,j} = 0$ pour $k > 0$.

Par contre, si $i \geq 1$ et $j \geq 1$, alors il faut piger. Après la pige, le joueur courant et l'adversaire sont interchangés, alors on tombe soit dans un état $(j, i-1)$, avec probabilité $\frac{i}{N}$, ou dans un état (j, i) avec probabilité $\frac{N-i}{N}$. Dans les deux cas, il faut que le reste de la partie se termine en exactement $k-1$ piges. Ainsi,

$$P_{k,i,j} = \frac{i}{N} P_{k-1,j,i-1} + \frac{N-i}{N} P_{k-1,j,i}$$

Code

Si on implémente directement l'équation de récurrence ci-dessus, il n'y a pas de risque de référence circulaire.

Par contre, utiliser uniquement une fonction récursive pour calculer chacun des termes de $1 - \sum_{k=0}^{49} P_{k,N,N-1}$ est interminable, alors la programmation dynamique est une nécessité.

Dans le code fourni, un tableau en 3 dimensions est utilisé pour calculer les valeurs de $P_{k,i,j}$ pour chaque triplet (k, i, j) possible. Le tableau est rempli en itérant des plus petites valeurs de k vers les plus grosses.

La probabilité d'atteindre 50 piges est proche de 67%, voir la valeur exacte ci-dessous.

Flag

FLAG{189757578277392553919070387842281595848 9/28352074999403673815253772280382766448 64}

Lien pour le code

https://github.com/desharnc27/incoming-csg-2024-python