Université Gustave Eiffel

Projet de mathématiques appliquées

2048

Cloé Quirin *&* *Vincent* Scavinner

supervisé par
Miguel Martinez

31 mai 2021

Résumé

Rapport de projet documentant la réflexion et les choix d'implémentation réalisés dans le cadre de la conception d'une version revisitée du jeu 2048, utilisant des lois de probabilités mathématiques.

Table des matières

Introduction

Cela fait maintenant quelques années que le jeu du 2048 a pris d'assaut Internet. Dans le monde entier, des milliers de personnes ont passé des millions d'heures à essayer de créer la tuile 2048.

Outre le côté addictif du jeu, il présente également une possibilité d'explorer les mathématiques. Dans le cadre d'un projet réalisé au cours d'une seconde année d'ingénieur à l'IMAC (Univeristé Gustave Eiffel, Champs-sur-Marne, France), ce document tente de démontrer la manière dont les théories et lois probabilistiques ont été appliquées au jeu original pour l'étendre et le renouveler dans une version revisitée.

Règles

Le 2048 est un jeu *"puzzle block"* développé par Gabriele Cirulli. C'est un jeu joué sur une grille 4x4 avec des tuiles numérotées 2^n où n représente un nombre naturel. L'objectif du jeu est de combiner des tuiles du même nombre pour finalement former le nombre 2048.

L'utilisateur peut se déplacer dans les quatre directions cardinales et après chaque mouvement, une nouvelle tuile est générée aléatoirement dans la grille qui est numérotée 2 ou 4. Un mouvement est considéré comme "légal" si au moins une tuile peut être glissée dans un emplacement vide ou si les tuiles peuvent être combinés dans la direction choisie. Le jeu se termine lorsque l'utilisateur n'a pas de mouvement "légal" à gauche.

Aspect mathématique de la version classique

Après avoir fait un mouvement, une nouvelle tuile est placée sur le plateau. Celle-ci est placée aléatoirement sur un emplacement vide de la grille. Cette nouvelle tuile peut avoir pour valeur un 2 ou un 4, là encore choisi aléatoirement. La valeur 2 a 90% de chance de tomber, tandis que la valeur 4 en a seulement 10. Le jeu continue ensuite jusqu'à ce qu'il n'y ait plus de mouvements possibles.

Outre cet aspect probabiliste, le coeur du jeu et de son intéractivité repose sur une combinaison de transformations matricielles relative à la grille. Néanmoins, il est possible de procéder à l'ensemble des mouvements nécessaires par un enchaînement de rotations matricielles. C'est ce que nous avons choisi de faire dans le cadre de notre implémentation.

Réinterprétation de l'original

Notre reviste du jeu classique se base essentiellement sur l'intégration de nouveaux types de tuiles.

Il y a bien évidemment la tuile classique, qui possède la valeur 2 ou 4, chacune ayant une chance indépendante d'apparaître. Mais nous avons intégrer trois types de tuiles supplémentaires : la tuile obstacle, mystère et bonus.

La tuile obstacle est associée à un nombre de mouvements. Elle ne peut en aucun cas être fusionnée avec une autre tuile et son but est donc de vous bloquer dans certains cas. Elle disparaîtra seulement lorsque vous aurez effectuer le nombre de mouvements requis.

La tuile bonus peut se fusionner à n'importe quelle autre tuile présente sur le plateau (même un obstacle). Pour cela, elle s'adapte et adopte la valeur de la tuile cible.

La tuile mystère représente une tuile de valeur, 2, 4, 8 ou 16. Seulement, il est impossible de le déterminer car la valeur est cachée. Il faut la jouer et espérer tomber sur une tuile à la valeur correspondante. Rassurez-vous, la valeur finie par se révéler après un certain nombre de mouvements.

.

Les composantes aléatoires utilisées

Le type de tuile

Espace d'état : Type de la nouvelle tuile.
Loi choisie : Loi géométrique.
Pourquoi : C'est une loi n'ayant que deux issues possibles.
Paramètres : Plus ou moins de chance de tomber sur des cellules spéciales qui peuvent aider ou bloquer le joueurs.
Comment : On effectue un tirage aléatoire d'un nombre entre 0 et 1. Si cette valeur est inférieure à la valeur calculée par la loi géométrique, c'est un succès et la case est spéciale. Sinon, c'est un échec et la case est normale.

$$p(X = k) = (1 - p)^{k-1}p$$

Le type de la tuile spéciale

Espace d'état : Type la tuile spéciale.
Loi choisie : Loi uniforme.
Pourquoi : Ici on cherche seulement à savoir quel type de tuile spéciale va apparaître sur la grille. Tous les secteurs sont de même taille et ont la même probabilité.
Paramètres : Le jeu devient plus ou moins compliqué en fonction du type de case qui apparaît.
Comment : La variable aléatoire suit une loi uniforme sur l'intervalle [0 ;3]. On effectue un tirage aléatoire d'un nombre entre 0 et 1. En fonction du tirage, une des trois cases spéciales est sélectionnée.

