ChessAInt

PROJET INFORMATIQUE

MEMBRES: A.LESCOUET, H.GANGLOFF, B.BAZARD, L.CRELIER

TUTEUR: DANIEL RANC

Cahier des charges :

Problématique :

- Comment coder un moteur d'échecs pouvant communiquer avec une interface utilisateur extérieure en utilisant le protocole UCI ?

Solution apportée :

- Gestion du protocole UCI
- Recherche du meilleur coup dans un arbre dont les nœuds représentent les différentes configurations de l'échiquier possibles, et les affecter d'un score déterminé par une heuristique
- Utilisation de A* pour effectuer cette recherche

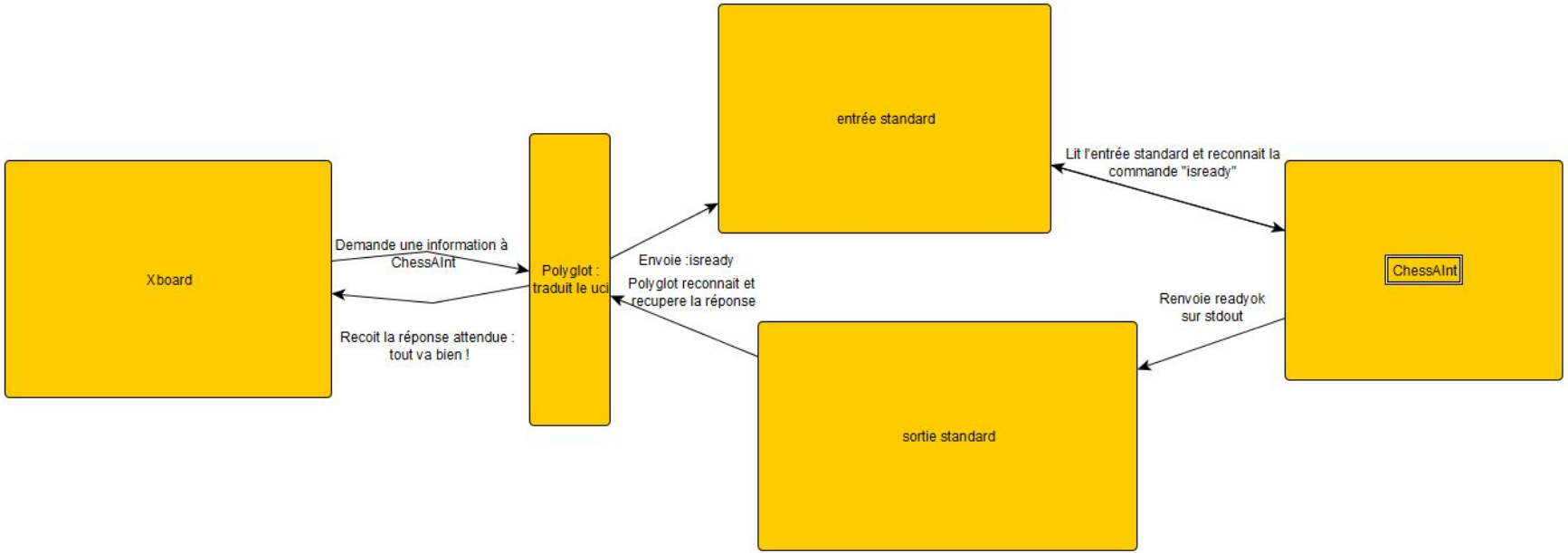
Plan

- Introduction
- Partie I : Le fonctionnement d'UCI
- Partie II : Les structures de données utilisées
- Partie III : Comment générer tous les coups possibles d'un échiquier
- Partie IV : Déterminer la pertinence d'un coup
- Partie V : Application de l'algorithme A* à notre projet
- Partie VI : Perspectives d'améliorations : les bibliothèques d'ouvertures
- Partie VII : Comment avons-nous fonctionné

I) Le protocole UCI : Son fonctionnement

- L'interface graphique a un rôle de maître, le moteur ne peut prendre aucune initiative
- Elle établit un pipe entre le gui et le moteur
- Le gui écrit sur l'entre standard des strings standardisées que le moteur lit, et le moteur renvoie sur la sortie standard à l'intention du gui

I) Le protocole UCI : Exemple



Partie II : Les différentes structures de données utilisées

- Chessboard
- Graph

II) Structures de données: le stockage des parties

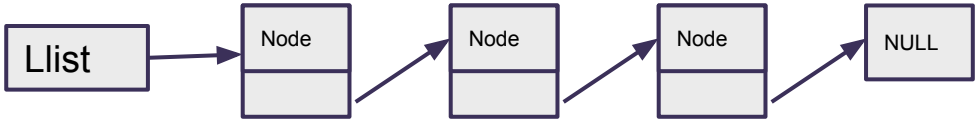
- Structure : Board
  - Cette structure donnée, contient toutes les infos pour continuer à jouer immédiatement
    - square[8][8]
      - piece, color
    - activeColor
    - …

- Structure manipulée dans tout le projet

II) Structures de données: Comment les graphes sont implémentés

Les graph sont composés principalement de :

- Une liste chaînée



- Un tableau principal
- Un tableau temporaire

II) Structures de données: Utilisation

L'insertion dans la liste se fait triée.

