Projet IN104 (2025) - Programmation d'une IA pour le jeu d'échecs en C

Encadrant : Tom Boumba

Fiche 5 : Perft

Tout moteur d'échecs dispose d'un module de test de performance appelé Perft (Performance Test by move path enumeration) permettant à la fois d'évaluer la performance du moteur en mesurant sa vitesse de parcours de l'arbre des coups possibles à partir d'une position donnée, et de vérifier le bon fonctionnement de sa fonctionnalité de génération des coups. Concrètement, en comparant le nombre de positions atteintes par le moteur à une certaine profondeur donnée avec celui atteint par un autre moteur que l'on considère de confiance, on peut déterminer si notre moteur "oublie" des coups ou s'il en "rajoute".

Implémentation

Dans le fichier `perft.h`, on a défini un compteur global long nb_feuilles, qui compte le nombre de feuilles de l'arbre du jeu actuellement parcourues pour la profondeur fixée. La fonction cross-plateforme `int get_time_ms()` permet de récupérer le temps actuel en milisecondes.

- Implémenter la fonction récursive `void perft(int profondeur)`, qui permet de parcourir en profondeur l'arbre du jeu, et compte chaque feuille atteinte pour la profondeur voulue.
- Implémenter la fonction `void exec_perft(int profondeur)`, qui permet de lancer le test de performance avec une profondeur d'entrée donnée, et affiche ses résultats.
- Implémenter aussi la fonction `void test_perft(int profondeur)`, qui permet d'obtenir des résultats plus détaillés du test de performance, en affichant le résultat du test Perft après exécution de chaque coup légal disponible dans la position actuelle. Dans le cas où le test Perft global indique que notre moteur oublie ou rajoute des coups, cette fonction vous permettra de débugger en indiquant à la suite de quel coup possible actuel ces coups sont omis ou ajoutés.

– Vous effectuerez différents tests Perft afin d'évaluer votre moteur, à comparer par exemple avec ceux de cette page (ici le test est effectué pour la position 2) :

```
generer_masques_d_attaque_statiques();
parse_fen(tricky_position);
print_echiquier();
exec_perft(4);
```

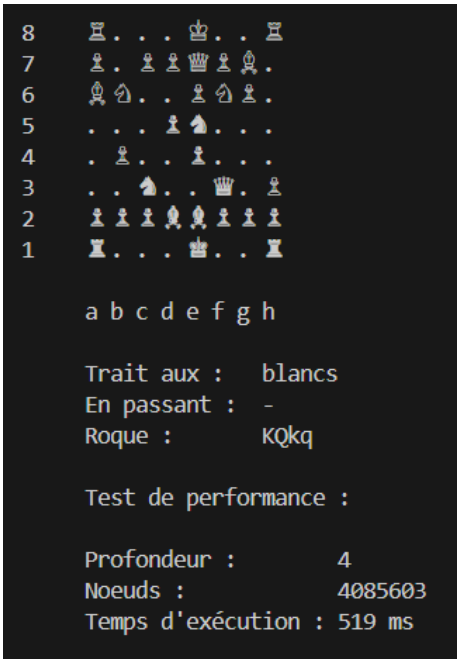

Figure 1: Perft à la profondeur 4 pour une position de l'échiquier complexe.

```
parse_fen(position_de_depart);
print_echiquier();
test_perft(2);
```

```
Test PERFT détaillé :

coup : a2a3       feuilles : 20
coup : a2a4       feuilles : 20
coup : b2b3       feuilles : 20
coup : b2b4       feuilles : 20
coup : c2c3       feuilles : 20
coup : c2c4       feuilles : 20
coup : d2d3       feuilles : 20
coup : d2d4       feuilles : 20
coup : e2e3       feuilles : 20
coup : e2e4       feuilles : 20
coup : f2f3       feuilles : 20
coup : f2f4       feuilles : 20
coup : g2g3       feuilles : 20
coup : g2g4       feuilles : 20
coup : h2h3       feuilles : 20
coup : h2h4       feuilles : 20
coup : b1a3n      feuilles : 20
coup : b1c3n      feuilles : 20
coup : g1f3n      feuilles : 20
coup : g1h3n      feuilles : 20

Profondeur :       2
Noeuds :           400
Temps d'exécution : 0 ms
```

Figure 2: Perft détaillé à la profondeur 2 pour la position initiale.