46196 WOZNY Sylvain

MPI

Échanges entre des piles de cartes

Je suis très intéressé par les jeux de société. De plus, la carte à jouer est un des principaux supports du jeu de société et comme connaître la composition des piles de cartes procure un grand avantage, cela m'a incité à étudier leur évolution au cours d'un ou plusieurs échanges.

Sachant que de très nombreux jeux se basent sur les cartes à jouer, étudier les échanges de cartes entre les piles permet de simuler certains jeux de sociétés simples. De plus, des algorithmes décrivant l'état du jeu peuvent aider à faire les bons choix d'action, comme une application compagnon.

Positionnement thématique (ÉTAPE 1) :

- *MATHEMATIQUES (Autres)*
- *INFORMATIQUE (Informatique pratique)*
- *INFORMATIQUE (Informatique Théorique)*

Mots-clés (ÉTAPE 2)

Mots-clés (en français)	**Mots-clés (en anglais)**
Chaîne de Markov	*Markov chain*
Cartes à jouer	*Playing cards*
Graphes	*Graphs*
Matrice d'adjacence	*Adjacency matrix*
Apprentissage non-supervisé	*Unsupervised learning*

Bibliographie commentée

Depuis les premiers jeux, jusqu'à aujourd'hui, la plupart s'appuient sur divers supports, et l'un des plus répandus est la carte à jouer. En effet, elle comporte plusieurs intérêts quant aux informations qu'elle transmet ou à l'inverse qu'elle ne transmet pas. Les informations au recto : symboles et valeurs, permettent de distinguer plusieurs types de carte et de les hiérarchiser. Le verso permet de dissimuler la nature de la carte et ainsi de permettre des tirages aléatoires et crée une dimension de secret. De plus, ce sont les règles du jeu, qui attribuent aux symboles et

valeurs leurs significations et le choix des informations pertinentes sur carte revient lui aussi aux règles du jeu. C'est ce qu'explique [3]. De ce fait, on peut adapter les informations transportées par les cartes à la convenance du TIPE, et ainsi simplifier la modélisation.

En premier lieu, pour déterminer au cours du temps les proportions de chaque type de carte dans les différentes piles, on peut représenter les cartes et leurs échanges comme s'ils étaient réalisés physiquement donne un résultat expérimental identique au réel. Cependant, cela permet la répétabilité de l'expérience ainsi qu'une exécution bien plus rapide, même avec de grandes masses de données. Mais il y a une limite à cette représentation : comme il faut exploiter des résultats instables de plusieurs expériences, cela ne donne pas de représentation mathématique de la situation. Heureusement, d'autres méthodes existent comme l'utilisation de graphes.

Pour les décrire, on peut notamment introduire une représentation matricielle des échanges, ce qui est présenté de plusieurs manières dans [1] avec la matrice de probabilité (8.2) et la matrice de taux de transitions (8.8). On peut alors adapter cette représentation à la composition de différentes piles de cartes plutôt qu'à des états de fonctionnement d'un système.

Cette méthode de représentation a le grand intérêt de permettre de considérer, dans certains cas, comme des chaînes de Markov. On les définit comme dans [2], ce qui revient à discrétiser le temps continu utilisé dans [1] pour ressembler à des tours de jeux. En utilisant les propriétés des chaînes de Markov décrites dans le cours [2] on peut simplifier les calculs. Notamment en utilisant la Loi Marginale (1.6) qui a pour conséquence que les opérations à effectuer, au final, sont des exponentiations de matrices. Pour que ces représentations fonctionnent, il faut créer une matrice de transition adaptée (Caractère stochastique décrit par [1] et [2]). Sachant que les piles n'échangent pas avec elles-mêmes, il n'y a pas d'interactions internes entre les types de cartes d'une pile. Cela donne une matrice réduite, et de plus, composée de blocs carrés de mêmes tailles, tous diagonaux.

Ces multiples propriétés permettent l'optimisation de leur exponentiation. En s'aidant de la bibliothèque Numpy de Python [5], on peut créer, de manière compacte et claire, un programme de calcul d'exponentiation rapide de matrice respectant les propriétés précédentes.

Enfin, pour exploiter les résultats obtenus grâce aux chaînes de Markov, on s'intéresse aux positions d'équilibre ou plus généralement aux d'états récurrents (2.4) expliqués dans [2]. Ainsi, d'autres théorèmes assurent l'existence d'au moins un état récurrent (2.11).

Pour s'en persuader expérimentalement, on utilise Python avec MatPlotLib.pyplot pour générer des graphiques de l'évolution de la composition des piles au fur et à mesure des échanges, ou encore tracer des courbes de moyennes sur l'ensemble des piles ou sur plusieurs expériences successives pour observer des tendances. Cependant, cette représentation est limitée et nécessite des améliorations quand le nombre de piles et de types de cartes augmente. Pour cela, l' utilisation d'une dimension de temps est très utile. C'est ce que fait la bibliothèque Imageio en créant des gifs à partir des images générées par MatPlotLib.pyplot. En utilisant les instructions du [4] on peut alors représenter un nombre de types et de piles de carte beaucoup plus grand, et

même le nombre d'échange n'est plus un problème, car il ne fait que prolonger de quelques images le gif.