Le tableau principal contient l'état de l'échiquier avant la recherche d'astar

Le tableau temporaire est modifié à chaque appel de la fonction de génération des noeuds.

Partie III: Générer les coups à partir d'une position donnée

- Fonctionnement général
- Légalité des coups : la gestion des positions d'échecs

III) Générer les coups suivants : Comment est-ce que cela fonctionne ?

Parcours de tout l'échiquier

- Appel à des sous-fonctions de mouvement
  - Génération sur une ligne
  - Génération au case par case
    - A l'aide d'incréments
- On stocke les coups au fur et à mesure

```
if (!thereIsNoKing && (isThreatened(kingX, kingY, threats))) {
  stopThreat(stack, board, pinned, threats, kingX, kingY);

  kingMoveGenerator(stack, kingX,  kingY, board.activeColor, board, threats);
} else {
  for (int j = 0 ; j < ROWCOL_NB ; ++j) {
    for (int i = 0 ; i < ROWCOL_NB ; ++i) {
      if (board.square[i][j].color == board.activeColor && !pinned[i][j]) {
        switch (board.square[i][j].piece) {
        case pawn:
          pawnMoveGeneratorCapture(stack, i, j, board.activeColor, board);
          pawnMoveGeneratorNoCapture(stack, i, j, board.activeColor, board);
          break;
        case bishop:
          bishopMoveGenerator(stack, i, j, board.activeColor, board);
          break;
        case knight:
          knightMoveGenerator(stack, i, j, board.activeColor, board);
          break;
        case rook:
          rookMoveGenerator(stack, i,  j, board.activeColor, board);
          break;
        case queen:
          queenMoveGenerator(stack, i, j, board.activeColor, board);
          break;
        case king:
          kingMoveGenerator(stack, i,  j, board.activeColor, board, threats);
          break;
        case empty:
          break;
```

III) Générer les coups suivants : Le problème de la légalité

- Définition de la légalité aux échecs
  - Priorité absolue à la défense de son roi
    - Variation dans la technique de génération de mouvements
- 3 structures pour répondre au problème
  - L'échiquier
  - Le tableau des menaces
  - Le tableau des pièces clouées

Partie IV : évaluation du score d'une configuration de l'échiquier

- Trois critères d'évaluation :
- Chaque pièce possède un score de base qui représente son utilité
- Une pièce protégée voit son score augmenter proportionnellement au score de base
- Une pièce menacée voit son score diminuer proportionnellement au score de base

Partie IV : L'attribution du score de base

Le score de base est attribué (et est fixe) selon l'utilité moyenne de la pièce.

Les scores pour l'instant sont les suivants :

Roi	Dame	Fou	Cavalier	Tour	Pion
0	70	30	30	50	10

IV) Heuristique : Protection et menace

Les protections et menaces modifient le score :

Permet d'implémenter un style de jeu basique où le joueur protège ses pièces et tente de prendre celles de l'adversaire.

Problème : le roi est un pièce particulière :

Au départ le roi avait une valeur importante

Roi valeur à 0 ensuite

Partie V : Adaptation de l'A* à notre problème

Il s'agissait d'utiliser une version adaptée de l'algorithme A-star pour permettre la recherche d'un coup optimal.

Recherche dans un graphe

Déroulement de l'algorithme

V) L'A* de ChessAInt : le meilleur coup d'un graphe

L'algorithme A-star tel que décrit dans l'état de l'art consiste à travailler sur deux listes, nommées open set and closed set.

Création d'une structure graph spéciale optimisée pour A-star.

Une liste avec insertion triée permet d'avoir toujours à disposition le meilleur noeud.

Sert à la fois d'open et closed set.

V) L'A* de ChessAInt : Calcul

- Choix du meilleur noeud, ie le premier de la liste
- Génération des noeuds fils à l'aide de movesGenerator
- Ajout trié des noeuds dans la liste chaînée
- On recommence tant que l'une des trois conditions de sortie n'est pas respectée :
  - Temps alloué dépassé
  - Score demandé atteint
  - Profondeur maximale de recherche atteinte pour tous les noeuds

V) L'A* de ChessAInt : Gestion de l'alternance Blanc/Noir

Lors de la génération des noeuds :

- Seul le meilleur noeud blanc est considéré (on suppose que le joueur adverse joue de manière optimale).
- Le score d'une branche est constitué de la différence alternée des scores de tous les noeuds qui la composent.