Valeur de la tuile classique

Espace d'état : Valeur de la tuile classique.
Loi choisie : Loi de Bernoulli.
Pourquoi : La valeur de la cellule classique n'admet que 2 issues : 2 ou 4. Si on tombe sur 2 c'est un succès, si on tombe sur 4 c'est un échec.
Paramètres : Plus ou moins de chance de tomber sur des cellules de valeur 4, rendant le jeu plus ou moins simple pour le joueur.
Comment : On effectue un tirage aléatoire d'un nombre entre 0 et 1. Si ce nombre est inférieur au paramètre p, c'est un succès et la nouvelle case à pour valeur 2. Sinon c'est un échec est la nouvelle case à pour valeur 4.

$$P(X = 1) = p, P(X = 0) = 1 - p$$

Tuile obstacle

Espace d'état : Nombre de mouvements avant que la tuile obstacle ne disparaisse.
Loi choisie : Loi de Poisson.
Pourquoi : Ici, on veut compter le nombre d'événement en ne connaissant que le nombre moyen de ceux-ci.
Paramètres : Plus ou moins de chance que la case obstacle reste longtemps sur la grille.
Comment : k correspond au nombre de mouvement que le joueur doit effectuer avant que la case ne disparaisse et , un paramètre modifiable par le joueur, correspond au nombre de mouvements moyen. On effectue des tirages aléatoires d'un nombre entre 0 et 1. On compte alors le nombre de tirage qu'il faut avant que le nombre tiré soit inférieur au calcul de la probabilité. Si c'est inférieur, c'est un succès et la case disparaît.

$$p(X = k) = e^{-\lambda}\frac{\lambda^k}{k!}$$

Tuile mystère

Espace d'état : Nombre de mouvements avant que la tuile mystère ne soit révélée.
Loi choisie : Loi binomial.
Pourquoi : On réitère n fois un tirage de Bernoulli de manière indépendante.
Paramètres : Plus ou moins de chance que la case mystère soit révélé.
Comment : n, un paramètre modifiable par le joueur, correspond au nombre de tirages et on fixe k à 1 car on souhaite obtenir un succès.

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

Technologies

Vue/TypeScript

Nous avons décidé d'utiliser le *Framework Vue* mais dans une version typée grâce au langage de programmation *TypeScript*. Nous avons complété l'implémentation en utilisant du développement objet.

Vue (prononcé comme le terme anglais *view*) est un *framework* évolutif pour construire des interfaces utilisateur. À la différence des autres *frameworks* existants, *Vue* a été conçu et pensé pour pouvoir être adopté de manière incrémentale. Le cœur de la bibliothèque se concentre uniquement sur la partie vue/*front*, et il est vraiment simple de l'intégrer avec d'autres bibliothèques ou projets existants.

Contrairement à *JavaScript*, le code écrit en *TypeScript* est beaucoup plus fiable et facilement refactorable. Cela permet d'éviter les erreurs et de procéder à de la réécriture plus facilement. La mise en place de types permet d'éviter des erreurs classiques et redondantes dans un code JavaScript classique, et permet de se concentrer directement dessus afin d'éviter leur accumulation et le risque de bug majeur.

TypeScript permet également de s'intéresser plus en profondeur à la manière dont le système fonctionne vraiment. Il permet de rendre abstrait des parties récurrentes afin de les réutiliser.

Néanmoins, même si on peut facilement défendre l'idée d'utiliser *TypeScript* dans un projet qui s'étend dans le temps, il n'en est pas de même pour un projet à court terme comme le nôtre. En effet, typer correctement un projet demande beaucoup de temps, beaucoup de minutie et beaucoup de réflexion.

Librairies personnalisées

Nous avons fait le choix de créer notre propre librairie mathématique, chargée de nous fournir facilement les distributions et lois dont nous avions besoin. Cela nous a permis de réaliser un couplage faible avec le reste de notre jeu, mais aussi de préparer l'éventualité d'une évolution de l'application avec de nouvelles probabilités. Notre jeu possède donc son propre *Manager*, chargé de faire les appels à la librarie.

En plus des probabilités, le jeu repose également sur des transformations matricielles. Nous avons alors créé notre propre fonction utilitaire afin de réaliser ces transformations le plus simplement possible. Cette fonction a été implémentée de manière générique grâce à TypeScript.

Enfin, au cours d'une partie, les structures de données chargées de stocker les états relatifs au différentes tuiles (couramment en jeu ou celles déjà jouées) peuvent potentiellement atteindre des tailles importantes. Pour optimiser les recherches, nous avons décidé de réinterpréter la fonction de filtre (*Array.filter()*) sur tableaux de *JavaScript*. Après quelques tests, notre implémentation s'avère en moyenne 60% plus rapide. Nous en avons aussi profiter pour écrire une fonction de filtre mais applicable à des objets. Pour cela, nous avons mis en place le principe de *currying*.

Nous avons mis en place des tests sur chacune des libraries pour vérifier la cohérence des résultats retournés par les fonctions. Ces tests sont notamment primordiaux sur les retours des méthodes des lois de distribution. Ainsi nous avons pu mettre en place des tests pour vérifier que les résultats retournés par nos lois uniformes, nos lois de Poisson et nos lois de Bernouilli étaient corrects. Pour cela, nous vérifions que la valeur de retour est bien supérieur au minimum attendu et inférieur au maximum attendu, mais nous vérifions également si la médiane et la variance sont bien proches de celles atendues (avec un degré d'erreur calculé en fonction du nombre d'essais).

Conclusion

Nous espérons avoir réussi à montrer que derrière un simple jeu, il y a un monde mathématique à explorer. Les concepts étudiés ce semestre et les ressources consultées au cours du projet nous ont aidés à analyser le jeu et son fonctionnement, mais aussi à l'aborder sous des angles différents afin de comprendre comment y implanter des lois de probabilités pour le rendre d'autant plus ludique. L'intéractivité offerte à l'utilisateur via un contrôle sur certains paramètres lui permet de visualiser l'impact qu'ils ont au cours du jeu et l'oblige potentiellement à adopter de nouvelles stratégies pour résoudre le puzzle.