Problématique retenue

Comment représenter l'évolution de la composition de plusieurs piles de cartes au cours du temps ?

Objectifs du TIPE du candidat

- Étudier l'évolution de la proportion d'un type de carte dans un ensemble de piles.

- Pouvoir représenter visuellement cette évolution dans différents cas de complexité croissante.

- Utiliser diverses méthodes, graduellement évoluées, pour modéliser puis simuler les échanges.

- Utiliser Numpy pour clarifier les opérations et optimiser l'exécution.

- Reconnaître le caractère markovien de la simulation et en utiliser les propriétés.

Références bibliographiques (ÉTAPE 1)

[1] MARVIN RAUSAND ET ARNLJOT HOYLAND : System Reliability Theory Model, Statistical Methods, and Applications (Chapitre 8 Markov Processus) *: Second Edition, Wiley (2003), ISBN 9780471471332*

[2] PIERRE-LOIC MELIOT : Chaînes de Markov : théorie et applications *: IMO université Paris-Saclay, https://www.imo.universite-paris-saclay.fr/~pierre-loic.meliot/markov/markov.pdf (01/06/23 - ...)*

[3] MIGUEL ROTENBERG : Les Jeux de Société, essai sur la production d'un outil d'analyse autour des mécaniques de jeu, (Chapitre 6 : Mécaniques en lien avec le matériel du jeu) (6.1.4 La carte) *: https://www.ville-jeux.com/IMG/pdf/m2_-_mecaiques_des_jeux_de_societe.pdf (17/02/2023)*

[4] IMAGEIO CONTRIBUTORS : User guide *: https://imageio.readthedocs.io/en/stable/user_guide/index.html*

[5] NUMPY DEVELOPERS : Numpy.org *: https://numpy.org/doc/stable/reference/arrays.html*

Références bibliographiques (ÉTAPE 2)

[1] MARVIN RAUSAND ET ARNLJOT HOYLAND : System Reliability Theory Model, Statistical Methods, and Applications (Chapitre 8 Markov Processus) *: Second Edition, Wiley (2003), ISBN 9780471471332*

[2] PIERRE-LOIC MELIOT : Chaînes de Markov : théorie et applications *: IMO université Paris-Saclay, https://www.imo.universite-paris-saclay.fr/~pierre-loic.meliot/markov/markov.pdf (Date de téléchargement : 01/06/23)*

[3] MIGUEL ROTENBERG : Les Jeux de Société, essai sur la production d'un outil d'analyse autour des mécaniques de jeu, (Chapitre 6 : Mécaniques en lien avec le matériel du jeu) (6.1.4 La carte) *: https://www.ville-jeux.com/IMG/pdf/m2_-_mecaiques_des_jeux_de_societe.pdf (Date de téléchargement : 17/02/2023)*

[4] IMAGEIO CONTRIBUTORS : User guide *: https://imageio.readthedocs.io/en/stable/user_guide/index.html*

[5] NUMPY DEVELOPERS : Numpy.org *: https://numpy.org/doc/stable/reference/arrays.html*

DOT

[1] *: [Février 2023] Cerclage du problème et établissement des premières règles sur la méthode d'échange et la forme des matrices. Ainsi qu'une recherche documentaire sur les conventions pour décrire les mécanismes de jeu, mais aucune information convaincante n'en sortira.*

[2] *: [Mars 2023] Première représentation par modélisation et manipulation des histogrammes de matplotlib.pyplot.*

[3] *: [Avril - Mai 2023] Transition vers la simulation avec une première tentative en utilisant les complexes, puis des matrices.*

[4] *: [Juin - Septembre 2023] Découverte de la librairie imageio, pour générer des gifs, et utilisation poussée des graphiques en barres de matplotlib.pyplot. Application aux programmes de simulation. Questionnement sur la manière de représenter comme un processus de Markov la situation.*

[5] *: [Octobre - Décembre 2023] Première tentative infructueuse à cause d'une erreur théorique sur la création de la matrice de relation. Celle-ci sera considérée comme une erreur d'arrondis, ce qui poussera à chercher plus de précision dans les calculs grâce aux entiers infinis.*

[6] *: [Janvier - Février 2024] Représentation sous forme de processus de Markov avec une méthode différente, correcte et plus efficace. Application de la génération de gifs*

[7] *: [Mars - Avril 2024] Recherche de méthodes pour trouver des équilibres et choix des algorithmes au programme pour les réaliser (Kosaraju-Sharir et Classification hiérarchique ascendante).*

[8] *: [Mai 2024] Choix du cas d'application et fil rouge de la présentation (Pouilleux simplifié) et utilisation des outils précédemment créés.*