Partie VI : Perspectives d'améliorations

- Bibliothèque d'ouverture & de finale
- Amélioration de l'heuristique
- Gestion de coups spéciaux
- UCI

Bibliothèque d'ouverture

- Fichiers au format spécifique : pgn trié
- Optimiser les premier coup de ChessAInt
- Format : "1.B N\n2. …"
- On choisit le meilleur ratio victoire/défaite

```
[Event "F/S Return Match"]
[Site "Belgrade, Serbia JUG"]
[Date "1992.11.04"]
[Round "29"]
[White "Fischer, Robert J."]
[Black "Spassky, Boris V."]
[Result "1/2-1/2"]

1. e4 e5 2. Nf3 Nc6 3. Bb5 a6 4. Ba4 Nf6 5. O-O Be7 6. Re1 b5 7. Bb3 d6 8. c3
O-O 9. h3 Nb8 10. d4 Nbd7 11. c4 c6 12. cxb5 axb5 13. Nc3 Bb7 14. Bg5 b4 15.
Nb1 h6 16. Bh4 c5 17. dxe5 Nxe4 18. Bxe7 Qxe7 19. exd6 Qf6 20. Nbd2 Nxd6 21.
Nc4 Nxc4 22. Bxc4 Nb6 23. Ne5 Rae8 24. Bxf7+ Rxf7 25. Nxf7 Rxe1+ 26. Qxe1 Kxf7
27. Qe3 Qg5 28. Qxg5 hxg5 29. b3 Ke6 30. a3 Kd6 31. axb4 cxb4 32. Ra5 Nd5 33.
f3 Bc8 34. Kf2 Bf5 35. Ra7 g6 36. Ra6+ Kc5 37. Ke1 Nf4 38. g3 Nxh3 39. Kd2 Kb5
40. Rd6 Kc5 41. Ra6 Nf2 42. g4 Bd3 43. Re6 1/2-1/2
```

```
1.d4 d5
2.Nc3 e6
3.e4 Bb4
4.Bd2 dxe4
5.Qg4 Qxd4
6.O-O-O f5
7.Bg5 Qxf2
8.Qh5 Be7
9.Kb1 Bd7
1/2-1/2
1.d4 d5
2.c4 e6
3.Nc3 Nf6
```

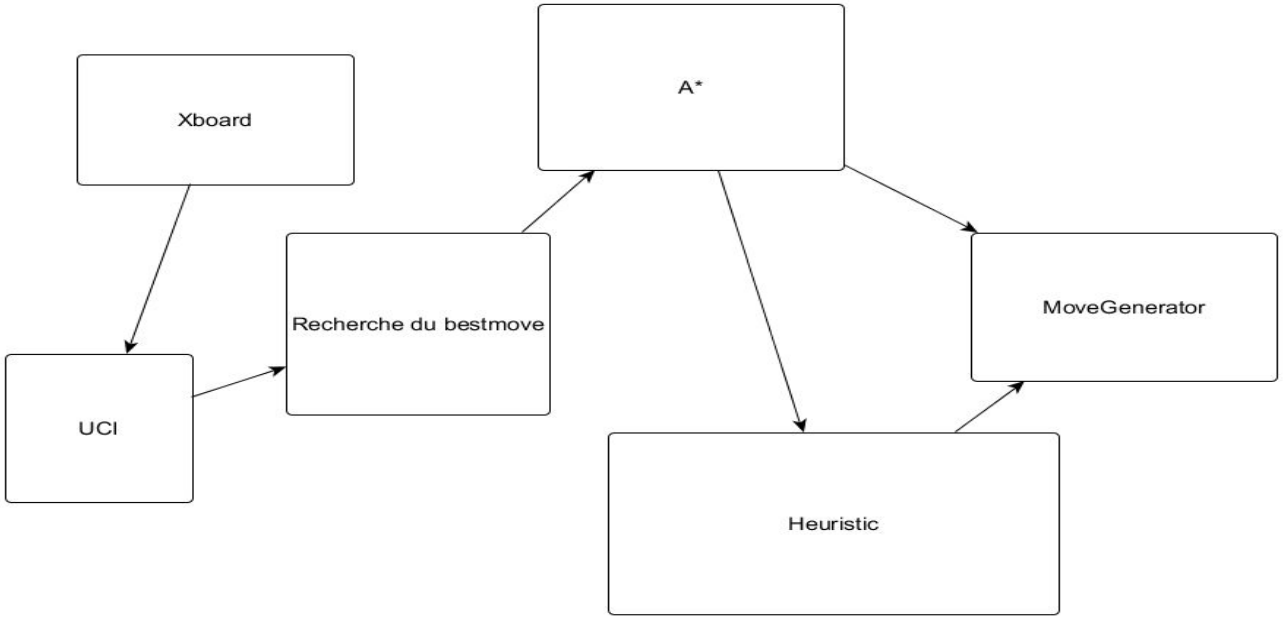
Bibliothèque d'ouverture

- Parsage pour les coups noirs : entre " " et "/n"
- On place les coups dans un tableau trié
- On retourne le coup ayant le meilleur ratio

VII) Gestion de projet : logiciels utilisés

Fonction	Langage	Organisation du travail	Documentation	Profilage	Social	Lint	Débogage
Outil(s) utilisé(s)	C	Subversion Planner	Doxygen	Valgrind	Skype	cpplint.py	gdb

Récapitulation des liens entre modules



Conclusion

- Avantage du travail en équipe
- Importance de la gestion du temps
- Regrets sur certaines parties laissées pour compte

QUESTIONS

- DES QUESTIONS ?

Le mot de la fin

Longue vie à ChessAInt !