Projet Python : jeu d'échecs

E. Kritsikis
IUT de Villetaneuse

1 Introduction

Le jeu d'échecs a probablement été inventé en Inde sous l'empire Gupta. Il se répand en Perse puis en Europe vers l'an mil. Des jeux apparentés existent ailleurs en Asie comme le *xiangqi* chinois, le *shogi* japonais ou le *makruk* thaï. Au 15e s. les européens introduisent des pièces plus puissantes comme la dame qui remplace le vizir. Le premier tournoi international se joue à Londres en 1851. Voir aussi l'histoire du café de la Régence. Le bloc soviétique se distingue au 20e s. par des structures publiques d'entraînement et compétition. On relève actuellement trois tendances :

— le jeu occidental gagne du terrain en Asie, notamment en Chine, en Inde et en Iran ;

— le succès du jeu en ligne avec p.ex. le site libre lichess de Thibault Duplessis et sur twitch ;

— si les ordinateurs dominent dès le début du siècle, des intelligences artificielles comme Leela apprennent à jouer seules et tirent avantage de la puissance des cartes graphiques. Le plus fort "joueur" du monde en 2020 est l'hybride Stockfish, libre bien sûr.

2 But

Le but est de programmer en Python une interface de jeu où l'utilisateur entre les coups, l'ordinateur vérifie qu'ils sont légaux et affiche la position sur l'échiquier, puis répond jusqu'à ce qu'un des joueurs perde son roi ou veuille arrêter.

Au début chaque camp a 8 pions et 8 pièces supérieures ou figures : 2 tours, 2 cavaliers, 2 fous, une dame et un roi. On peut trouver les règles de déplacement sur wikipédia. Dans les règles officielles, un joueur ne peut mettre ou laisser son roi en échec (c.à.d. sous la menace d'une pièce adverse) mais on ignore cela, on cherche simplement à manger le roi.

3 Structures de données

On représentera l'échiquier par un tableau `ech` de 64 entiers. Un zéro est une case vide ; pour le reste on prend une convention comme 1 : pion blanc, 2 : tour blanche, ... et ainsi de suite. On aura deux autres tableaux qui seront la liste des pièces blanches et noires respectivement. On représente chaque pièce par un dictionnaire à trois entrées. La première est le type de pièce : `'p'` pour pion p.ex. La deuxième est la case où elle se trouve, p.ex. `"c4"` : la colonne est représentée par une lettre qui va d'`a` à `h`, la ligne ou rangée par un chiffre d'1 à 8. La troisième entrée du dictionnaire est le tableau des cases que la pièce peut atteindre. Il faut l'actualiser à chaque coup, de même que la localisation. Quand une pièce est mangée, on la supprime de la liste de son camp.

4 Modules

Je vous conseille de structurer votre programme en cinq modules.

4.1 Module `case`

Contient des utilitaires pour calculer si une case existe, est vide ou occupée par une pièce noire ou blanche, ainsi que son numéro dans le tableau `ech`. Il est utile de convertir une case, la chaîne `"c4"` p.ex. en colonne 3 et rangée 4 et inversement, par les fonctions `ord` et `chr`.

4.2 Module `init`

Remplit les tableaux pour commencer à jouer. On y écrira une fonction `saisie_pieces` qui permet à l'utilisateur de saisir une position de départ, p.ex. la chaîne `rg1 f2 g2 h2` pour un roi en g1, des pions en f2, g2, h2 ; puis `rg7` mettons pour les noirs. Cela permettra de tester les déplacements avec peu de pièces. Puis une autre fonction `posit` qui place toutes les pièces dans la position initiale standard.

4.3 Module `affiche`

Permet l'affichage de la position actuelle, avec nom des colonnes et rangées. On cherchera les symboles Unicode des pièces pour l'affichage dans le terminal. Pour les couleurs on peut utiliser le module `colored`. On pourra mettre le dernier coup en évidence par une autre couleur. Prévoir aussi d'afficher l'échiquier à l'envers, du côté noir. La rangée 8 sera alors en bas.

4.4 Module `coups`

Recense les coups de chaque pièce. On constate qu'elles se divisent en trois catégories :

— les pièces à longue portée : tour, fou et dame. Les cases disponibles sont les cases vides dans chaque direction, orthogonale ou diagonale, et la première case non vide si une pièce adverse s'y trouve.

— roi et cavalier ont accès à 8 cases proches, si ces cases existent et ne sont pas occupées par le même camp. J'ai ignoré le roque.

— les pions. Il faut distinguer les prises en diagonale et l'avancée (un double pas est possible au départ). Attention, pions noirs et blancs n'avancent pas dans le même sens. De plus, quand un pion arrive en fin de carrière (rangée 8 pour les blancs, 1 pour les noirs), une promotion change son type. On peut considérer pour simplifier qu'il est damé. (En réalité on a le choix dans le type de promotion.)

Une fonction `actu` examine les pièces dans chaque camp après chaque coup et appelle une sous-fonction correspondant au type de pièce, qui renvoie la liste des coups possibles. Ainsi on peut actualiser le tableau `piece["coups"]`.

4.5 Module `echecs`

Il contient le programme principal qu'on exécute pour jouer. Une fonction `partie` lance la partie grâce au module `init` et à `coups.actu`, demande au joueur de choisir sa couleur et entre dans une boucle qu'on peut quitter en tapant p.ex. `q`. Autrement, après affichage de la position, le joueur saisit son coup, p.ex. `e2e4` et le programme vérifie sa légalité, en consultant le tableau des pièces et leurs destinations possibles. Au besoin la saisie est répétée. Si c'est au tour de l'ordinateur, il choisit un coup au hasard (parmi les coups légaux). On modifie le tableau `ech` et la liste des pièces (position, suppression éventuelle et actualisation des coups possibles). Quand on mange le roi, la `partie` renvoie une valeur pour savoir qui a gagné.

Bonne chance.